UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:
INGENIERÍA DE SISTEMAS

Trabajo de titulación previo a la obtención del título de:
Ingenieros de Sistemas

TEMA:
ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE GESTIÓN DE INVENTARIOS DE ACTIVOS DE TI PARA LOS LABORATORIOS DE LA CARRERA DE INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN DE LA UNIVERSIDAD POLITÉCNICA SALESIANA SEDE QUITO CAMPUS SUR

AUTORES:
DANIEL ANDRES CHISAGUANO CASTELLANO
MILTON JOEL LUDEÑA VELIZ

TUTOR:
DANIEL GIOVANNY DÍAZ ORTIZ

Quito, febrero del 2019

CESIÓN DE DERECHOS DE AUTOR

Nosotros, Daniel Andres Chisaguano Castellano, con documento de identificación N° 1725359945, y Milton Joel Ludeña Veliz, con documento de identificación N° 1725511552, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de titulación con el tema: ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE GESTIÓN DE INVENTARIOS DE ACTIVOS DE TI PARA LOS LABORATORIOS DE LA CARRERA DE INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN DE LA UNIVERSIDAD POLITÉCNICA SALESIANA SEDE QUITO CAMPUS SUR., mismo que ha sido desarrollado para optar por el título de INGENIEROS DE SISTEMAS en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en nuestra condición de autores nos reservamos los derechos morales de la obra antes citada.

En concordancia, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.

.....

CHISAGUANO CASTELLANO
DANIEL ANDRES
CI: 1725359945

.....

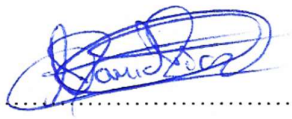
LUDEÑA VELIZ
MILTON JOEL
CI: 1725511552

Quito, febrero 2019

DECLARATORIA DE COAUTORÍA DEL TUTOR

Yo declaro que bajo mi dirección y asesoría fue desarrollado el Proyecto Técnico, con el tema: ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE GESTIÓN DE INVENTARIOS DE ACTIVOS DE TI PARA LOS LABORATORIOS DE LA CARRERA DE INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN DE LA UNIVERSIDAD POLITÉCNICA SALESIANA SEDE QUITO CAMPUS SUR., realizado por Daniel Andres Chisaguano Castellano y Milton Joel Ludeña Veliz, obteniendo un producto que cumple con todos los requisitos por la Universidad Politécnica Salesiana, para ser considerados como trabajo final de titulación.

Quito, febrero 2019



.....

DANIEL GIOVANNY DÍAZ ORTIZ
CI. 1716975501

DEDICATORIA

Dedico el presente trabajo de titulación en primer lugar a Dios por haberme guiado con su sabiduría, el conocimiento necesario, por todas las bendiciones y por la valentía que me brindo para enfrentar todas las circunstancias que se me presento en mi etapa universitaria.

A mi padre Silverio Chisaguano quien siempre fue un ejemplo de lucha y superación constante y estuvo a mi lado brindándome su apoyo en los momentos difíciles de mi vida para no permitir que mis sueños mueran.

A mi madre Fanny Castellano que me enseño que en la vida hay que ser valientes y que no importan las circunstancias, hay que afrontarlas y continuar. Todo su cariño y comprensión se ven reflejados en este logro.

A mis hermanos Pablo y Ana, mis sobrinas Alejandra y Milena y mi cuñado Leonardo, quienes con su amor y palabras de ánimo me impulsaron a superarme cada día.

A mis familiares y amigos que me brindaron su amistad, gratos momentos y su compañía durante mi carrera universitaria.

A mi amigo Joel Ludeña por su amistad, los malos y buenos momentos que hemos pasado en esta etapa y a su familia quienes calurosamente nos acompañaron durante este proceso de titulación.

Daniel Andres Chisaguano Castellano

Dedicado a Dios, quien con sus bendiciones me ha sabido ayudar a superar cada obstáculo presentado y así permitirme llegar a esta instancia de mi vida profesional.

A mis padres, Milton Ludeña y Patricia Veliz, que, gracias a sus innumerables sacrificios y apoyo incondicional, me han permitido conocer el significado de trabajo duro y perseverancia, permitiéndome así culminar con mis estudios universitarios.

A mi hermano Elian, y a toda mi familia por estar presentes en los momentos de mayor dificultad, brindándome consejo y respaldo para cumplir con mis metas propuestas.

A mis amigos de la Universidad, que de alguna u otra forma supieron acompañarme y guiarme en todo este trayecto, en especial a Andres Chisaguano con quien compartí buenos y gratos momentos durante la realización de este proyecto.

Milton Joel Ludeña Veliz

AGRADECIMIENTOS

A Dios por brindarnos la vida, la salud y los conocimientos necesarios para la culminación de este proyecto.

A nuestros padres por su constante comprensión, paciencia, trabajo y apoyo durante todo el tiempo de nuestra formación personal y profesional.

A la Universidad Politécnica Salesiana, por brindarnos todos los recursos necesarios para la elaboración del presente proyecto y por su excelente formación académica y humana que nos han brindado.

Al Ingeniero Jorge López por impulsarnos a desarrollar un proyecto innovador y por proporcionarnos la guía necesaria para la elaboración del mismo.

A la Ingeniera Marcela Gallegos y a la Ingeniera Thalía Ati, quienes desinteresadamente brindaron el tiempo y la información necesaria para el análisis y diseño del proyecto.

A nuestro tutor Ing. Daniel Díaz por el compañerismo, la guía y los consejos proporcionados durante el transcurso del proyecto.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El presente documento tiene como objetivo detallar el proceso de construcción de un sistema de gestión de inventarios y ticketing para soporte técnico, como una herramienta que permita agilizar el control de inventarios y dar respuesta inmediata a las incidencias producidas en los recursos del Data Center y laboratorios de la carrera de Ingeniería en Ciencias de la Computación.

El sistema desarrollado cuenta con una aplicación Web la cual permite gestionar activos, máquinas virtuales, tickets, roles, usuarios y generación de reportes. También cuenta con una aplicación móvil Android la cual permite la generación de tickets para soporte técnico de activos de TI y laboratorios, además de la consulta y modificación de activos de TI por código QR.

El documento se estructura de la siguiente manera:

El capítulo uno, en donde se detalla la información referente a la organización y una breve explicación sobre los conceptos utilizados para el desarrollo del sistema.

El capítulo dos, en donde se realiza el análisis y diseño de los procesos, análisis de requerimientos y el diseño conceptual del sistema.

El capítulo tres, en donde se detalla la construcción del sistema, implementación, pruebas de funcionamiento y análisis de los resultados obtenidos.

Al final del documento, se encuentran las conclusiones y recomendaciones del proyecto.

ABSTRACT

The purpose of this document is to detail the process of developing an inventory management and ticketing for technical support system, as a tool that allows the inventories control and provide a fast response to the incidents that are produced in the resources of the Data Center and laboratories of Engineering in Computer Science career.

The developed system has a Web application that allows managing assets, virtual machines, tickets, roles, users and reports generation. It also has an Android application that allows the IT assets and laboratories tickets generation for technical support, as well as IT assets queries and updates by QR code.

The structure of the document is as follows:

Chapter one, in which the organization information is detailed and a brief explanation about the concepts used for the system development.

Chapter two, in which the analysis and processes design, requirements analysis and the system conceptual design are carried out.

Chapter three, in which is detailed the system development, implementation, performance tests and the results analysis obtained.

At the end of the document, there are the conclusions and recommendations of the project.

INTRODUCCIÓN

Antecedentes

En la actualidad, los Data Center se han convertido en un activo estratégico fundamental en cualquier organización centrada en las TI, debido a que, entre las innumerables ventajas, facilita la gestión y el acceso a aplicaciones distribuidas, así como también las diversas alternativas para la conectividad, que van desde conexiones privadas o plataformas en la nube. La constante demanda de recursos y la necesidad de los procesos del negocio para estar siempre operativos y disponibles exigen a los Data Centers que se adapten y respondan a los requerimientos a un ritmo cada vez más acelerado. Por estas razones, su apropiada administración y mantenimiento mediante aplicaciones para la organización propietaria como para los responsables de su funcionamiento. (Jiménez, 2017)

Dentro de lo que conlleva la administración de un Data Center, es imprescindible contar con sistemas, herramientas o estándares que faciliten la gestión de todos los activos y recursos que se requieran en una infraestructura de este tipo. El uso de hojas de cálculo como control de inventarios presenta una desventaja con respecto a una aplicación desarrollada para esta finalidad, debido a que son estáticas y no presentan la información actualizada. (Pacio, 2013)

En la actualidad existen diversas herramientas que permiten sobrellevar esta tarea, sin embargo, la desventaja se encuentra en que estas difícilmente cumplirán con todas las necesidades que requiere la gestión de activos de TI de un Data Center en particular, debido a que cada organización cuenta con sus propios parámetros.

Actualmente, el proceso de inventarios en el Data Center y laboratorios de la carrera de Ingeniería en Ciencias de la Computación de la Universidad Politécnica Salesiana, sede Quito campus Sur, es llevado en hojas de cálculo Excel, en donde cada activo

de TI es registrado con sus características propias. Un software de gestión de inventarios desarrollado a medida para este tipo de infraestructura se convierte en la mejor opción para la ejecución de este proceso.

El objetivo principal de un software de gestión de inventarios es tener el control centralizado de los recursos, generar reportes, mantener la información actualizada, y tomar decisiones que permitan optimizar los recursos, generando ahorros para la organización. Lo que se intenta lograr con la implementación de un sistema de información es evolucionar desde el "Caos" que son las hojas de cálculos, a un ambiente "Informado y con aplicaciones consolidadas" para luego evolucionar en un proceso de "Optimización" y finalmente llegar a un modelo de "Data Center planificado estratégicamente". (Pacio, 2013)

Jorge Jiménez, DCIM & Data Center Life Cycle Business Development de Schneider Electric, menciona que actualmente en las empresas los usuarios han tomado conciencia de que la tecnología juega un papel importante en el desarrollo de las actividades diarias, más aún cuando estas aplicaciones cubren con sus necesidades y objetivos. (DataCenterMarket, 2015)

Justificación

Realizar un análisis de procesos tecnológicos de gestión, permite conocer cómo funciona un determinado proceso, quienes son los responsables, los objetivos que se deben cumplir y el alcance de este. Además, representar el proceso mediante un diagrama BPMN (Business Process Management Notation), ayuda a que se pueda visualizar el flujo de trabajo de un determinado proceso, lo que conlleva a que la comprensión sea fluida y de una manera rápida, además de reducir el tiempo de capacitación. Esto permite que la automatización del proceso tenga una menor duración y se respete el flujo de actividades que está documentado, para cumplir con los objetivos que se han planteado los responsables del Data Center y laboratorios del ICC con respecto a un determinado proceso. (White & Miers, 2009, págs. 9-12)

La conveniencia de desarrollar un sistema para la gestión de inventario de activos es que se puede realizar el procesamiento y almacenamiento de la información de los activos de TI de una manera efectiva y ágil, para así lograr un mejor control de los recursos que se encuentran registrados. (Aranda Software, 2018)

Con la implementación de un sistema para la gestión de inventario de activos, se tendrá una constancia de los equipos de hardware y software con los que cuenta el Data Center y laboratorios del ICC, para conocer la cantidad de equipos y así poder realizar diferentes operaciones como actualización, ingreso, reportería, entre otros. Además, es una inversión que ayuda a disminuir los errores humanos, ahorrar tiempo, agilizar los procesos y planificar adecuadamente los recursos para dar una mejor atención a los usuarios. (Gustavo, 2017)

Para poder agilizar la consulta y actualización de la información de los activos de TI, es conveniente implementar un sistema que genere códigos QR para cada activo. Este sistema de códigos QR ayudará a que el tiempo de actualización del inventario

se reduzca, permitirá conocer las características de un activo de TI fácilmente y que los responsables del Data Center se movilicen de un laboratorio a otro, sin necesidad de un lector de código de barras o equipo especial. (Can, 2015)

Implementar un sistema de ticketing para la gestión de soporte técnico, proporcionará a los responsables del Data Center y laboratorios del ICC un registro de las incidencias de los activos, con la finalidad de que sean atendidos con la mayor brevedad posible, previniendo así, la interrupción de las actividades diarias. Además, permitirá al usuario la visualización del estado en el cual se encuentra su incidencia reportada. (ServiceTonic, 2018)

Objetivos

Objetivo General

Analizar, diseñar e implementar un sistema de gestión de inventarios de activos de TI para los laboratorios de la carrera de Ingeniería en Ciencias de la Computación de la Universidad Politécnica Salesiana sede Quito campus Sur.

Objetivos Específicos

Analizar las necesidades y procesos tecnológicos de gestión con los que cuenta actualmente el Data Center y laboratorios del ICC a través de diagramas BPMN, para la elaboración de un manual técnico de procesos. Así como los requerimientos funcionales y no funcionales mediante la especificación de requisitos para diseñar el sistema de inventarios y ticketing para soporte técnico de activos de TI.

Diseñar el sistema de inventarios y ticketing de soporte técnico a través de diagramas UML para visualizar, describir y documentar los principales aspectos que comprende el desarrollo del sistema.

Desarrollar el sistema de inventarios y ticketing para soporte técnico, utilizando la metodología de desarrollo XP (eXtreme Programming), con la finalidad de mejorar la comunicación entre el cliente y el equipo de trabajo.

Implementar el sistema desarrollado en un entorno web, haciendo uso de los recursos del servidor del Data Center, con la finalidad de mantener el sistema operativo durante la jornada de trabajo. Además de una aplicación en Android, que permita realizar la consulta o actualización de los activos de TI a través de códigos QR y el registro de una incidencia para soporte técnico.

Evaluar al sistema de forma técnica, por medio de pruebas de funcionalidad que permitan el análisis e interpretación de los resultados para tomar acciones de mejoramiento del sistema. Además de una valoración económica y legal.

Análisis de la metodología

Para la elección de la metodología de desarrollo se ha planteado tres opciones que son: XP, SCRUM y RUP; de las cuales se ha evaluado cada una de las características para seleccionar la metodología que se ajuste a las necesidades que implica este proyecto.

Tabla 1. Características de metodologías ágiles.

Criterio	XP	SCRUM	RUP
Desarrollo de software iterativo	X	X	X
Verificación continua de calidad	X	X	X
Enfocado en el equipo	X	X	X
Programación en parejas	X		
Flexibilidad en el desarrollo	X		
Requerimientos cambiantes	X	X	X
Definición del proceso de la organización	X		X
Orientada para pequeños o medianos grupos	X		
Interacción con el usuario final	X		
Centrado en las personas	X	X	X
Centrado en la creación del producto	X		
Pruebas unitarias continuas	X		
Cumplimiento de requisitos	X	X	X
Gestión de riesgos	X	X	X
Plan de trabajo cambiante	X		
Indicadores cambiantes	X		
Aumento de la productividad	X	X	X
Alto intercambio de conocimientos	X		

Nota: Características de metodologías ágiles.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En base a varios factores acorde a la naturaleza del proyecto como son: el tiempo de duración, el número de integrantes o desarrolladores, los requerimientos cambiantes, la organización, la calidad y funcionalidad del producto, entre otros, se ha establecido como la metodología más apropiada para este proyecto a XP, ya que cuenta con los criterios o características que ayuden a cumplir las necesidades que se requieren para el presente proyecto.

CAPÍTULO 1

1.1. Estado del arte

1.1.1. Infraestructura de TI

La carrera de Ingeniería en Ciencias de la Computación de la Universidad Politécnica Salesiana Campus Sur ha visto la necesidad de adquirir nueva infraestructura de TI[1] con la finalidad de brindar formación académica innovadora y de calidad, desarrollo de proyectos, investigación y trabajos de titulación para toda la comunidad universitaria. A continuación, se detalla de una manera general la infraestructura de TI:

1.1.1.1. Infraestructura de servidores y almacenamiento

Tabla 2. Infraestructura de procesamiento y almacenamiento.

Equipo	Marca	Modelo	Capacidad	Cantidad
Chasis modular	HPE	Apollo 6000	10 ranuras	1
Servidores de una ranura	HPE	ProLiant XL230a Gen 9	256 GB RAM 1.2 TB Disco	3
Servidor de doble ranura	HPE	ProLiant XL250a	512 GB RAM 1.2 TB Disco	1
Storage	HPE	3PAR 8200	20 TB Disco	1
Cloud Suite 7	VMWARE	Suite 7	-	1

Nota: Detalles de la infraestructura de procesamiento y almacenamiento.
Fuente: (López, 2018)

1.1.1.2. Infraestructura de redes y comunicaciones

Tabla 3. Infraestructura de redes y comunicaciones

Equipo	Marca	Modelo	Cantidad
Switch SAN	HPE	SN3000B	2
Switch Lan	CISCO	Catalyst 9300	3
Switch Lan	CISCO	Catalyst 2960-X	1
Acces Point	CISCO	2XGBE A DOMAIN	3

Nota: Detalles de la infraestructura de redes y comunicaciones.
Fuente: (López, 2018)

1.1.1.3. Infraestructura eléctrica y de seguridad

Tabla 4. Infraestructura eléctrica y de seguridad.

Equipo	Marca	Modelo	Cantidad

[1] El término Infraestructura de TI hace referencia al conjunto de instalaciones, redes, hardware, software, etc., utilizados para brindar servicios de TI a la organización. (Maldonado, 2018)

UPS	APC	Symmetra LX	2
Tablero de Control	SHP	10-063 Series	1
Sistema de Incendios	ECARO	Ecaro 25	1

Nota: Detalles de la infraestructura eléctrica y de seguridad.
Fuente: (López, 2018)

1.1.1.4. Hardware

Tabla 5. Hardware adquirido.

Equipo	Marca	Modelo	Cantidad
Computadoras	DELL	Optiplex 7050	30
iPad Mini	APPLE	Space Grey	31
Tablets	SAMSUNG	GT-P5100	13
iMac	APPLE	Turbo Boost	31

Nota: Detalles del hardware adquirido.
Fuente: (López, 2018)

Además de la infraestructura mencionada en las tablas anteriores, los laboratorios de Networking ya cuentan con equipos de computación y comunicación que se utilizan para las prácticas estudiantiles.

1.1.2. Metodología de desarrollo

1.1.2.1. Introducción

Según un artículo publicado por la Universidad Católica Los Ángeles, "En un proyecto de desarrollo de software la metodología ayuda a definir: Quién debe hacer Qué, Cuándo y Cómo debe hacerlo." (Universidad Católica Los Ángeles Chimbote, 2017, pág. 6), llegando a ser este uno de los aspectos fundamentales a tratar, debido a que la misma marca las pautas correspondientes con las que se regirán los aspectos más importantes durante todo el proceso de desarrollo.

Durante mucho tiempo en el ámbito del desarrollo del software, ha prevalecido la metodología tradicional, aunque esta metodología parecía ser la indicada para cualquier tipo de proyectos, la misma no se ajusta a la flexibilidad que exigen los proyectos de hoy en día, cuya principal característica es el constante cambio en los requerimientos y la corta duración que deben tener en base a la magnitud del proyecto.

1.1.2.2. Metodologías ágiles

Las metodologías agiles permiten ajustar los requerimientos de acuerdo con las necesidades del cliente. Esto supone una gran posibilidad para el cliente a participar más y de mejor manera en el desarrollo de proyecto, ajustando y puliendo cada característica hasta entregar un producto alineado a sus objetivos.

1.1.2.3. Extreme Programming (XP)

1.1.2.3.1. Definición

La metodología XP básicamente facilita al equipo de trabajo en el cumplimiento de hitos en un entorno de constante cambio. XP está sustentada en ciertos principios que facilitarán la fluidez del proceso de desarrollo, los mismos se resumen en: la flexibilidad de requerimientos, la adaptabilidad en lugar de la predicción, la comunicación y las incrementaciones iterativas.

Según Letelier & Penadés, "XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios." (Letelier & Penadés, 2012, pág. 8), presentando además otras ventajas como: pruebas constantes del sistema, programación en parejas, integración continua y la definición de estándares de programación. (Joskowicz, 2008, págs. 12-15)

1.1.2.3.2. Artefactos

1.1.2.3.2.1. Historias de usuario

Las historias de usuario son artefactos utilizados para definir aquellas características que el cliente necesita para su software, en el mismo se detallan los requerimientos funcionales o no funcionales que en un inicio se puedan prever, para luego ir ajustando durante las iteraciones que se puedan presentar. (Joskowicz, 2008, pág. 10)

1.1.2.3.2.2. Plan de entregas

El plan o cronograma de entregas determina cuáles serán las historias de usuario que agrupadas forman un entregable que será presentado al cliente, además define el orden en el que deben ser presentados. (Joskowicz, 2008, págs. 10-11)

1.1.2.3.2.3. Plan de iteraciones

Previo al desarrollo de cada historia de usuario, se realiza una reunión entre el equipo de trabajo en donde se planificará el desarrollo de la iteración. (Joskowicz, 2008, pág. 11)

1.1.2.3.2.4. Reuniones diarias de seguimiento

Las reuniones diarias en el equipo de trabajo son realizadas con la finalidad de tratar problemas y soluciones que pueda presentarse en el transcurso del proyecto. (Joskowicz, 2008, pág. 11)

1.1.2.3.3. Roles

Algunos de los roles propuestos por Kent Beck, considerado el padre de XP, son los siguientes: programador, cliente, tester, encargado de seguimiento, consultor y gestor, el cual es considerado como el nexo entre el cliente y el equipo de trabajo, y es el encargado de las tareas de coordinación y logística en la realización del proyecto. (Letelier & Penadés, 2012, págs. 10-11)

1.1.2.3.4. Fases

Entre las fases comunes que utiliza la metodología XP se encuentran, Fase de exploración, en donde se define el alcance del proyecto y se recolecta los requerimientos iniciales; Fase de planificación, en donde se define el cronograma para las entregas sobre las funcionalidades de las historias de usuario; Fase de iteraciones, en donde se desarrolla cada una de las historias de usuario y se realizan pequeñas entregas del producto final al usuario y Fase de puesta en producción, la

cual consiste en la entrega de todo el producto con sus respectivos módulos funcionando correctamente. (Joskowicz, 2008, págs. 8-9)

1.1.3. Sistema de inventarios

1.1.3.1. Definición de activos de TI

Los activos de TI son todos aquellos recursos, elementos o equipos tecnológicos con los que cuenta una organización, y son considerados como instrumentos fundamentales para el cumplimiento de los objetivos propuestos. (Mendoza, 2018)

1.1.3.2. Definición de inventarios de activos de TI

El inventario permite llevar a cabo el control detallado de cada uno de los bienes que se encuentren registrados, además de mantener actualizada la información, facilitando así la ejecución de procesos como adquisición, mantenimiento, soporte técnico, préstamo de equipos, toma de decisiones, entre otros.(Martín, 2017)

1.1.3.3. Definición de un sistema de gestión de inventarios de activos de TI

Un sistema de gestión de inventarios es una herramienta automatizada que permite llevar a cabo el proceso de recopilación y seguimiento de los activos de TI, proporcionando funcionalidades que se adapten a las necesidades propias de una determinada organización. Además, permite a los responsables de los activos contar con un único sitio que proporcione toda la información del inventario al instante, simplificando así la labor de administración de toda la infraestructura disponible. (Martín, 2017)

1.1.3.4. Clasificación de activos en un inventario

Para facilitar el manejo y mantenimiento del inventario, es conveniente clasificar los activos por categorías, según su naturaleza:

- **Aplicaciones:** El software que se utiliza para la gestión del proceso.
- **Hardware:** Equipos físicos que posea la organización.

- **Red:** Dispositivos de conectividad de redes.
- **Tecnología:** Otros equipos como pueden ser: servidores, equipos de usuario, teléfonos, impresoras, cableado, etc.
- **Personal:** Personal que tengan acceso de una manera u otra a la organización.
- **Instalaciones:** Lugares en los que se alojan los activos. (INCIBE, 2016)

1.1.3.5. Información de los activos de un inventario

Un inventario de activos tiene que proveer información suficiente para futuros proyectos o incidentes, así como proporcionar una visión precisa de sus valores, de forma que sea posible establecer criterios y realizar un análisis de riesgos. Para ello se debe recoger la información más relevante de cada activo, como podría ser:

- **Características:** Puede incluir modelo, marca, nombre descriptivo, etc.
- **Descripción:** Debe contener información sobre el uso del activo.
- **Identificador:** Código único de identificación para el activo. Debe seguir un patrón elegido por la organización.
- **Tipo:** Recoge el grupo al que pertenece el activo.
- **Responsable:** Persona encargada de que el activo se encuentre operativo, así como de gestionar los accesos al mismo.
- **Ubicación:** Lugar donde se encuentra físicamente el activo. (INCIBE, 2016)

1.1.4. Sistema de ticketing para soporte técnico

1.1.4.1. Definición de incidencia

"Una incidencia es toda interrupción o reducción de la calidad no planificada del servicio. Pueden ser fallos o consultas reportadas por los usuarios, el equipo del servicio o por alguna herramienta de monitorización de eventos." (ServiceTonic, 2018)

Básicamente una incidencia en el contexto de las tecnologías de la información es cualquier anomalía o evento inesperado que ocurre dentro de los dominios o recursos tecnológicos de una organización. Estos eventos deben ser solucionados con la mayor brevedad posible con la finalidad que no interfieran con el flujo normal de los procesos que allí operan.

1.1.4.2. Definición de sistema de ticketing para soporte técnico

Un sistema de ticketing para el soporte técnico de incidencias permite llevar el registro de las incidencias, consultas y sugerencias de los usuarios, con el fin de que se brinde la respectiva atención por parte del personal técnico responsable. Además, permite dar un seguimiento, al estado en el que se encuentra cada incidencia. (R2Docuo, 2018)

Los estados que se emplean con regularidad dependiendo de las reglas del negocio son los siguientes: Abierto, en el cual se ha reportado una nueva petición de soporte técnico y se asigna a un responsable para dar solución al mismo; En proceso, en el cual el responsable asignado se encuentra trabajando en la solución del requerimiento; En espera, en el cual se espera la solución por parte de una tercera persona por ejemplo un proveedor o soporte técnico especializado y Resuelto, en el cual se dio una solución a la petición de soporte técnico. (Young, 2017)

Básicamente, el objetivo principal de un sistema de ticketing es que las operaciones vuelvan a su curso normal tan pronto como sea posible, después de un incidente o irregularidad en alguno de los equipos de la organización. Al corregir la incidencia, se estará cumpliendo con los requerimientos de disponibilidad de los servicios de TI, aumentando la eficiencia y productividad en las operaciones de la organización, manteniendo así la satisfacción del usuario. (Maldonado, 2018)

1.1.5. Bases de datos relacionales

Las bases de datos relacionales son un conjunto de herramientas que permiten el almacenamiento, acceso y administración de los datos. Se guían mediante el modelo relacional creado por Codd, en el cual se representa a los datos como un conjunto de tablas y relaciones. Las tablas poseen un nombre único y contienen columnas que representan las características de las mismas, además de filas o tuplas que representan los atributos de las entidades. Para identificar las tuplas de una tabla se hace uso de claves únicas. (Gómez, 2014, págs. 15-16) .

Entre las principales características que presentan las bases de datos relacionales se tiene: Integridad de datos mediante un conjunto de propiedades denominadas ACID (Atomicidad-Consistencia-Aislamiento-Durabilidad) y la capacidad de manejar transacciones mediante la característica denominada "Todo o Nada". (Silberschatz, Korth, & Sudarshan, 2002, págs. 367-368)

1.1.6. Aplicaciones móviles

Es un software que se desarrolla para su ejecución en dispositivos móviles como tablets, smartphones u otros dispositivos móviles, con el fin de apoyar en diferentes actividades empresariales de comercio o en el entretenimiento para los usuarios. Las aplicaciones pueden descargarse e instalarse desde las tiendas digitales o directamente desde un IDE de desarrollo. Además, se ejecutan en diversos sistemas operativos como Android, IOS, Windows Phone, etc. (Servisoftcorp, 2019)

Existen aplicaciones 2 tipos de aplicaciones: las nativas que son elaboradas específicamente para ejecutarse en un dispositivo móvil con la finalidad aprovechar las características del hardware y del sistema operativo y las web apps que haciendo uso de HTML, CSS Y JavaScript permiten su ejecución en navegadores que proporcionan los dispositivos móviles. (IBM, 2018)

1.1.7. MVC

El patrón MVC básicamente divide a una aplicación en tres capas que son: El modelo, que consiste en un conjunto de clases que representa la lógica del negocio; La vista, encargada de representar los datos contenidos en el modelo mediante una interfaz de usuario y finalmente el controlador, encargado de procesar las instrucciones que solicite el usuario, el mismo interactúa tanto con la vista como con el modelo. (Junta de Andalucía, 2013)

1.1.8. Código QR

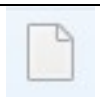
Un código QR básicamente es un código de barras bidimensional que contiene datos codificados que por lo general son textos o enlaces URL. Un código QR permite proporcionar información a cualquier objeto físico, lo cual facilita el acceso a los datos desde cualquier dispositivo móvil hacia a algún servicio web que esta pueda disponer. (Unitag, 2018)

1.1.9. Notación BPMN

1.1.9.1. Componentes básicos

Tabla 6. Componentes básicos de la notación BPMN.

Componente	Objeto	Descripción
Evento de inicio		Marca el inicio de un proceso.
Evento de fin		Marca la finalización de un flujo de proceso.
Evento de finalización de mensaje		Marca la finalización de un flujo de proceso mediante un mensaje.
Evento intermedio simple		Indica que una actividad ocurre entre el inicio y final de un proceso.
Componente de decisión		Es utilizada para determinar un camino a seguir dependiendo de una condición o pregunta.
Tarea		Sirve para describir una tarea específica del proceso.
Anotación		Permite proporcionar información adicional sobre una tarea determinada.

Objetos de datos		Permite identificar que un documento, mensaje o datos están siendo enviados o recibidos de una tarea a otra.
Pool		Contiene el flujo de un determinado proceso.
Lane		Es utilizada para definir roles, cargos, posicione o, departamentos internos al proceso.
Línea de flujo		Define el orden en que se ejecutarán las tareas dentro del proceso.
Flujo de mensaje		Es utilizada para el envío y recepción de mensajes de un pool a otro.

Nota: Componentes de un diagrama BPMN.
Fuente: (Bizagi, 2018)
Elaborador por: Andres Chisaguano, Joel Ludeña.

1.1.9.2. Tipos de diagramas

Los tipos de diagramas de procesos que soporta la notación BPMN son los siguientes:

1.1.9.2.1. Orquestación

La orquestación consiste en indicar el flujo de un proceso determinado de una única entidad. Sin embargo, un diagrama BPMN puede contener más de una orquestación, las cuales estarán contenidas dentro de un pool. (Bizagi, 2018). La Figura No. 1 muestra un ejemplo de orquestación.

Ejemplo de orquestación de procesos

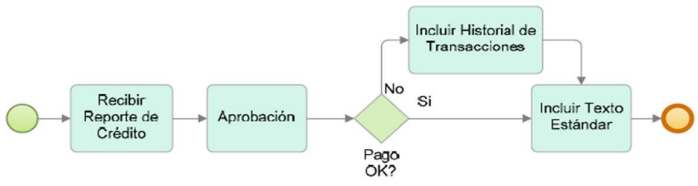


Figura 1. Ejemplo de orquestación en BPMN.
Fuente: (White & Miers, 2009, págs. 29-31)

1.1.9.2.2. Coreografía

La coreografía describe la interacción que existe entre dos o más participantes de un proceso. Los participantes se definen mediante un "Lane" y pueden ser: roles, cargos, posiciones, departamentos, etc. (Bizagi, 2018). Un ejemplo de coreografía es el diagrama definido en la Figura No. 2 que se presenta a continuación.

Ejemplo de coreografía de procesos

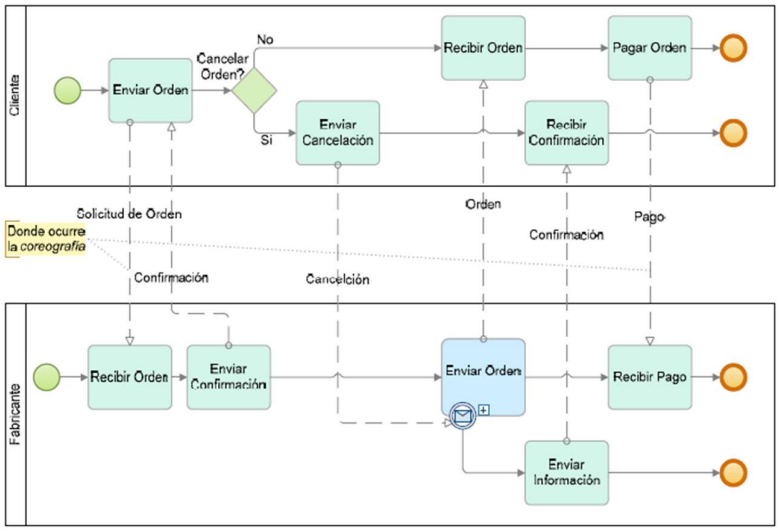


Figura 2. Ejemplo de coreografía en BPMN.
Fuente: (White & Miers, 2009, págs. 29-31)

1.1.9.2.3. Colaboración

La colaboración es un diagrama BPMN que muestra la interacción entre dos o más participantes que pueden estar contenidos en uno o varios pools referentes a un proceso determinado. En un diagrama de colaboración no es obligatorio que un pool describa el proceso que realiza, lo que interesa es visualizar el flujo de mensajes. (Bizagi, 2018). Un ejemplo de colaboración es el diagrama definido en la Figura No. 3 que se presenta a continuación.

Ejemplo de colaboración de procesos

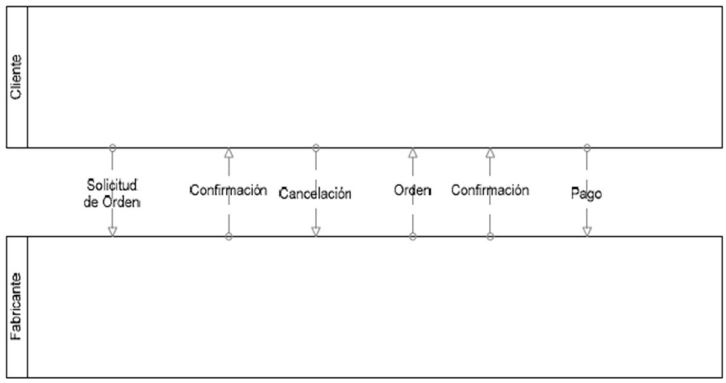


Figura 3. Ejemplo de colaboración en BPMN.
Fuente: (White & Miers, 2009, págs. 29-31)

1.1.10. Herramientas de desarrollo

1.1.10.1.Lenguaje de programación C#

C# es un lenguaje de programación que permite a los desarrolladores crear aplicaciones guiándose en el paradigma de orientación a objetos, las cuales se ejecutan en el entorno .NET Framework. Este lenguaje puede ser utilizado para crear aplicaciones de escritorio en Windows, servicios web, aplicaciones cliente-servidor, aplicaciones de base de datos, entre otras. Cualquier desarrollador que haya trabajado con C, C++ o Java, reconocerá la sintaxis de C# y se adaptará fácilmente a su uso. (Microsoft, 2015)

1.1.10.2.ASP .NET

ASP.NET es una plataforma de trabajo utilizada para crear aplicaciones Web mediante HTML, CSS y JavaScript. Está compilado en .NET Framework, por lo que todas sus características y funciones se encuentran disponibles. (Microsoft, 2018)

1.1.10.3.ASP .NET MVC

ASP.NET MVC es un framework basado en el patrón modelo, vista y controlador, utilizado para desarrollar sitios web dinámicos y potentes. Permite separar eficazmente las capas de datos, procesamiento e interfaz. Contiene una variedad de

características y funciones que permiten el desarrollo ágil de una aplicación web. (Microsoft, 2018)

1.1.10.4.Servicios REST

Un servicio REST puede definirse como cualquier interfaz entre sistemas que usan el protocolo HTTP para obtener datos o generar operaciones, estos se realizan sobre formatos como XML y JSON. (BBVAOPEN4U, 2016)

1.1.10.5.ASP .NET CORE

ASP.NET Core es un marco multiplataforma, de alto rendimiento y de código abierto utilizado para desarrollar aplicaciones basadas en la nube y conectadas a Internet. (Roth, Anderson, & Luttin, 2018)

1.1.10.6.Visual Studio Community 2017

El entorno de desarrollo integrado de Visual Studio es un panel creativo que permite la visualización, edición, depuración, compilación y publicación de aplicaciones. (Microsoft, 2018)

Características:

- Permite seleccionar e instalar cargas de trabajo, necesarias para el lenguaje de programación o la plataforma requerida.
- Permite crear aplicaciones web mediante ASP.NET, Node.js, Python, JavaScript, etc.
- Permite depurar el código que se ejecuta en el proyecto local, en un dispositivo remoto o en un emulador de dispositivo.
- Posee mejor capacidad de respuesta y utiliza menos memoria.
- Se puede distribuir el software en cualquier plataforma de destino.

1.1.10.7.JWT Token

"El token web de JSON es un estándar abierto que define una forma compacta y autónoma para transmitir de forma segura información entre las partes como un objeto JSON." (JWT, 2018)

Un JWT básicamente es una cadena de texto que contiene 3 partes codificadas en Base64. Las partes del JWT son: Header, el cual determina el algoritmo y el tipo del token; Payload, son los datos con los que fue configurado el token, como el usuario, fecha de generación y caducidad, entre otros; finalmente la firma o signature, el cual permite comprobar que la validez del token. (Macias, 2018)

1.1.10.8.Bizagi Modeler

Bizagi Modeler es una aplicación de escritorio gratuita que se guía según el estándar BPMN para el diseño y representación de procesos empresariales. Posee una interfaz gráfica intuitiva y cuenta con el paradigma de arrastrar y soltar lo que hace que la experiencia de modelar un proceso sea rápida y ágil, convirtiéndola en un software útil para dar a conocer los procesos a todos los departamentos de la empresa.(Bizagi, 2018)

1.1.10.9.Android Studio

Es un entorno de desarrollo de aplicaciones Android basado en IntelliJ IDEA, con licencia de software libre apache 2.0 que incluye varias herramientas como editores de texto, compiladores, monitores de recursos, pruebas unitarias, etc. que ayudan al desarrollador a elaborar software de calidad. Además, permite crear diseños de forma rápida a través de la interfaz gráfica y de texto del sistema "Gradle" proporcionando una muestra visual de la estructura de los componentes de la pantalla para realizar una configuración adecuada dependiendo del dispositivo.

Una de las ventajas más notables es que Android Studio permite realizar varias versiones de una misma aplicación, esto es útil cuando se realiza una versión gratuita, una versión de paga o una versión con diferentes configuraciones para cada dispositivo. (Developers Android, 2018)

1.1.10.10. PostgreSQL

Es un sistema gestor de base de datos objeto-relacional de código abierto que soporta herencia, diferentes tipos de datos, funciones, restricciones, disparadores, triggers, reglas, transaccionalidad, entre otras funciones. Además, soporta integridad referencial para garantizar la validez de los datos, soporta hasta 32 TB de información, provee un entorno de alta disponibilidad para los clientes, ofrece estabilidad, es potente, es robusto, implementa estándares y su administración es sencilla. Añadiendo a las características mencionadas.(postgresql, 2017)

1.1.10.11. UML

El Lenguaje de Modelado Unificado es un estándar con especificación de diseño orientado a objetos que permite describir las diferentes etapas de un proyecto de software mediante diversos diagramas, con la finalidad de tener una visión dinámica del sistema. (Canchala, 2016)

1.2. Marco institucional

Los laboratorios, equipos de comunicación y el Data Center de la Carrera de Ingeniería en Ciencias de la Computación, se encuentran ubicados en Quito, en la Av. Rumichaca Ñan y Av. Moran Valverde, en la Universidad Politécnica Salesiana Campus Sur, bloque D.

La carrera de Ingeniería en Ciencias de la Computación es una carrera nueva que está enfocada en la enseñanza de infraestructura y gestión de sistemas, para lo cual ha

puesto sus esfuerzos en la implementación, adquisición y remodelación de los siguientes componentes:

- Tres laboratorios de networking.
- Un laboratorio de Interacción Humano Máquina.
- Un laboratorio de Servidores.
- Un laboratorio de Sistemas Embebidos.
- Servidores y sistemas de almacenamiento.
- Equipos de comunicación y networking.
- Equipos eléctricos.
- Estaciones de Trabajo.

El Data Center fue inaugurado el mayo del 2018 y su implementación fue planificada durante el mes de noviembre del 2016. El proyecto en sus inicios fue pensado y diseñado por los estudiantes de la carrera de Ingeniería en Sistemas, mención Telemática en la materia Administración y Gestión de Redes, como proyecto de fin de semestre. El diseño propuesto por los estudiantes tuvo acogida por la carrera de Ingeniería en Ciencias de la Computación y fue un hito que marcó el comienzo de su implementación.

La propuesta del Data Center fue analizada por dos empresas que se dedican a la comercialización de servidores e infraestructura para Data Centers, determinando que la empresa "AKROS" era la opción más rentable y que se ajustaba para la solución propuesta.

Una vez determinada la empresa, se empezó con la construcción del espacio físico para albergar la nueva solución. La construcción del espacio físico duró seis meses aproximadamente, la implementación del sistema eléctrico y de ventilación a cargo de la empresa "PLAGECONS" y tuvo una duración de siete meses

aproximadamente, y la capacitación e implementación de la solución del Data Center tuvo una duración de ciento sesenta horas.

Una vez finalizado el proceso de implementación, empezó el proceso de configuración del entorno virtual para ofrecer espacio de almacenamiento, procesamiento y creación de máquinas virtuales para ofrecer estos servicios a los docentes e investigadores de la universidad Politécnica Salesiana Campus Sur.

Para asegurar el correcto uso de la nueva adquisición y todos los componentes antes mencionados, se designó como administrador al Ing. Jorge López MSc, y a un equipo de dos auxiliares técnicos para apoyar en las labores de administración y soporte.

Entre las tareas y servicios que ofrecen el Data Center y laboratorios del ICC se encuentran las siguientes: soporte técnico, Asignación y administración de los recursos del servidor, configuración de máquinas virtuales, monitoreo del estado del servidor y sus componentes, préstamos de laboratorios, mantenimiento de laboratorios, mantenimiento de equipos de computación y networking, configuración de estaciones de trabajo, préstamo de equipos de networking, control de incidencias e inventarios.

1.3. Misión

Ofrecer servicios de virtualización, préstamo de laboratorios y equipos de comunicación, servicios de procesamiento y almacenamiento, para el desarrollo académico e investigativo de la comunidad universitaria, con el fin de incentivar y fortalecer los conocimientos académicos y humanos adquiridos.

1.4. Visión

Ser un Centro de Datos enfocado en la prestación de servicios tecnológicos de calidad para los estudiantes de la Universidad Politécnica Salesiana, que facilite el

óptimo proceso de aprendizaje y enseñanza, basado en la investigación, innovación e infraestructura de vanguardia para la comunidad universitaria entre ellos estudiantes, docentes, personal administrativo, investigadores y egresados que actúan en función de la mejora continua y excelencia académica.

1.5. Organigrama actual

Organigrama actual de la organización

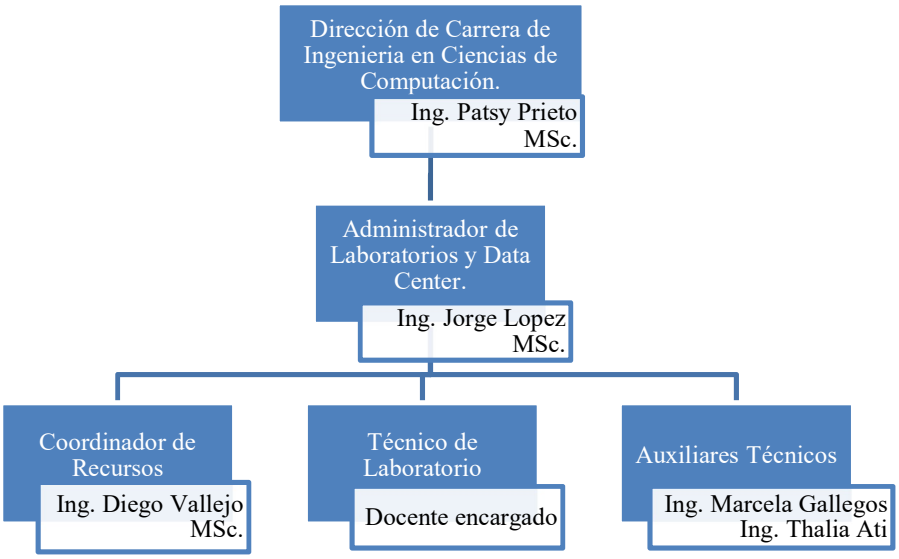
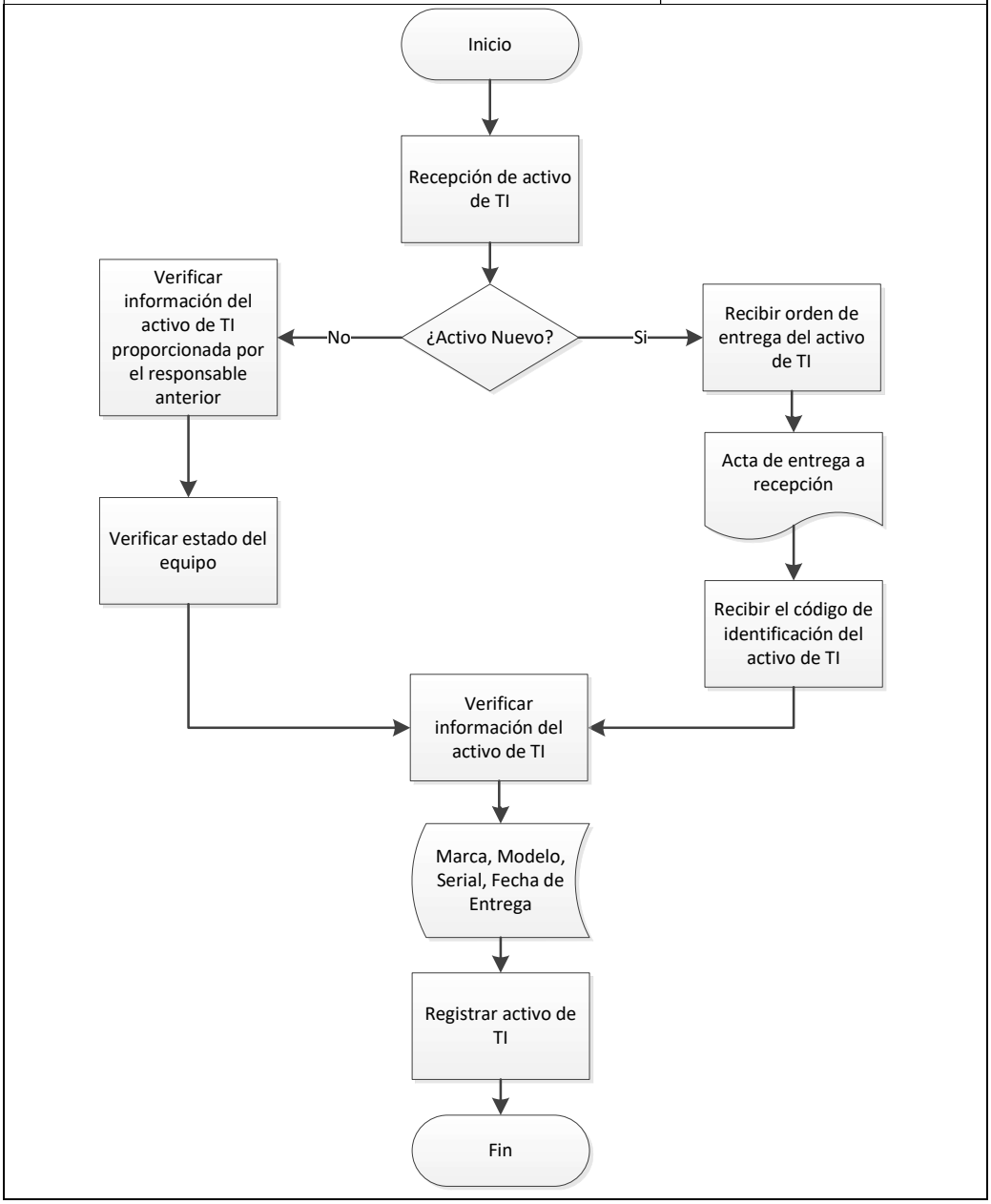


Figura 4. Organigrama actual del Data Center y laboratorios del ICC.
Elaborado por: Andres Chisaguano, Joel Ludeña.

1.6. Flujo de inventario

Tabla 7. Flujo de proceso de inventarios.

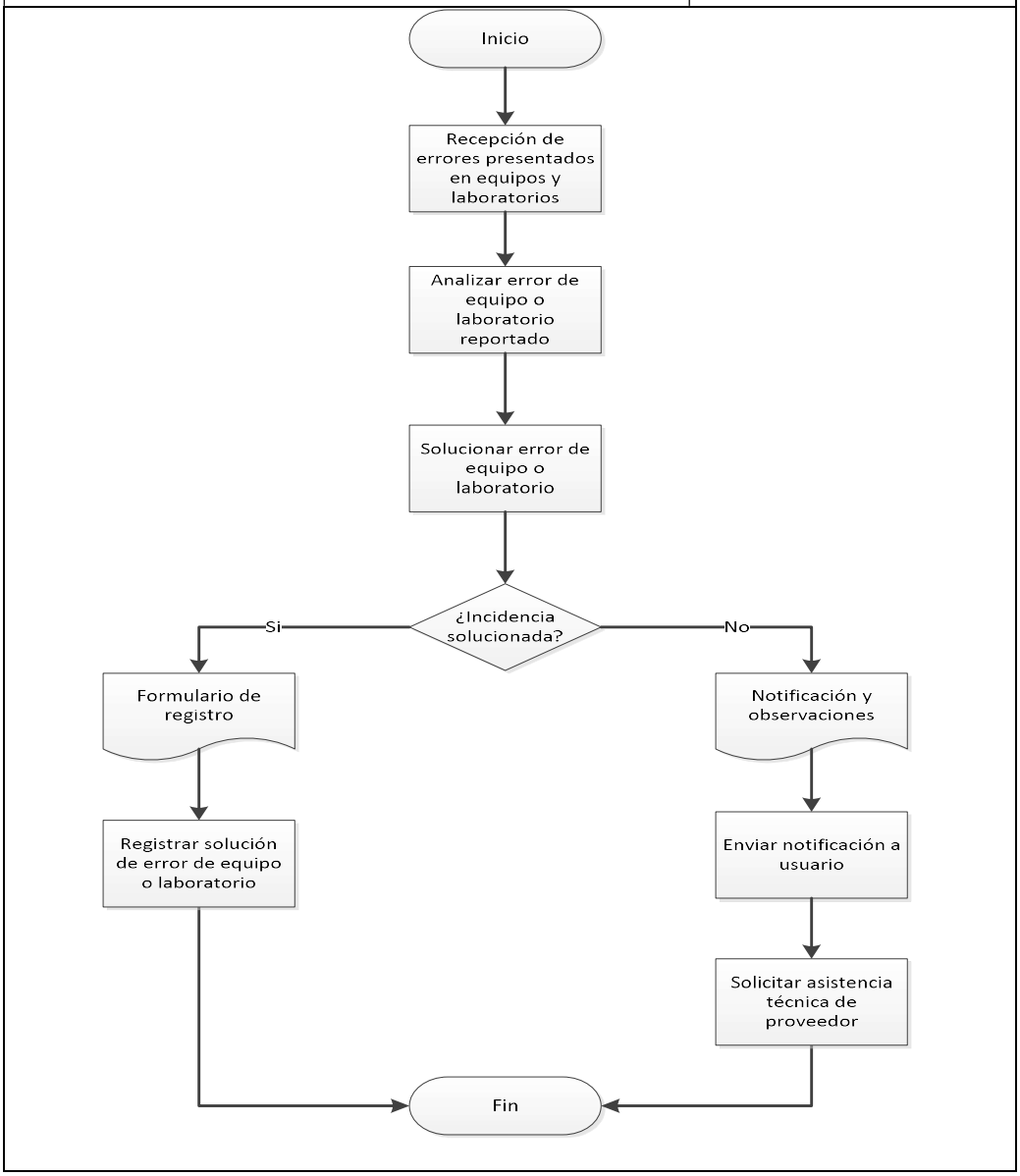
Diagrama de Flujo de Proceso: Control de Inventario de Activos de TI del Data Center y laboratorios del ICC.	**Proceso:** Inventario
Descripción: El proceso de control de inventarios consiste en el registro de los activos fijos de TI del Data Center y laboratorios del ICC, adquiridos y existentes, mediante la verificación respectiva de la documentación al ser nuevo y su estado al ya pertenecer a los laboratorios del ICC.	**Subproceso:** Generación de código de identificación del activo de TI. **Involucrados:** Administración de Campus, Administrador de los laboratorios del ICC, Auxiliares Técnicos.



Nota: Proceso de inventarios.
Elaborador por: Andres Chisaguano, Joel Ludeña.

1.7. Flujo de ticketing para soporte técnico

Tabla 8. Flujo de proceso de ticketing para soporte técnico.

<table>
<tr><td>Diagrama de Flujo de Proceso: Ticketing para soporte técnico de los laboratorios del ICC.</td><td>Proceso: Ticketing para soporte técnico</td></tr>
<tr><td rowspan="2">Descripción: El proceso de ticketing para soporte técnico consiste en el reporte y registro de un error o requerimiento, suscitado en las actividades normales de los usuarios de los laboratorios del ICC para informar a los auxiliares técnicos que se necesita soporte técnico. Los auxiliares técnicos analizan el tipo de soporte y se encargan de dar solución al mismo. Si el error fue atendido se procede a llenar un formulario para mantener la constancia, caso contrario se notifica al usuario las razones por las que no se atendió a su pedido de soporte técnico y se notifica al proveedor para hacer uso de la garantía proporcionada o solicitar servicio técnico de un especialista.</td><td>Subproceso: Reporte de errores en equipos o laboratorios por parte de los usuarios.</td></tr>
<tr><td>Involucrados: Usuarios de los laboratorios del ICC, Auxiliares Técnicos.</td></tr>
<tr><td colspan="2"> 
</td></tr>
</table>

Nota: Proceso de ticketing para soporte técnico.
Elaborador por: Andres Chisaguano, Joel Ludeña.

CAPÍTULO 2

2.1. Descripción de procesos del Data Center y laboratorios del ICC

A continuación, se describen los procesos que actualmente se llevan a cabo en el Data Center y laboratorios del ICC.

2.1.1. Proceso de control de inventarios

2.1.1.1. Objetivo

Mantener una constancia y registro de los activos de TI con los que cuenta el Data Center y los laboratorios del ICC.

2.1.1.2. Alcance

Inicia con la recepción del activo, junto con la documentación pertinente y finaliza con el registro del activo con sus características técnicas. Aplica para los activos fijos pertenecientes a los nuevos laboratorios de la carrera de Ingeniería en Ciencias de la Computación. No aplica para activos fuera del bloque D y embebidos, materiales de oficina o muebles y enseres.

2.1.1.3. Responsables

Tabla 9. Responsables de control de inventarios.

Responsable	Cargo	Actividad
Ing. Jorge López MSc.	Administrador	• Recepción de activos y documentación.
Ing. Marcela Gallegos	Auxiliar Técnico	• Registro de inventario.

Nota: Proceso de control de inventarios.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.1.4. Descripción

El proceso de control de inventarios consiste en el registro de los activos fijos de TI del Data Center y laboratorios del ICC, iniciando con la recepción para la comprobación de su existencia. En caso de ser un activo nuevo, se procede a solicitar la documentación respectiva, caso contrario se recolecta la información del activo, para finalmente registrar la información en el almacén de datos.

2.1.1.5. Diagrama BPMN

Proceso de control de inventarios

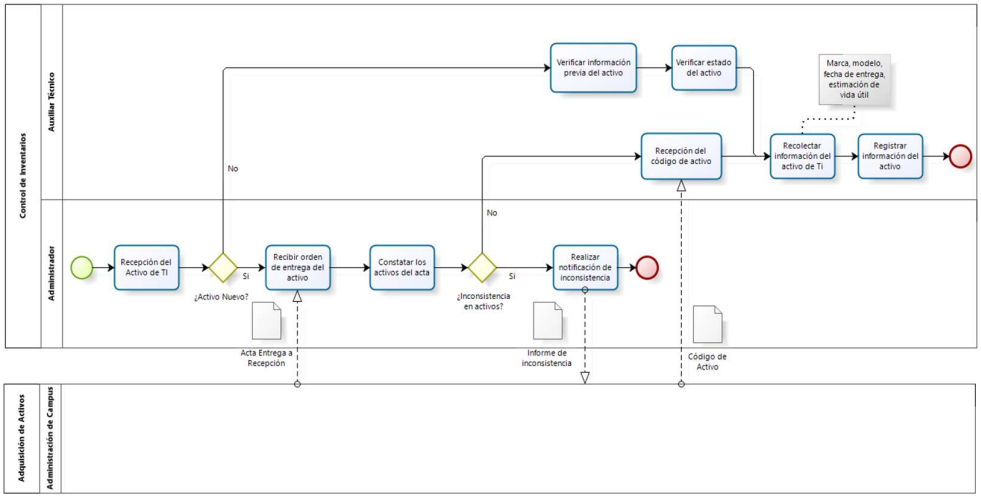


Figura 5. Diagrama BPMN proceso de control de inventarios.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.2. Proceso de ticketing para soporte técnico

2.1.2.1. Objetivo

Proporcionar asistencia técnica a los usuarios que reportan problemas de hardware o software, así como requerimientos técnicos en los laboratorios del ICC, con la finalidad de mantener el correcto funcionamiento de los recursos de cada laboratorio.

2.1.2.2. Alcance

Inicia con la notificación de un problema de software, hardware o requerimiento técnico suscitado en los laboratorios del ICC, y finaliza con la solución de dicho problema. Aplica para infraestructura, software y hardware; El horario de atención de soporte técnico será dentro del horario académico.

2.1.2.3. Responsables

Tabla 10. Responsables de soporte técnico.

Responsable	Cargo	Actividad
Ing. Marcela Gallegos Ing. Thalia Ati	Auxiliares Técnicos	• Recepción de notificación. • Soporte técnico.

Nota: Proceso de soporte técnico.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.2.4. Descripción

Cuando se presenta un problema o inconveniente al hacer uso de los recursos de los laboratorios del ICC por parte de los usuarios, se notifica personalmente o vía correo electrónico para solicitar soporte técnico. Una vez recibida la notificación, el auxiliar técnico a cargo analiza el tipo de soporte y soluciona el problema. Si el error fue atendido, se procede a llenar un formulario para mantener la constancia de que el soporte técnico solicitado fue exitoso, caso contrario se notifica al usuario las razones por las que no se pudo solucionar el error, y se notifica al proveedor para hacer uso de la garantía proporcionada o solicitar servicio técnico de un especialista.

2.1.2.5. Diagrama BPMN

Proceso de ticketing para soporte técnico

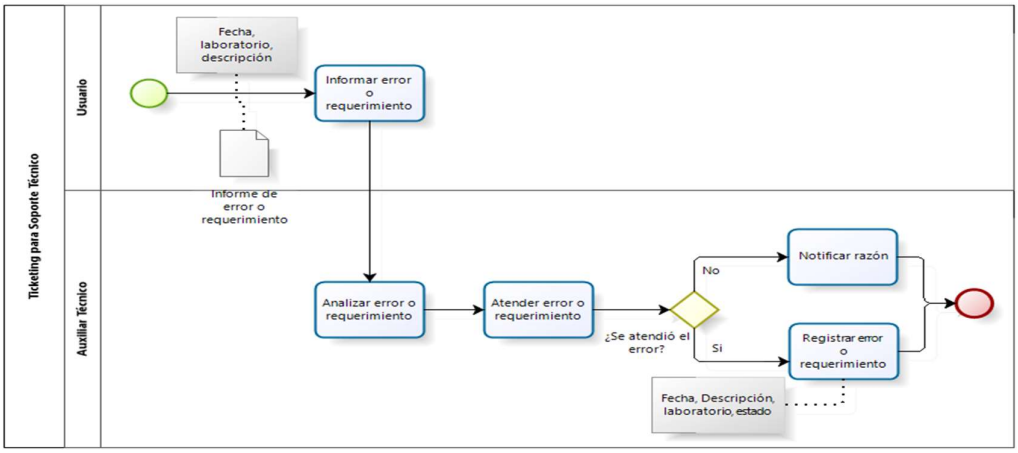


Figura 6. Diagrama BPMN proceso de ticketing para soporte técnico.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.3. Proceso de control de incidencias

2.1.3.1. Objetivo

Analizar, registrar y solucionar las incidencias que se producen eventualmente en los recursos de hardware, software e infraestructura del Data Center del ICC.

2.1.3.2. Alcance

Inicia con la notificación de una incidencia que ha ocurrido en alguno de los recursos del Data Center, y finaliza con la solución y registro de la misma. Aplica para eventos no previstos y en recursos como: infraestructura, hardware, equipos electrónicos, software de máquinas virtuales y servidores, software que ha sido corrompido por actualizaciones. No aplica para software de equipos de escritorio, actualizaciones de Windows, equipos de networking e infraestructura de los laboratorios del ICC. Las notificaciones de incidencias pueden ser previstas tanto por el usuario como por los sistemas adquiridos por el Data Center.

2.1.3.3. Responsables

Tabla 11. Responsables de control de incidencias.

Responsable	Cargo	Actividad
Ing. Jorge López MSc.	Administrador	• Supervisión de incidencias.
Ing. Marcela Gallegos	Auxiliar Técnico	• Registro de incidencias. • Resolución de incidencias.
Ing. Thalia Ati	Auxiliar Técnico	• Resolución de incidencias.

Nota: Proceso de control de incidencias.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.3.4. Descripción

El control de incidencias consiste en la notificación, registro y resolución de incidencias que se producen inesperadamente en los recursos del Data Center. La notificación es recibida por uno de los responsables; la incidencia se analiza y se la soluciona conforme se van notificando. Posteriormente se registra la incidencia en un formulario en donde constan los detalles de este, se notifica al administrador que la incidencia ha sido solucionada y finalmente se almacena en un registro en donde constan todas las incidencias que han sido resueltas.

2.1.3.5. Diagrama BPMN

Proceso de control de incidencias

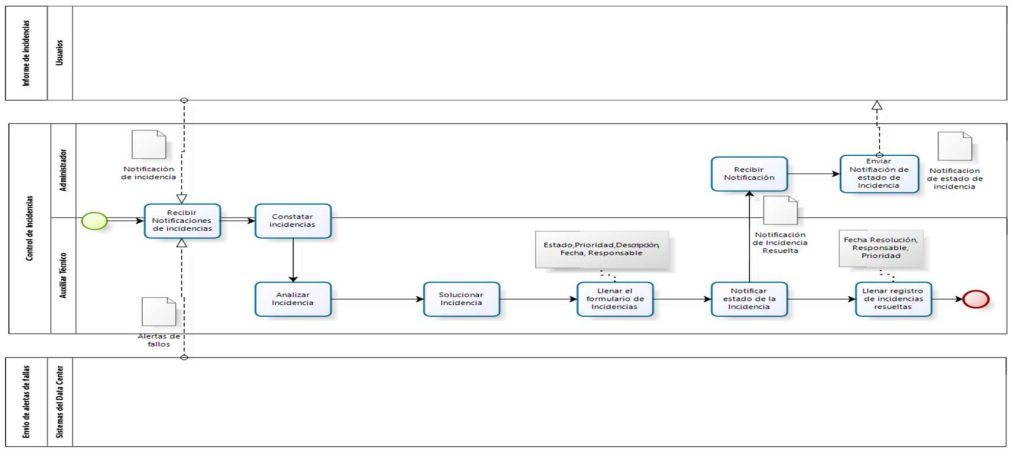


Figura 7. Diagrama BPMN proceso de control de incidencias.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.4. Proceso de préstamo de equipos

2.1.4.1. Objetivo

Proporcionar equipos de networking a los estudiantes y docentes que conforman la Universidad Politécnica Salesiana campus Sur, para fines académicos, proyectos y prácticas dentro de los laboratorios del ICC.

2.1.4.2. Alcance

Inicia con la solicitud de equipos de networking por parte del usuario mediante un formulario para comprobar la disponibilidad de los mismos, y finaliza con la entrega de los equipos para hacer uso de ellos. Solo aplica para equipos de networking y su uso dentro de los laboratorios del ICC. Los equipos se prestarán dependiendo de la disponibilidad y la duración de préstamo dependerá de la práctica que se realice.

2.1.4.3. Responsables

Tabla 12. Responsables de préstamo de equipos.

Responsable	Cargo	Actividad
Ing. Jorge López MSc.	Administrador	• Supervisión de préstamos.
Ing. Marcela Gallegos Ing. Thalia Ati	Auxiliares Técnicos	• Recepción de solicitud. • Entrega y recepción de equipos.

Nota: Proceso de préstamo de equipos.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.4.4. Descripción

El préstamo de equipos consiste en la solicitud de equipos de networking mediante un formulario por parte de los docentes y estudiantes, para hacer uso en prácticas, proyectos y con fines académicos. Una vez realizada la solicitud, el auxiliar técnico verifica la disponibilidad de los equipos solicitados; Si los equipos están disponibles, se hace entrega al usuario y se realiza un registro para tener constancia del préstamo, caso contrario el usuario es notificado de que no hay disponibilidad de préstamo. Una vez que el usuario finaliza el uso de los equipos de networking, se hace entrega al auxiliar técnico encargado para verificar su correcto estado y realizar el registro de la finalización de uso.

2.1.4.5. Diagrama BPMN

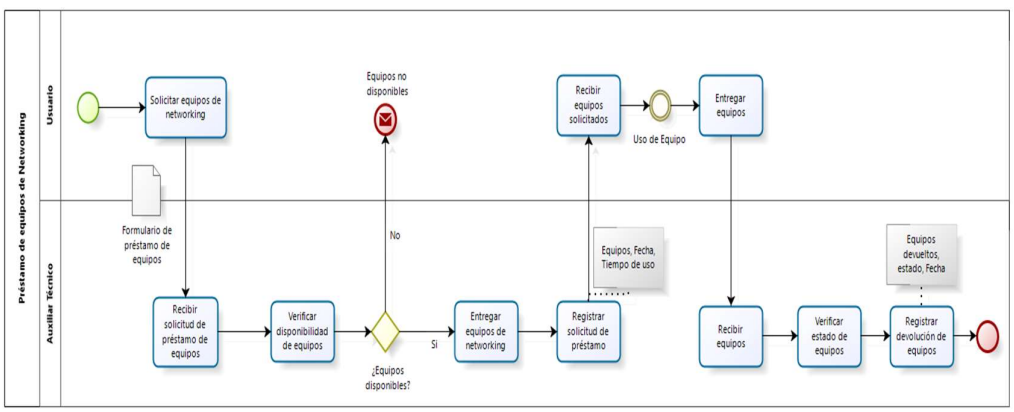


Figura 8. Diagrama BPMN proceso de préstamo de equipos.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.5. Proceso de reserva de laboratorios

2.1.5.1. Objetivo

Proporcionar un laboratorio del ICC a estudiantes y docentes que conforman la Universidad Politécnica Salesiana campus Sur, para fines académicos y prácticas estudiantiles.

2.1.5.2. Alcance

Inicia con la solicitud de un laboratorio por parte del usuario mediante un formulario, para comprobar la disponibilidad del horario, y finaliza con la reserva para su uso. Solo aplica para laboratorios pertenecientes al ICC. Un laboratorio se reservará dependiendo de la disponibilidad conforme al horario establecido por dirección de carrera y la duración de préstamo dependerá de la disponibilidad del horario académico. Aplica para reservas de un solo laboratorio, además no existe un límite de tiempo de uso.

2.1.5.3. Responsables

Tabla 13. Responsables de reserva de laboratorios.

Responsable	Cargo	Actividad
Ing. Jorge López MSc.	Administrador	• Supervisión de reservas.
Ing. Marcela Gallegos Ing. Thalia Ati	Auxiliares Técnicos	• Recepción de solicitud. • Reserva de laboratorios.

Nota: Proceso de reserva de laboratorios.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.5.4. Descripción

La reserva de laboratorios consiste en la solicitud mediante un formulario por parte de los docentes y estudiantes, para hacer uso en prácticas, proyectos y con fines académicos. Una vez realizada la solicitud, el auxiliar técnico verifica la disponibilidad del laboratorio solicitado dependiendo del horario académico establecido; Si el laboratorio está disponible, se hace la reserva y entrega al usuario, posteriormente se realiza un registro para tener constancia del préstamo, caso contrario el usuario es notificado de que no hay disponibilidad de préstamo.

2.1.5.5. Diagrama BPMN

Proceso de reserva de laboratorios

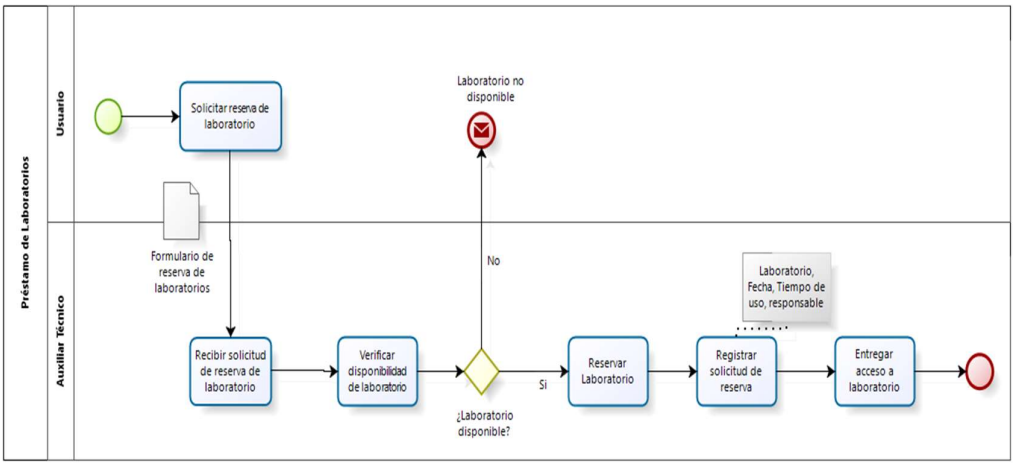


Figura 9. Diagrama BPMN proceso de reserva de laboratorios.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.6. Proceso de asignación de recursos del servidor

2.1.6.1. Objetivo

Proporcionar un espacio virtual de trabajo en el servidor del Data Center a estudiantes, docentes e investigadores de la Universidad Politécnica Salesiana campus Sur sede Quito, para la elaboración de proyectos de investigación, proyectos de titulación y prácticas académicas.

2.1.6.2. Alcance

Inicia con la solicitud del formato de asignación de recursos del servidor al coordinador de recursos, para su análisis y aprobación, y finaliza con la asignación y entrega de credenciales de acceso a la máquina virtual. La asignación de recursos en el servidor para los estudiantes aplica únicamente si se está desarrollando un proyecto de titulación o investigación. La asignación de recursos se realizará, si es que existe disponibilidad de recursos en el servidor y se cumple con los parámetros de memoria, disco y procesamiento establecidos. El tiempo de uso del recurso en el

servidor, tendrá una duración de un periodo académico, con la opción de ampliar su duración previa solicitud del responsable del recurso.

2.1.6.3. Responsables

Tabla 14. Responsables de asignación de recursos del servidor.

Responsable	Cargo	Actividad
Ing. Diego Vallejo MSc.	Coordinador de recursos	• Análisis y aprobación de solicitud de asignación de recursos.
Ing. Marcela Gallegos Ing. Thalia Ati	Auxiliares Técnicos	• Configuración de máquina virtual. • Elaboración de acta de entrega de máquina virtual.

Nota: Proceso de asignación de recursos del servidor.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.6.4. Descripción

La asignación de recursos del servidor consiste en la solicitud de una máquina virtual a través de un formato en donde se detallan las especificaciones que tendrá dicho espacio virtual. El usuario envía la solicitud y el coordinador de recursos analiza si los requerimientos que se solicitan son coherentes y cumplen con los límites de espacio y procesamiento establecidos. Si la solicitud es factible, el coordinador de recursos envía a los auxiliares técnicos el formato con las especificaciones, para que se de paso a la configuración de la máquina virtual, caso contrario se notifica al usuario que debe corregir los requerimientos de su solicitud. Una vez finalizada la configuración, el auxiliar técnico a cargo elabora el acta de entrega en donde constan los accesos y detalles de la máquina virtual al usuario solicitante.

2.1.6.5. Diagrama BPMN

Proceso de asignación de recursos del servidor

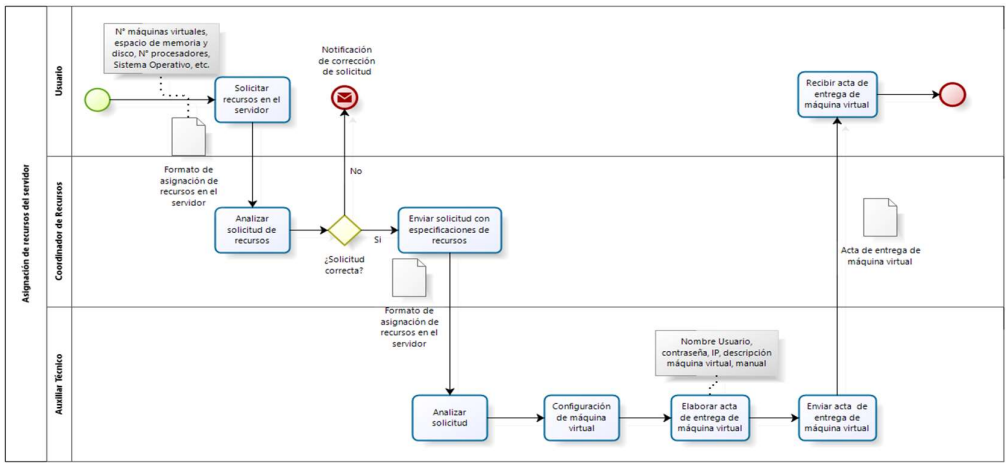


Figura 10. Diagrama BPMN proceso de asignación de recursos del servidor.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.7. Proceso de mantenimiento de hardware y software

2.1.7.1. Objetivo

Proporcionar mantenimiento preventivo y correctivo a los recursos de hardware y software en los laboratorios del ICC, con la finalidad de cubrir los requerimientos de los docentes para un óptimo uso de las instalaciones que se verá reflejado en el proceso de enseñanza y aprendizaje.

2.1.7.2. Alcance

Inicia con el análisis de los requerimientos académicos para cada laboratorio y finaliza con el mantenimiento preventivo y correctivo en cada laboratorio del ICC. Se realiza cada seis meses al finalizar cada periodo académico. Durante este periodo, los laboratorios se encuentran cerrados para realizar el mantenimiento correspondiente por los auxiliares técnicos.

2.1.7.3. Responsables

Tabla 15. Responsables de mantenimiento de hardware y software.

Responsable	Cargo	Actividad
Docentes	Docentes	• Entrega de requerimientos.
Ing. Marcela Gallegos Ing. Thalia Ati	Auxiliares Técnicos	• Recepción de

		requerimientos. • Mantenimiento.

Nota: Proceso de mantenimiento de hardware y software.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.7.4. Descripción

El mantenimiento preventivo y correctivo consiste en la instalación de software, limpieza de equipos y laboratorios, reemplazo de componentes eléctricos y de comunicación. Los auxiliares técnicos se encargan de enviar un correo electrónico a los docentes, solicitando los requerimientos técnicos que tendrán los equipos de cómputo de cada laboratorio. Una vez obtenida la lista de requerimientos, se procede a la instalación del software requerido o al mantenimiento del hardware.

2.1.7.5. Diagrama BPMN

Proceso de mantenimiento de hardware y software

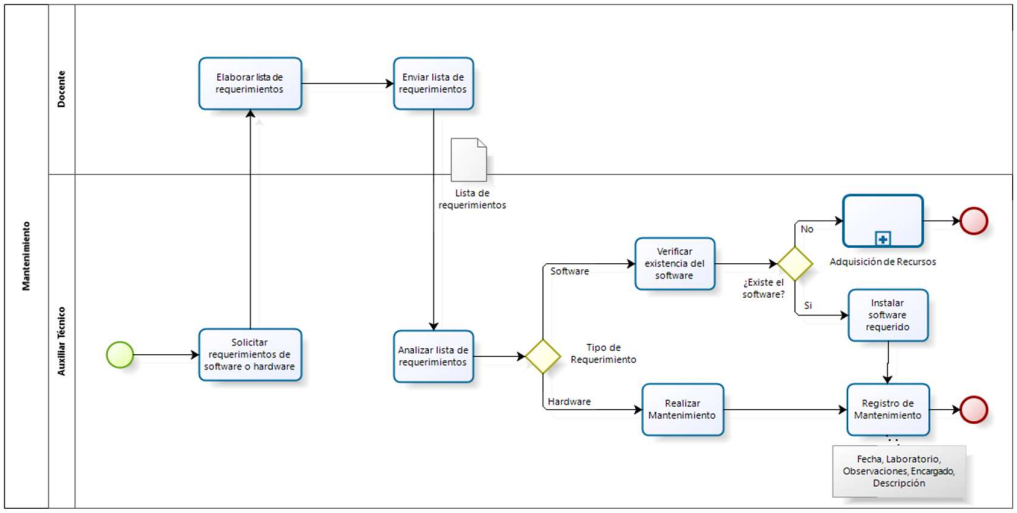


Figura 11. Diagrama BPMN proceso de mantenimiento de hardware y software.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.8. Proceso de adquisición de recursos

2.1.8.1. Objetivo

Analizar las necesidades de hardware y software que se pueden presentar en los laboratorios del ICC, para su posterior adquisición con el fin de aumentar la calidad de enseñanza y aprendizaje, y brindar servicios con tecnología de punta.

2.1.8.2. Alcance

Inicia con el análisis de las condiciones de los laboratorios, para determinar que equipos se deben renovar o adquirir, y finaliza con la aprobación y adquisición por parte de la administración de la universidad. Aplica para la adquisición o renovación de todo tipo de hardware y software sin restricción. La adquisición depende netamente de la administración de la universidad, quien analiza si es factible realizar la adquisición.

2.1.8.3. Responsables

Tabla 16. Responsables de adquisición de recursos.

Responsable	Cargo	Actividad
Ing. Patsy Prieto MSc.	Dirección de Carrera	• Análisis de TDR. • Aprobación de adquisición.
Sra. Paty Quiroz	Administración de Campus	• Análisis de TDR. • Envío de aprobación.
Sr. Christian Guachilema	Administración General de Sede	• Análisis de TDR • Aprobación de adquisición. • Tramite de compra.
Ing. Jorge López MSc.	Administración de Data Center	• Elaboración de TDR. • Envío de TDR
Ing. Marcela Gallegos Ing. Thalia Ati	Auxiliares Técnicos	• Análisis de requerimientos de laboratorios.

Nota: Proceso de adquisición de recursos.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.8.4. Descripción

La adquisición o renovación de recursos consiste en analizar las necesidades que presentan los laboratorios del ICC, tanto de hardware como de software. Comienza con el análisis por parte de un auxiliar técnico, quien determina los recursos faltantes o por renovar. Este análisis es enviado al administrador del Data Center para la

elaboración de un TDR[2], el mismo es enviado a dirección de carrera para su respectivo análisis y aprobación. Si no se presentan errores, dirección de carrera envía un informe con los requerimientos solicitados a administración de campus, caso contrario se notifica al administrador que debe corregir el TDR. Administración de campus realiza un análisis de los requerimientos y envía el informe a administración de sede, quien aprueba y tramita la adquisición siempre y cuando sea factible.

Recomendación: Una vez realizada la adquisición de recursos, se procede a registrar los recursos adquiridos en el proceso de inventarios.

2.1.8.5. Diagrama BPMN

Proceso de adquisición de recursos

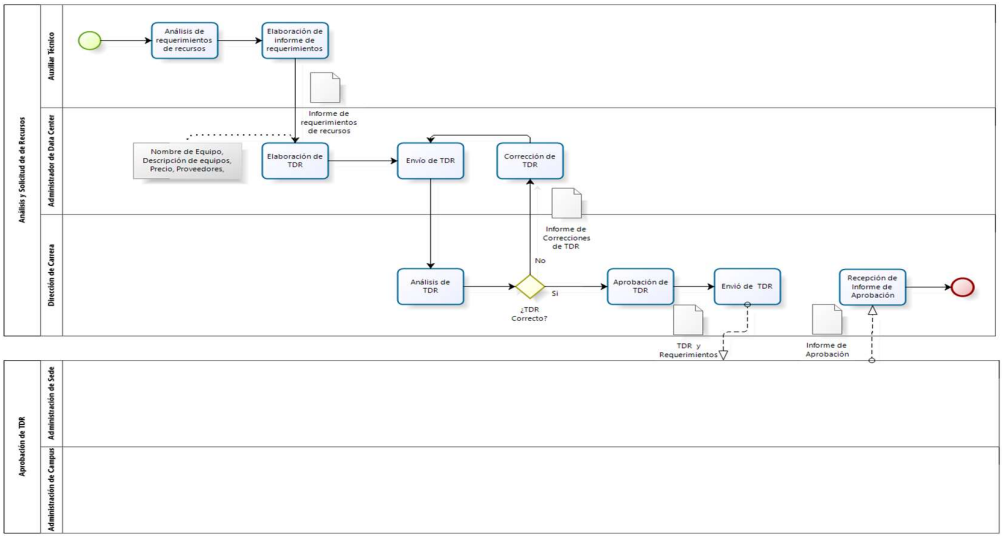


Figura 12. Diagrama BPMN proceso de adquisición de recursos.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.9. Proceso de comunicación

2.1.9.1. Objetivo

[2] Término de Referencia, documento en el cual se describen las características de los equipos, precios, proveedores, etc. (López, 2018)

Proporcionar una comunicación centralizada de todos los sistemas que componen el Data Center, para mantener informado a los responsables acerca de los errores, fallas y alertas que se producen en los recursos.

2.1.9.2. Alcance

Inicia con la notificación de errores, fallas y alertas producidas por algún sistema que compone el Data Center y finaliza con su solución o asistencia. Aplica para los sensores, servidores, cámaras de seguridad, biométricos, plataformas de administración y otros sistemas que se desarrollen a futuro. Las alertas deben enviarse a las personas que estén encargadas de la administración y soporte técnico del Data Center.

2.1.9.3. Responsables

Tabla 17. Responsables de proceso de comunicación.

Responsable	Cargo	Actividad
Ing. Jorge López MSc.	Administración de Data Center	• Supervisión de alertas.
Ing. Marcela Gallegos Ing. Thalia Ati	Auxiliares Técnicos	• Supervisión de alertas. • Solución y asistencia.

Nota: Proceso de comunicación.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.9.4. Descripción

El proceso de comunicación consiste en la recepción y solución de errores, fallas y alertas producidas por los sistemas que compone el Data Center. Cuando se produce una irregularidad, el sistema envía mediante correo electrónico una alerta con la información de lo ocurrido al administrador o auxiliar técnico encargado para su posterior solución. Esta información es registrada para su control y elaboración de informes pertinentes.

2.1.9.5. Diagrama BPMN

Proceso de comunicación

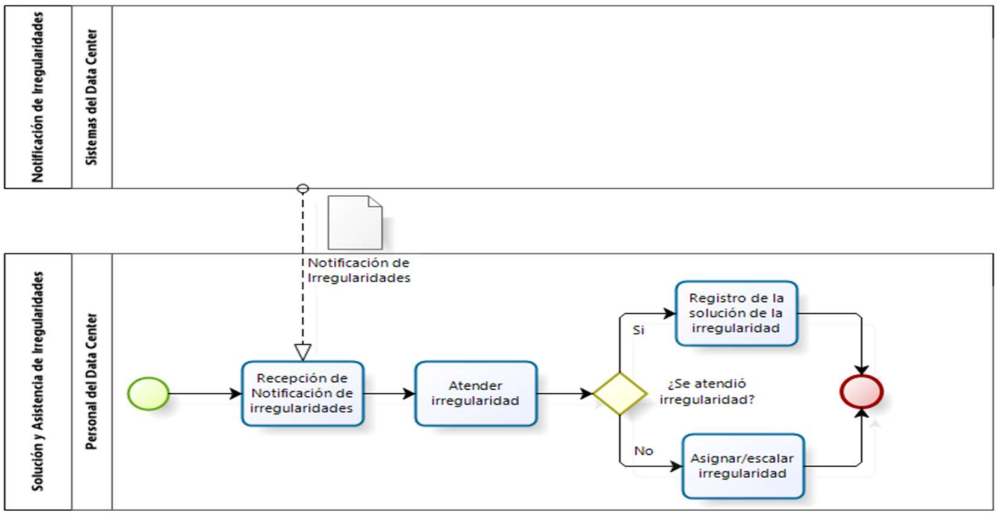


Figura 13. Diagrama BPMN proceso de comunicación.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.10. Proceso de asignación de horarios para uso de los laboratorios

2.1.10.1. Objetivo

Organizar y asignar los horarios de los laboratorios del ICC dependiendo de la disponibilidad de espacio físico y de los horarios académicos de los docentes, con la finalidad de evitar los cruces de horarios y permitir que los responsables de los laboratorios estén informados sobre el horario asignado a cada docente.

2.1.10.2. Alcance

Inicia con la organización de los horarios por nivel, docente y espacio físico, y finaliza con la entrega de un documento con el esquema de los horarios asignados a los responsables de los laboratorios del ICC. Aplica únicamente para los laboratorios del bloque D y embebidos. En la actualidad, las carreras que hacen uso del espacio físico en los laboratorios del ICC son: Ingeniería en Ciencias de la Computación e Ingeniería Electrónica, con la posibilidad de que a futuro se integren las demás carreras que lo requieran.

2.1.10.3. Responsables

Tabla 18. Responsables de asignación de horarios.

Responsable	Cargo	Actividad
Ing. Patsy Prieto MSc.	Dirección de Carrera	• Organización y envió de esquema de horarios.
Ing. Jorge López MSc.	Administración de Data Center	• Recepción de esquema de horarios.
Ing. Marcela Gallegos Ing. Thalia Ati	Auxiliares Técnico	• Recepción de esquema de horarios.

Nota: Proceso de asignación de horarios.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.10.4.Descripción

El proceso de asignación de horarios consiste en la organización del espacio físico que se utilizará en un periodo académico para impartir catedra. Dirección de carrera de Ingeniería en Ciencias de la Computación se encarga de organizar los horarios por nivel, docente y espacio físico para evitar cruces de horarios, posteriormente se verifica la existencia de algún cruce de horario y de darse el caso se realiza la corrección respectiva. Una vez finalizada la organización, se envía un documento con el esquema de los horarios a la dirección de las carreras del campus sur para que se asigne laboratorios a sus docentes dependiendo de los horarios disponibles en el esquema. Este documento se envía nuevamente a Dirección de Carrera de Ingeniería en Ciencias de Computación y posteriormente a los responsables de los laboratorios del ICC para que puedan realizar los procesos de control de accesos y mantenimiento de hardware y software dependiendo de los horarios ya asignados.

2.1.10.5.Diagrama BPMN

Proceso de asignación de horarios

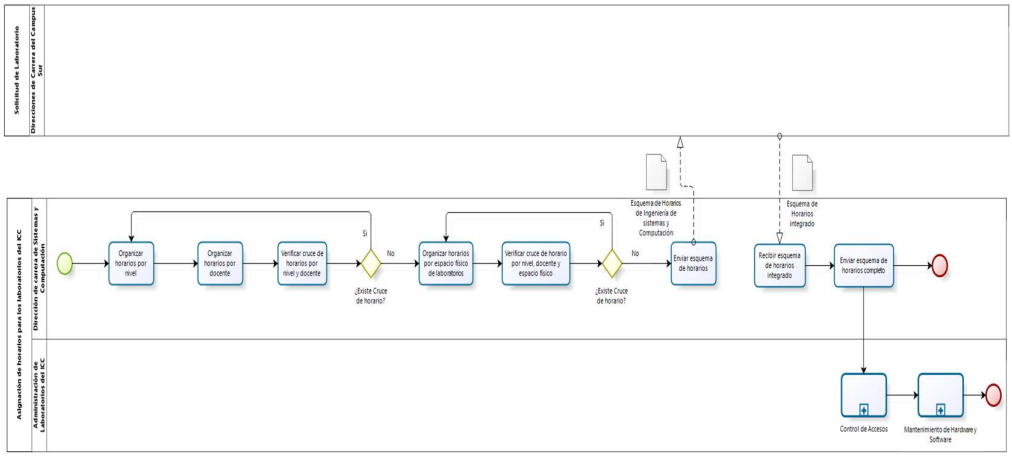


Figura 14. Diagrama BPMN proceso de asignación de horarios.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.11. Proceso de control de accesos

2.1.11.1.Objetivo

Asignar a los docentes y personal técnico externo autorización de entrada a los laboratorios del ICC, con el fin de mantener el control y registro de los accesos que se realizan en las instalaciones durante la jornada académica.

2.1.11.2.Alcance

Inicia con la recepción de los horarios académicos o petición de acceso a las instalaciones, y finaliza con el registro de la huella dactilar o tarjeta magnética en el sistema de los biométricos. Los accesos para los docentes aplican para un periodo académico a través de huella dactilar, y luego se da de baja su registro; mientras que para personal exterior se le otorga un acceso mediante una tarjeta magnética, por el tiempo que dure su labor. El docente podrá acceder únicamente al laboratorio y horario que ha sido asignado por parte de dirección de carrera.

2.1.11.3.Responsables

Tabla 19. Responsables de control de accesos.

Responsable	Cargo	Actividad
Ing. Patsy Prieto MSc.	Dirección de Carrera	• Envió de notificación de horarios

Ing. Marcela Gallegos Ing. Thalia Ati	Auxiliares Técnico	• Registro de acceso.

Nota: Proceso de control de accesos.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.1.11.4.Descripción

El control de accesos consiste en el envío de los horarios por parte de dirección de carrera a los auxiliares técnicos del Data Center y laboratorios del ICC, para posteriormente registrar el acceso a los docentes en el sistema biométrico. El auxiliar técnico notifica al docente para que acuda a registrar sus huellas dactilares en el sistema biométrico, mientras que, para un usuario externo a la universidad, se le otorga una tarjeta magnética que le dará acceso al laboratorio durante el tiempo que realice su labor.

2.1.11.5.Diagrama BPMN

Proceso de control de accesos

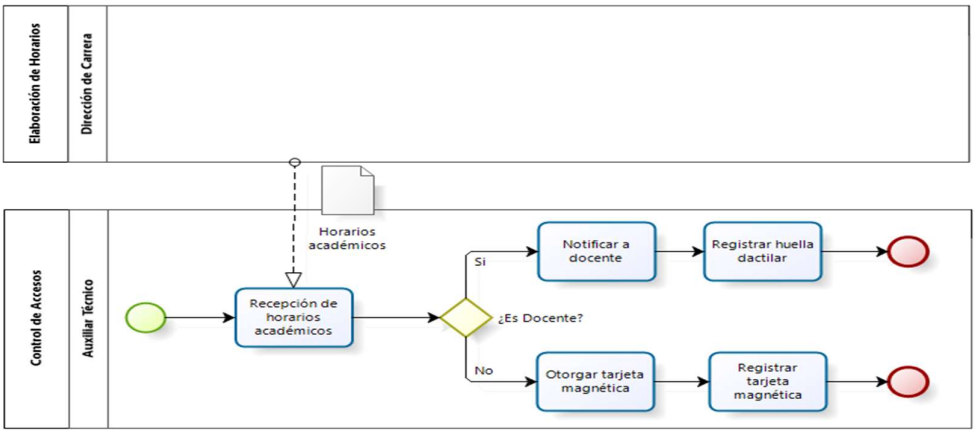


Figura 15. Diagrama BPMN proceso de control de accesos.
Elaborado por: Andres Chisaguano, Joel Ludeña.

2.2. Análisis de factibilidad

El siguiente análisis de factibilidad se realizó con la finalidad de determinar si el proyecto desarrollado es viable para los involucrados, tanto en la parte técnica, operacional, económica y legal.

2.2.1. Factibilidad técnica

Para la elaboración del proyecto será necesario contar con las siguientes características de hardware y software:

Tabla 20. Factibilidad Técnica para el desarrollo del sistema.

Recurso	Nombre	Característica
Servidor	Servidor	100 GB Disco 8 GB RAM
Sistema operativo	Windows Server	2012 R2 o superior
Base de datos	PostgreSQL	9.5
IDE de desarrollo	Visual Studio	Versión 15.8 o superior
	Android Studio	Versión 2.2.3 o superior
Estaciones de trabajo	Computadoras personales	Navegadores Web actualizados
	Tablets o Smarthphones	Versión de Android 6.0 o superior
Impresora	-	Impresión para papel adhesivo

Nota: Características de hardware y software para el desarrollo del sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.2.2. Factibilidad operacional

Los involucrados del proyecto buscan en esta factibilidad darle el correcto uso y aprovechar los beneficios que ofrece el sistema. Para lo cual, se han determinado los siguientes factores:

- Con la finalidad de evitar la resistencia por parte de los usuarios, el sistema deberá ser amigable e intuitivo.
- Se deben establecer roles para que la información que se generé no pueda ser alterada por algún usuario no autorizado. El sistema hará uso de validaciones de los campos de ingreso para evitar errores de ingreso por parte de los usuarios.
- Los responsables del Data Center y laboratorios de ICC poseen el conocimiento necesario para utilizar el sistema, debido a que ya conocer la información existente y su gestión. Sin embargo, se proporcionarán capacitaciones y manuales de usuario.

- La tecnología adecuada para la implementación y funcionamiento del sistema existe. Además, el sistema contará con opciones que permitan el cambio de administración de darse el caso o de implementarse nuevas políticas.

2.2.3. Factibilidad económica

Para el desarrollo del sistema, se han considerado los costos relacionados con el equipo de desarrollo, recursos tecnológicos e insumos varios que son necesarios para su correcto desenvolvimiento. En la Tabla No. 21 se detallan los costos y recursos que se utilizaron en un periodo aproximadamente de 5 meses que tuvo duración el proyecto:

Tabla 21. Factibilidad económica de los recursos para el desarrollo del sistema.

Descripción	Cantidad	Costo	Costo Total
Recursos Humanos			
Desarrollador Jr.	2	$600 / Mes	$6000
Recursos Tecnológicos			
Licencia Windows Server 2012 R2	1	$726	$726
Licencia de Visual Studio Community 2017 [3]	1	$0	$0
PostgreSQL 9.5	1	$0	$0
Insumos Varios			
Resma de 50 unidades de papel adhesivo A4	1	$6	$6
Servicios básicos	-	$40 / Mes	$200
Movilización	-	$20 / Mes	$100
		Total:	**$7032**

Nota: Costos aproximados de los recursos para el desarrollo del sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al ser un proyecto académico, los costos de los recursos tecnológicos fueron cubiertos por la Universidad Politécnica Salesiana sede Quito Campus Sur, al momento de otorgarse una máquina virtual para implementar el sistema. Los demás valores fueron cubiertos por los autores del proyecto. Los valores de los costos

[3] La licencia de Visual Studio Community 2017 no tiene costo, siempre y cuando se tenga menos de 5 desarrolladores individuales. Se puede desarrollar y comercializar sin tener una licencia de pago. (Microsoft, 2017)

mencionados en la Tabla No. 21, están considerados para una implementación en un entorno empresarial.

2.2.4. Factibilidad legal

Debido a que, para la identificación de los bienes es política de la Universidad Politécnica Salesiana colocar un código de barras, se consultó con los responsables del Data Center y los laboratorios del ICC con la finalidad de evitar inconvenientes legales sobre la colocación de un código QR en los activos de TI; llegando a la conclusión de que, al ser un proyecto interno, no incumpliría con la normativa, ya que no se busca reemplazar el código de barras, sino de brindar una herramienta de apoyo para agilizar la gestión, control y cuidado de los bienes provistos por la universidad.

2.3. Especificación de requisitos

Se presenta a continuación la descripción de los requisitos funcionales y no funcionales solicitados por los involucrados del proyecto.

2.3.1. Requisitos funcionales

2.3.1.1. Sistema de gestión de inventarios

2.3.1.1.1. Login

Tabla 22. Requisito funcional para el Login de usuarios.

Identificación	RF01
Nombre del requerimiento	Login de usuarios
Pre-Condición	Que el usuario se encuentre registrado en la base de datos.
Proceso	El usuario ingresará a un formulario donde constan los campos para acceder al sistema. El usuario deberá ingresar su Nick y su contraseña con los cuales fue registrado al sistema.
Postcondición	La contraseña por ingresar debe ser mayor a 6 caracteres.
Salida	El usuario ha ingresado correctamente al sistema.
Prioridad	Alta

Nota: Especificación para ingreso de nuevos activos de TI en el inventario.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.1.1.2. Ingreso de nuevos activos de TI

Tabla 23. Requisito funcional para el ingreso de nuevos activos de TI.

Identificación	RF02
Nombre del requerimiento	Ingreso de nuevos activos de TI.
Pre-Condición	Ingresar al módulo de registro de activos de TI.
Proceso	El usuario ingresará a un formulario donde constan los campos para el registro del activo de TI. Una vez registrado el activo de TI se generará un código QR.
Postcondición	El activo de TI es registrado dentro de la base de datos y el código QR será impreso para colocarlo en el activo de TI.
Salida	El activo de TI se ha registrado correctamente.
Prioridad	Alta

Nota: Especificación para ingreso de nuevos activos de TI en el inventario.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.1.1.3. Consulta de activos de TI por código QR

Tabla 24. Requisito funcional para la consulta de activos de TI por código QR.

Identificación	RF03
Nombre del requerimiento	Consulta de activos de TI por código QR.
Pre-Condición	Que el activo de TI esté registrado y cuente con el código QR. Ingresar al módulo de consultas.
Proceso	El usuario podrá consultar la información de un activo de TI mediante un dispositivo móvil haciendo uso del código QR correspondiente.
Postcondición	Ninguna.
Salida	La información del activo de TI.
Prioridad	Alta

Nota: Especificación para consulta de activos de TI en el inventario por código QR.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.1.1.4. Consulta general de los activos de TI

Tabla 25. Requisito funcional para la consulta general de los activos de TI.

Identificación	RF04
Nombre del requerimiento	Consulta general de los activos de TI
Pre-Condición	Que existan activos de TI en la base de datos. Ingresar al módulo de consultas.
Proceso	El usuario visualizará la tabla en donde se muestra la información de los activos de TI registrados, además podrá filtrar los registros dependiendo a los criterios de búsqueda establecidos.
Salida	La información del activo de TI.
Prioridad	Alta

Nota: Especificación para consulta general de los activos de TI en el inventario.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.1.1.5. Actualización de activos de TI

Tabla 26. Requisito funcional para la actualización de activos de TI.

Identificación	RF05
Nombre del requerimiento	Actualización de activos de TI
Pre-Condición	Que exista el activo de TI que se quiera actualizar en la base de datos. Ingresar al módulo de actualización. Buscar el activo de TI que se quiere modificar.
Proceso	El usuario podrá modificar los datos del activo de TI requerido.
Postcondición	Ninguna.
Salida	El activo de TI se ha modificado correctamente.
Prioridad	Alta

Nota: Especificación para actualización de los activos de TI en el inventario.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.1.1.6. Asignación de activos de TI a un laboratorio

Tabla 27. Requisito funcional para la asignación de activos de TI a un laboratorio.

Identificación	RF06
Nombre del requerimiento	Asignación de activos de TI
Pre-Condición	Que exista el activo de TI que se quiera actualizar en la base de datos. Ingresar al módulo de actualización. Buscar el activo de TI que se quiere modificar.
Proceso	El usuario podrá asignar el activo de TI al respectivo laboratorio modificando su ubicación. El usuario podrá crear un nuevo laboratorio.
Postcondición	Ninguna.
Salida	El activo de TI se ha modificado correctamente.
Prioridad	Alta

Nota: Especificación para asignación de activos de TI a un laboratorio del inventario.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.1.1.7. Gestión de usuarios

Tabla 28. Requisito funcional para la gestión de usuarios.

Identificación	RF07
Nombre del requerimiento	Gestión de usuarios
Pre-Condición	Ninguna.
Proceso	El usuario Administrador podrá crear nuevos usuarios para que accedan al sistema. Además, podrá asignar roles y permisos, así como su modificación. Cada usuario podrá modificar su información personal en su perfil de usuario.
Postcondición	Ninguna.
Salida	Ninguna.
Prioridad	Alta

Nota: Especificación para gestión de Usuarios.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.1.1.8. Reportería

Tabla 29. Requisito funcional para reportería.

Identificación	RF08
Nombre del requerimiento	Reportería
Pre-Condición	Que existan activos de TI en la base de datos.
Proceso	El usuario podrá obtener reportes en formato PDF o Excel de los activos de TI registrados.
Postcondición	Ninguna.
Salida	Reportes en formato PDF o Excel.
Prioridad	Media

Nota: Especificación para reportería de los activos del inventario.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.1.2. Sistema de ticketing para soporte técnico

2.3.1.2.1. Registro de ticket

Tabla 30. Requisito funcional para el registro de tickets.

Identificación	RF09
Nombre del requerimiento	Registro de ticket
Pre-Condición	Que se suscite un evento inesperado en los recursos o laboratorios del ICC.
Proceso	El usuario ingresará a un formulario en donde registrará un error o evento inesperado en algún activo de TI o laboratorio del ICC. Además, el usuario contará con la opción de hacer el registro de un error a través de la lectura del código QR de cada activo de TI.
Postcondición	Ninguna.
Salida	Mensaje de éxito.
Prioridad	Alta

Nota: Especificación para registro de tickets para soporte técnico.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.1.2.2. Consulta de estado de ticket

Tabla 31. Requisito funcional para la consulta de estado de ticket.

Identificación	RF10
Nombre del requerimiento	Consulta de estado de ticket
Pre-Condición	Que existan tickets registrados.
Proceso	El usuario ingresará a un formulario en donde podrá consultar el estado del ticket que reportó.
Postcondición	Ninguna.
Salida	Estado de ticket.
Prioridad	Media

Nota: Especificación para consulta de estado de tickets para soporte técnico.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.1.2.3. Gestión de tickets

Tabla 32. Requisito funcional para la gestión de tickets.

Identificación	RF11
Nombre del requerimiento	Gestión de tickets
Pre-Condición	Que existan tickets registrados.
Proceso	Los usuarios con permisos para gestión de tickets podrán visualizar los tickets reportados por los usuarios para su posterior solución. Una vez solucionado el ticket, se podrá cambiar su estado e incluir una observación para notificar al usuario.
Salida	Estado de ticket y observación.
Prioridad	Alta

Nota: Especificación para gestión de tickets para soporte técnico.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.2. Requisitos no funcionales

2.3.2.1. Interfaz de usuario

Tabla 33. Requisito no funcional para la interfaz de usuario.

Identificación	RNF01
Nombre del requerimiento	Interfaz de usuario
Descripción	1. Los módulos de la aplicación deben ser fáciles de usar e intuitivos, permitiendo el flujo de información de una forma ágil y dinámica. 2. La aplicación podrá adaptarse al tamaño de pantalla de dispositivos móviles (smartphones/tablets), y se podrá visualizar en navegadores Web como Mozilla, Microsoft Edge, Google Chrome u Opera. 3. La aplicación web tendrá una estructura organizada, dividiendo el contenido en un menú vertical que contendrá las principales funciones como: Ingreso, actualización, asignación y consulta de activos, reportería, gestión de usuarios y roles, gestión de tickets para soporte técnico. 4. La aplicación en Android tendrá una estructura organizada, dividiendo el contenido en un menú que contendrá las principales funciones como: Consulta y actualización de activos de TI, registro y consulta de tickets para soporte técnico. 5. En los formularios de ingreso y actualización, se validarán los campos dependiendo de su tipo. 6. Los fallos relacionados con los procesos dentro del sistema serán mostrados mediante mensajes o alertas.
Prioridad	Media

Nota: Especificación de la interfaz de usuario.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.2.2. Arquitectura

Tabla 34. Requisito no funcional para la arquitectura.

Identificación	RNF02
Nombre del requerimiento	Arquitectura
Descripción	1. El sistema tendrá una aplicación web y una aplicación móvil en Android, en los cuales se podrá acceder a la información. 2. La información que se obtiene de ambas aplicaciones se almacenará en una base de datos PostgreSQL. 3. La aplicación web será desarrollada bajo el patrón MVC.
Prioridad	Media

Nota: Especificación de la arquitectura.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.2.3. Seguridad

Tabla 35. Requisito no funcional para la seguridad.

Identificación	RNF03
Nombre del requerimiento	Seguridad
Descripción	1. La aplicación será instalada únicamente en los dispositivos proporcionados por los responsables de los laboratorios del ICC. 2. Para el acceso al sistema, será necesario hacer uso de las credenciales de cada usuario. 3. Para aumentar la seguridad de las aplicaciones, la contraseña será cifrada. 4. La información del sistema solamente podrá ser modificada por los usuarios registrados en el sistema. 5. El rol Administrador será el único que podrá crear nuevos roles y registrar nuevos usuarios. 6. El perfil Administrador tendrá acceso a todas las funcionalidades del sistema, tales como: Ingreso, actualización, asignación y consulta de activos, reportería, gestión de usuarios y roles, gestión de tickets para soporte técnico. 7. Se registrarán las actividades de los usuarios realizadas dentro del sistema referente al CRUD mediante logs.
Prioridad	Alta

Nota: Especificación de la seguridad.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.2.4. Interfaz de comunicación

Tabla 36. Requisito no funcional para la interfaz de comunicación.

Identificación	RNF04

Nombre del requerimiento	Interfaces de comunicación
Descripción	1. La aplicación web se basará en el protocolo HTTP y estándares de la web. 2. Se utilizará Web Service REST para la comunicación entre las aplicaciones Web, Android y la base de datos.
Prioridad	Media

Nota: Especificación de la interfaz de comunicación.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.2.5. Rendimiento y disponibilidad

Tabla 37. Requisito no funcional para el rendimiento y disponibilidad.

Identificación	RNF05
Nombre del requerimiento	Rendimiento y disponibilidad
Descripción	1. La aplicación deberá ser tolerante a fallos. 2. El sistema funcionará únicamente en la red interna de la universidad. 3. Las peticiones concurrentes realizadas en la base de datos no afectarán al funcionamiento del sistema. 4. La disponibilidad del sistema dependerá del correcto funcionamiento del servidor del Data Center.
Prioridad	Media

Nota: Especificación del rendimiento y disponibilidad.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.3.2.6. Mantenibilidad

Tabla 38. Requisito no funcional para la mantenibilidad.

Identificación	RNF06
Nombre del requerimiento	Mantenibilidad
Descripción	Para el correcto uso del sistema, se definirá documentación técnica y funcional referente a las funciones del sistema para disminuir los costos de su mantenimiento. En la documentación técnica se elaborará el manual de configuración de la plataforma, mientras que para la documentación funcional un manual de usuario.
Prioridad	Media

Nota: Especificación de la mantenibilidad.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4. Diagramas UML

Se presenta a continuación la descripción de los diagramas de casos de uso y de secuencia para los principales requisitos funcionales descritos previamente.

2.4.1. Inicio de sesión al sistema

2.4.1.1. Especificación del caso de uso

Tabla 39. Especificación del ingreso de inventarios de activos de TI.

Caso de uso	Iniciar sesión en el sistema
Actores	Usuarios registrados en el sistema (Administradores, pasantes, reportería, etc.)
Funciones básicas	Ingresar al sistema. Proporcionar credenciales. Permitir acceso al sistema.
Tipo	Primario

Nota: Especificación del caso de uso.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.1.2. Flujo de actividades entre el actor y el sistema

Tabla 40. Flujo de actividades del ingreso de inventarios de activos de TI.

Actividades de los actores	Acciones del sistema
1. El caso de uso empieza cuando los usuarios requieren ingresar al sistema de inventarios.	
2. Los usuarios ingresan sus credenciales en el formulario de inicio de sesión.	3. El sistema verifica las credenciales otorgadas por el usuario.
	4. El sistema permite el acceso si las credenciales son correctas.
	5. Si se introducen credenciales incorrectas, se producirá una excepción mostrando un mensaje de error.
6. Ingreso al sistema de gestión de inventarios y ticketing.	7. El sistema registra el log de inicio de sesión.

Nota: Flujo de actividades entre el actor y el sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.1.3. Diagrama de caso de uso UML

Inicio de sesión al sistema

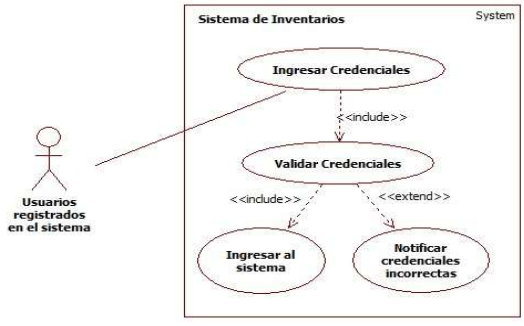


Figura 16. Diagrama de caso de uso de inicio de sesión en el sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.1.4. Diagrama de secuencia UML

Inicio de sesión al sistema

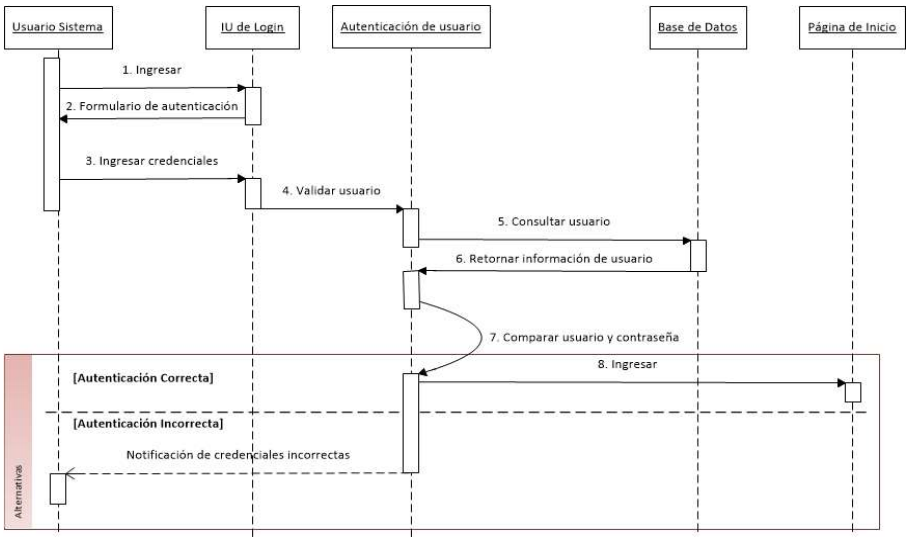


Figura 17. Diagrama de secuencia de inicio de sesión en el sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.2. Ingreso de activos de TI en el inventario

2.4.2.1. Especificación del caso de uso

Tabla 41. Especificación del ingreso de activos de TI en el inventario.

Caso de uso	Inventariar activos de TI
Actores	Usuarios con permisos para la gestión de activos de TI (Administradores, pasantes, etc.).
Funciones básicas	Capturar información del activo de TI. Registrar el activo de TI. Imprimir el código QR del activo.
Tipo	Primario

Nota: Especificación del caso de uso.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.2.2. Flujo de actividades entre el actor y el sistema

Tabla 42. Flujo de actividades del ingreso de activos de TI en el inventario

Actividades de los actores	Acciones del sistema
1. El caso de uso empieza cuando se recibe el activo de TI.	
2. Los responsables recolectan la información del activo de TI.	
3. Los responsables ingresan la información del activo de TI.	4. Si existen campos obligatorios que no han sido ingresados, se producirá un mensaje de alerta.
	5. El sistema registra el activo de TI y almacena en la base de datos.
	6. El sistema registra el log de

	ingreso.
7. Los responsables imprimen el código QR del activo.	
8. Los responsables colocan el código QR en el activo.	

Nota: Flujo de actividades entre el actor y el sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.2.3. Diagrama de caso de uso UML

Ingreso de inventarios de activos de TI en el inventario

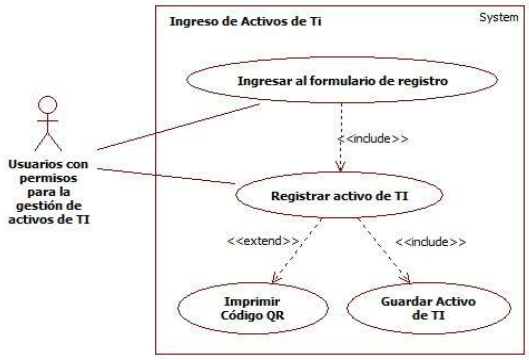


Figura 18. Diagrama de caso de uso de ingreso de activos de TI en el inventario.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.2.4. Diagrama de secuencia UML

Ingreso de inventarios de activos de TI en el inventario

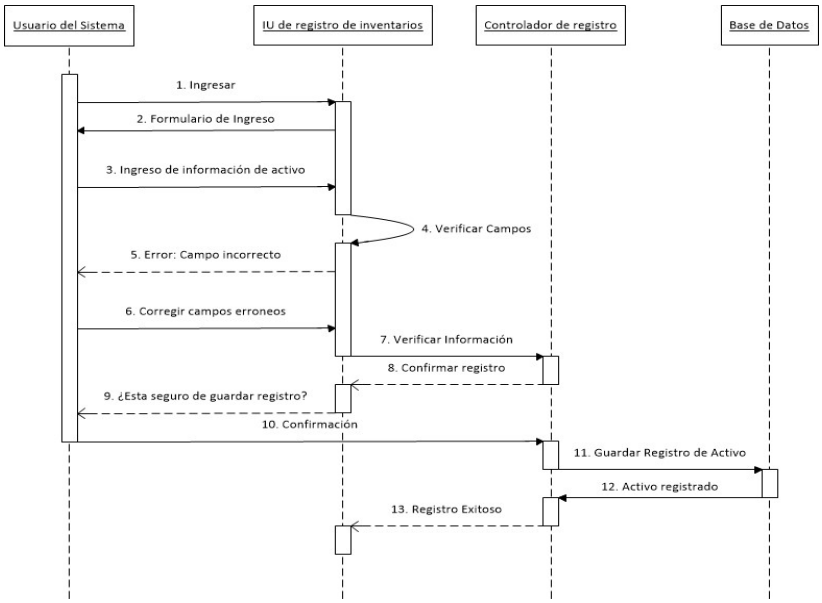


Figura 19. Diagrama de secuencia de ingreso de activos de TI en el inventario.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.3. Consulta de activos de TI

2.4.3.1. Especificación del caso de uso

Tabla 43. Especificación de la consulta de activos de TI.

Caso de uso	Consultar activos de TI
Actores	Usuarios con permisos para la gestión de activos de TI (Administradores, pasantes, etc.).
Funciones básicas	Verificar información del activo de TI. Realizar reportes de la información.
Tipo	Secundario

Nota: Especificación del caso de uso.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.3.2. Flujo de actividades entre el actor y el sistema

Tabla 44. Flujo de actividades de la consulta de activos de TI.

Actividades de los actores	Acciones del sistema
1. Los usuarios autorizados consultan el activo de TI mediante el código QR en la aplicación Android o un filtro de búsqueda en la aplicación web.	2. El sistema verifica los parámetros del activo y entrega la información.
3. Los usuarios autorizados obtienen la información consultada a través de tablas en la aplicación web o en un formulario en la aplicación Android.	4. Si el activo es consultado a través de la aplicación Android, no se podrá realizar reportería. Si el código QR escaneado no se encuentra registrado en la base de datos, se producirá un mensaje de alerta.
5. En la aplicación web, los usuarios autorizados podrán realizar reportes sobre la información obtenida.	6. El sistema verifica el formato del reporte (PDF, Excel) y entrega el mismo con la información solicitada.
	7. El sistema registra el log de consulta.

Nota: Flujo de actividades entre el actor y el sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.3.3. Diagrama de caso de uso UML

Consulta de activos de TI

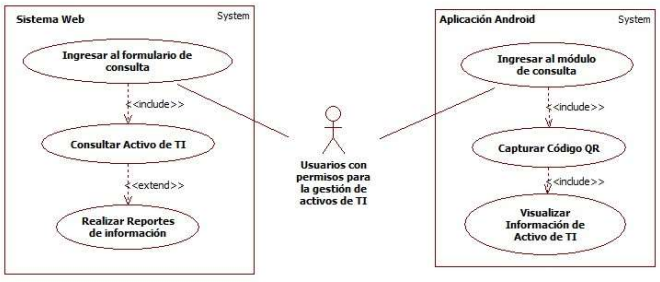


Figura 20. Diagrama de caso de uso de consulta de activos de TI.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.3.4. Diagrama de secuencia UML

Consulta de activos de TI

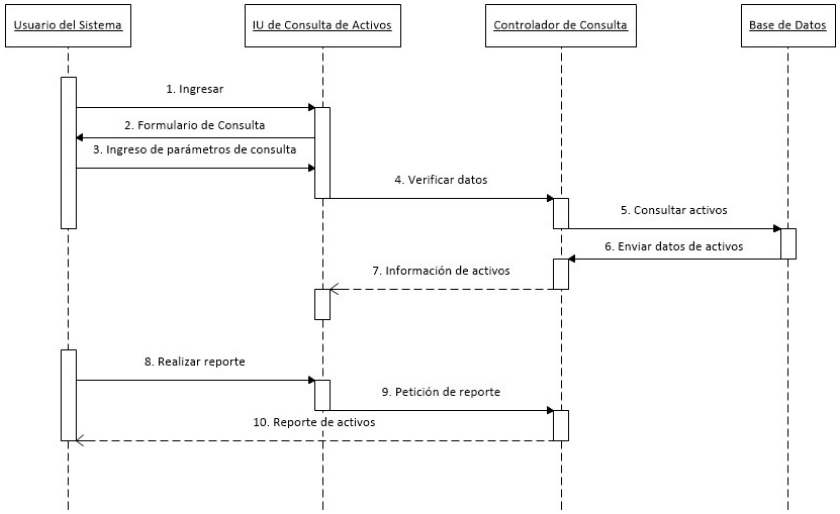


Figura 21. Diagrama de secuencia de consulta de activos de TI.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.4. Registro de tickets para soporte técnico

2.4.4.1. Especificación del caso de uso

Tabla 45. Especificación del registro de tickets para soporte técnico.

Caso de uso	Registro de tickets para soporte técnico
Actores	Usuarios con permisos para generar tickets (docentes, generadores de tickets, etc.).
Funciones básicas	Ingresar información del error o evento inesperado. Seleccionar tipo de ticket para soporte técnico. Ingresar o capturar la información del activo de TI o laboratorio.
Tipo	Primario

Nota: Especificación del caso de uso.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.4.2. Flujo de actividades entre el actor y el sistema

Tabla 46. Flujo de actividades del registro de tickets para soporte técnico.

Actividades de los actores	Acciones del sistema
1. El caso de uso empieza cuando el usuario de algún laboratorio del ICC requiere la atención de soporte técnico por algún error o evento inesperado de un activo de TI o un laboratorio.	
2. El usuario selecciona si desea reportar un error en un activo de TI o en un laboratorio.	3. El sistema muestra el formulario dependiendo de la opción seleccionada por el usuario.
4. Si la opción seleccionada es un error	5. El sistema consulta el código QR

de activo de TI, el usuario captura el código QR e ingresa la información del error.	del activo de TI en la base de datos para retornar su información respectiva.
6. Si la opción seleccionada es un evento inesperado de un laboratorio del ICC, el usuario seleccionará el laboratorio e ingresará una descripción.	7. Si existen campos obligatorios que no han sido ingresados, se producirá un mensaje de alerta.
	8. El sistema registra el ticket para soporte técnico.
	9. El sistema registra el log de ingreso ticket.

Nota: Flujo de actividades entre el actor y el sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.4.3. Diagrama de caso de uso UML

Registro de tickets para soporte técnico

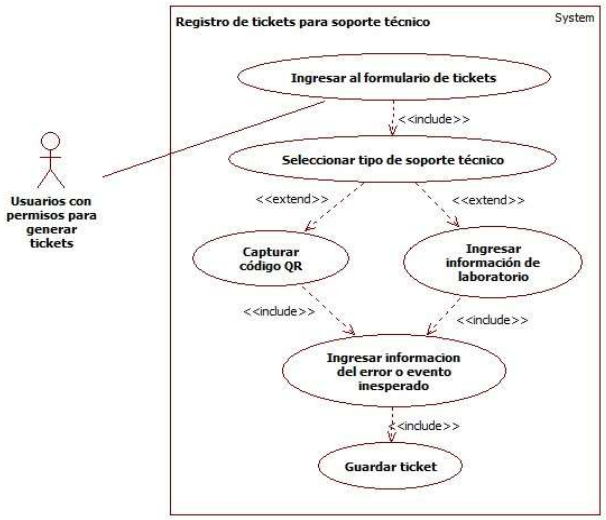


Figura 22. Diagrama de caso de uso de registro de tickets para soporte técnico.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.4.4. Diagrama de secuencia UML

Registro de tickets para soporte técnico

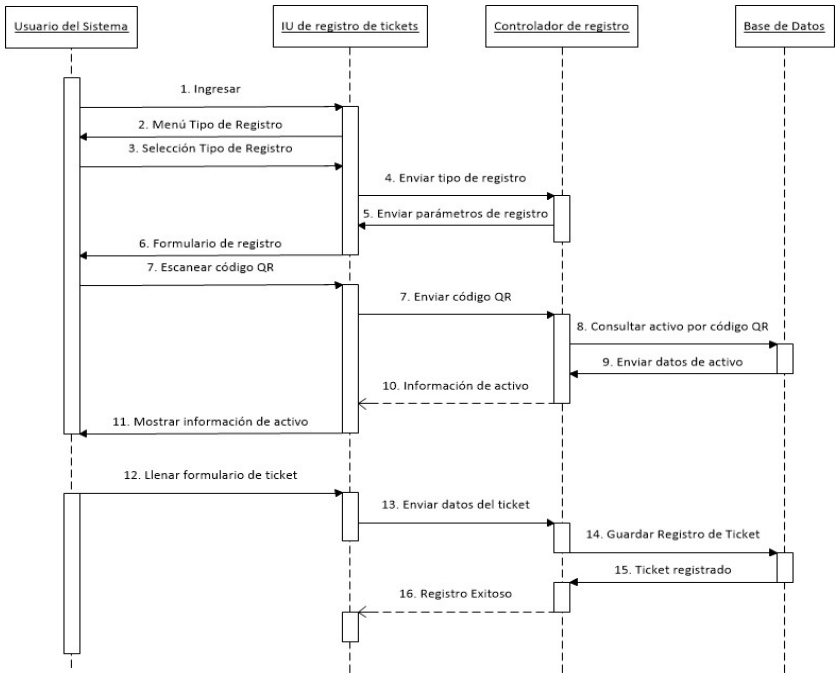


Figura 23. Diagrama de secuencia de registro de tickets para soporte técnico.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.5. Gestión de tickets para soporte técnico

2.4.5.1. Especificación del caso de uso

Tabla 47. Especificación de la gestión de tickets para soporte técnico.

Caso de uso	Gestión de tickets para soporte técnico
Actores	Usuarios con permisos para la gestión de tickets (administradores, pasantes, etc.).
Funciones básicas	Verificar información del ticketing para soporte técnico. Resolver error o evento inesperado. Actualizar estado del ticket.
Tipo	Primario

Nota: Especificación del caso de uso
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.5.2. Flujo de actividades entre el actor y el sistema

Tabla 48. Flujo de actividades de la gestión de tickets para soporte técnico.

Actividades de los actores	Acciones del sistema
1. El caso de uso empieza cuando el usuario autorizado ingresa al módulo de gestión de tickets para soporte técnico.	2. El sistema retorna un listado de los tickets reportados por los usuarios.
3. El usuario autorizado analiza la información del ticket y da solución al problema.	

4. Si el problema fue solucionado, el auxiliar técnico cambia de estado al ticket.	5. El sistema actualiza la información del ticket.
	6. Registrar log de actualización de estado de ticket.

Nota: Flujo de actividades entre el actor y el sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.5.3. Diagrama de caso de uso UML

Gestión de tickets para soporte técnico

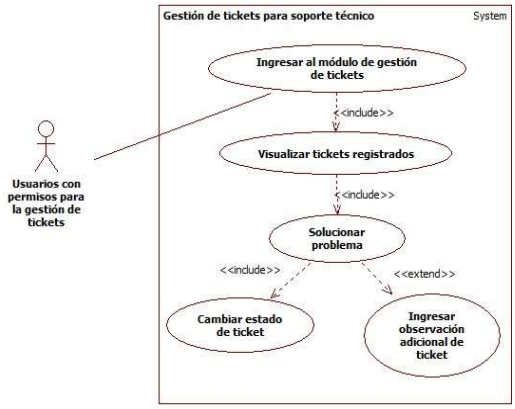


Figura 24. Diagrama de caso de uso de gestión de tickets para soporte técnico.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.4.5.4. Diagrama de secuencia UML

Gestión de tickets para soporte técnico

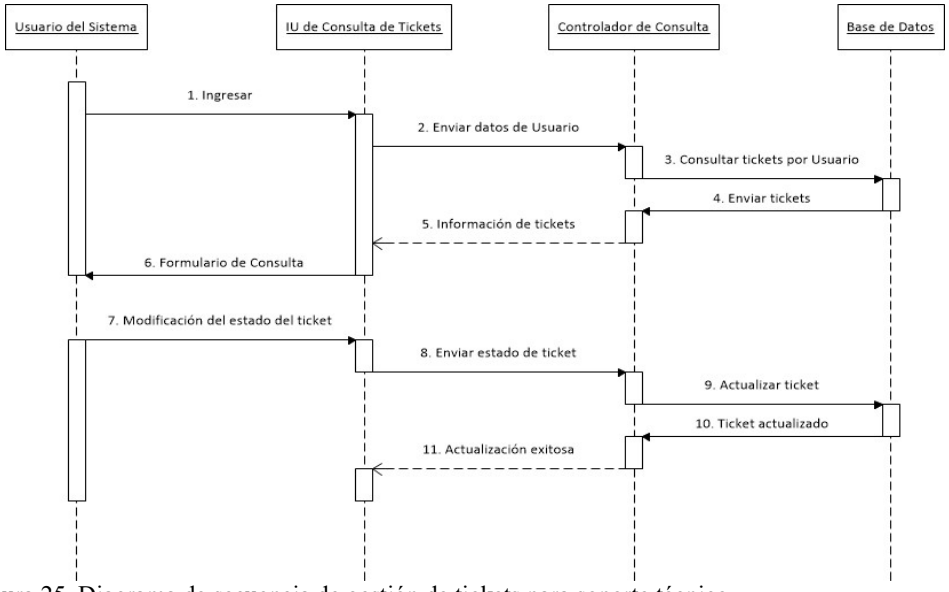


Figura 25. Diagrama de secuencia de gestión de tickets para soporte técnico.
Elaborador por: Andres Chisaguano, Joel Ludeña.

2.5. Diagrama entidad – relación

El diagrama entidad – relación permitirá comprender la lógica del negocio para la elaboración del modelo físico y lógico con el cual se desarrollará la base de datos.

Diagrama Entidad – Relación

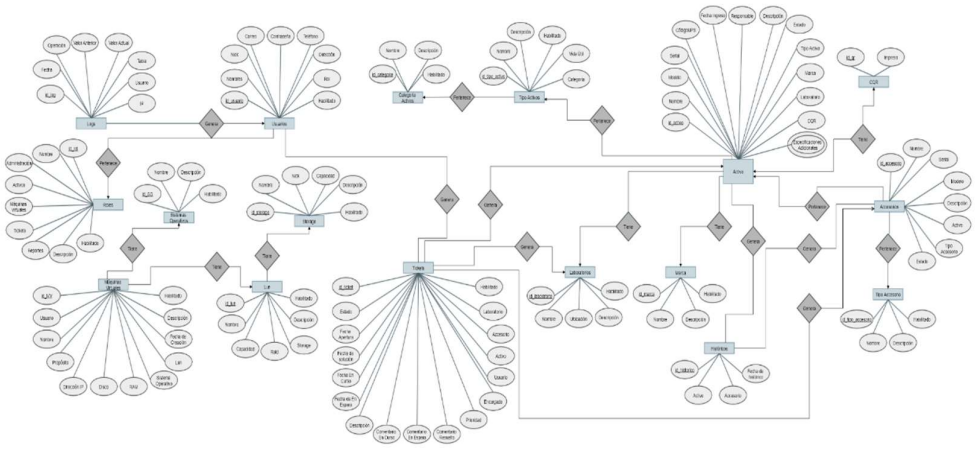


Figura 26. Diagrama entidad - relación de inventarios y ticketing.
Elaborado por. Andres Chisaguano, Joel Ludeña

CAPÍTULO 3

3. Construcción del sistema

3.1. Base de datos

3.1.1. Estándares para base de datos

A continuación, se define el estándar que se utilizará en la implementación de la base de datos.

3.1.1.1. Estándar de tablas

1. El nombre de las tablas iniciará con el nombre de la organización en mayúscula (DCICC), seguido de un guion bajo y finalmente el nombre de la tabla.
2. Se hará uso de nombres que representen el concepto de cada tabla.
3. Para la definición del nombre de las tablas se utilizará la notación "CamelCase", en el cual el nombre comienza con la letra inicial en mayúscula y el resto con minúscula. En caso de que el nombre de la tabla sea compuesto, la primera letra de la segunda palabra debe comenzar en mayúscula y así sucesivamente.
4. Los nombres de las tablas estarán compuestos únicamente por tres palabras.
5. El número máximo de caracteres que tendrá el nombre de cada tabla es de 30.
6. En caso de que el nombre de la tabla sea mayor a 30 caracteres, la segunda o tercera palabra del nombre será una abreviatura de tres caracteres para cumplir con el estándar.

Ejemplos:

1. DCICC_NombreTabla.
2. DCICC_Usuarios.
3. DCICC_CategoriaActivo.
4. DCICC_ControlHorariosDocente.
5. DCICC_ControlHorariosDocente (28 Caracteres).

6. DCICC_EspecificacionesAdicionales (33 Caracteres) - DCICC_EspecificacionesAdi (28 Caracteres).

3.1.1.2. Estándar de Campos

1. El nombre de los campos iniciará con el nombre en minúscula, seguido de un guion bajo y finalmente el nombre referente a la tabla.
2. Para la definición del nombre de los campos se utilizará la notación "CamelCase" con una variación, en el cual el nombre comienza con la primera palabra en minúscula y la primera letra de la segunda palabra debe comenzar en mayúscula y así sucesivamente.
3. Los nombres de los campos estarán compuestos únicamente por tres palabras.
4. El número máximo de caracteres que tendrá el nombre de cada campo es de 35.
5. En caso de que el nombre del campo sea mayor a 35 caracteres, la segunda o tercera palabra del nombre será una abreviatura de tres caracteres para cumplir con el estándar.
6. Se omitirán las tildes y caracteres especiales.

Ejemplos:

1. nombreCampo_Tabla
2. idCategoria_CategoriaActivo
3. hpePartNumber_ActivoTi
4. ubicacion_Laboratorio (22 Caracteres)
5. velocidadTransferenciaDisco_ActivoTI (37 Caracteres) - velocidadTransferenciaDis_ActivoTI (35 Caracteres)

3.1.2. Diagrama conceptual

El diagrama conceptual de inventarios y ticketing permitirá representar el modelo entidad – relación en un diagrama en el cual constan las entidades y relaciones necesarias para la construcción de la base de datos.

Diagrama conceptual

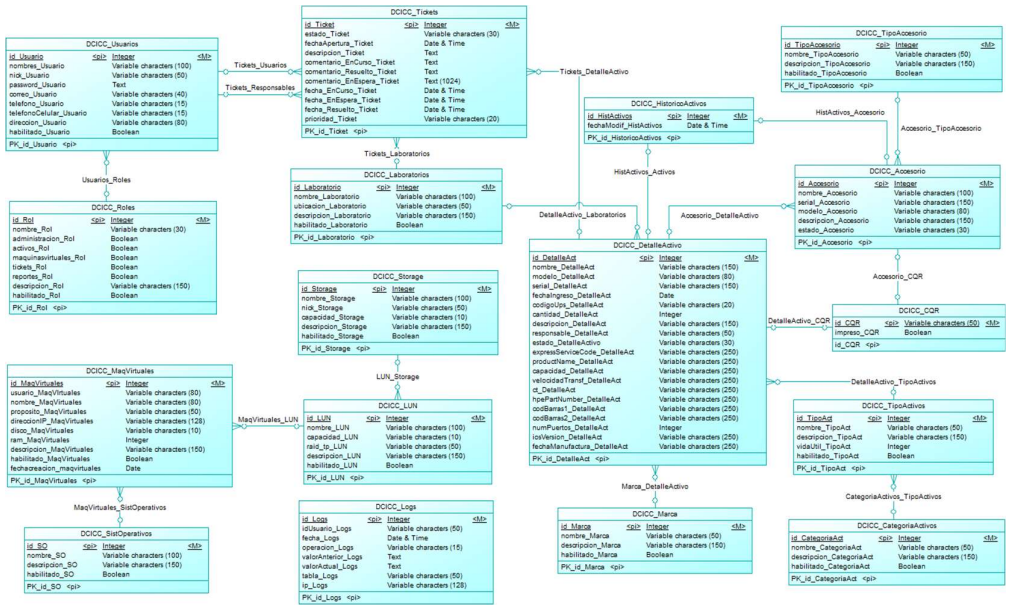


Figura 27. Diagrama conceptual de inventarios y ticketing.
Elaborado por: Andres Chisaguano, Joel Ludeña.

3.1.3. Diagrama físico

El diagrama físico presenta las tablas y relaciones obtenidas a partir del diagrama conceptual de la base de datos, en el cual se presentan las características de cada entidad con su respectivo tipo de dato.

Diagrama físico

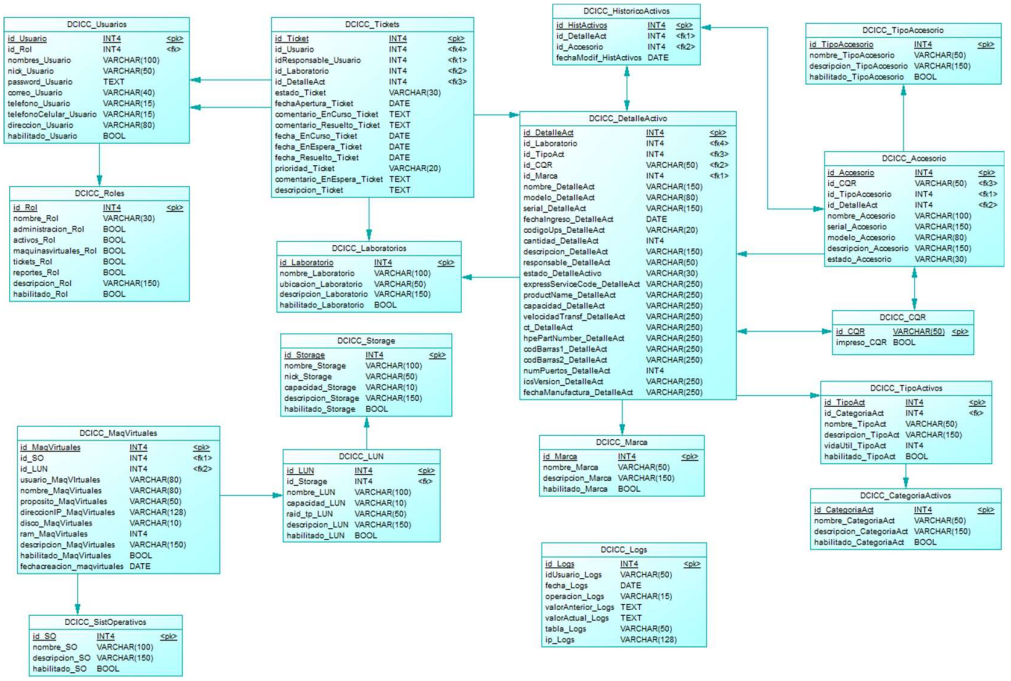


Figura 28. Diagrama físico de inventarios y ticketing.
Elaborado por: Andres Chisaguano, Joel Ludeña.

3.2. Desarrollo e implementación

3.2.1. Diagrama de despliegue

A continuación, se presenta el diagrama de despliegue que permitirá conocer la arquitectura del sistema, así como la interacción de sus componentes con el usuario.

Diagrama de despliegue

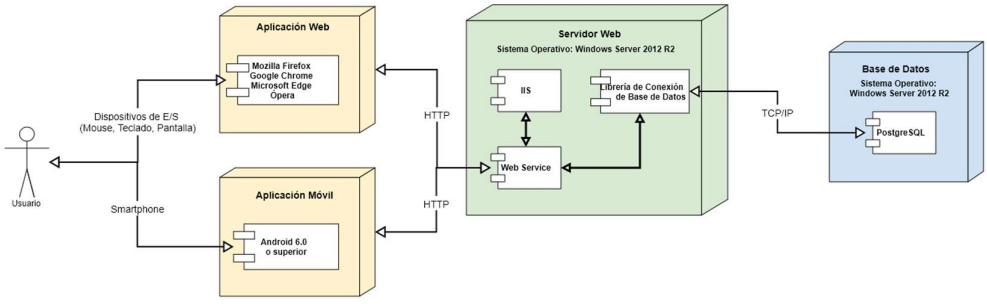


Figura 29. Diagrama de despliegue del sistema de inventarios y ticketing.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.2. Diagrama de navegación

A continuación, se presenta el diagrama de navegación el cual permitirá conocer la estructura de la aplicación Web.

Diagrama de navegación

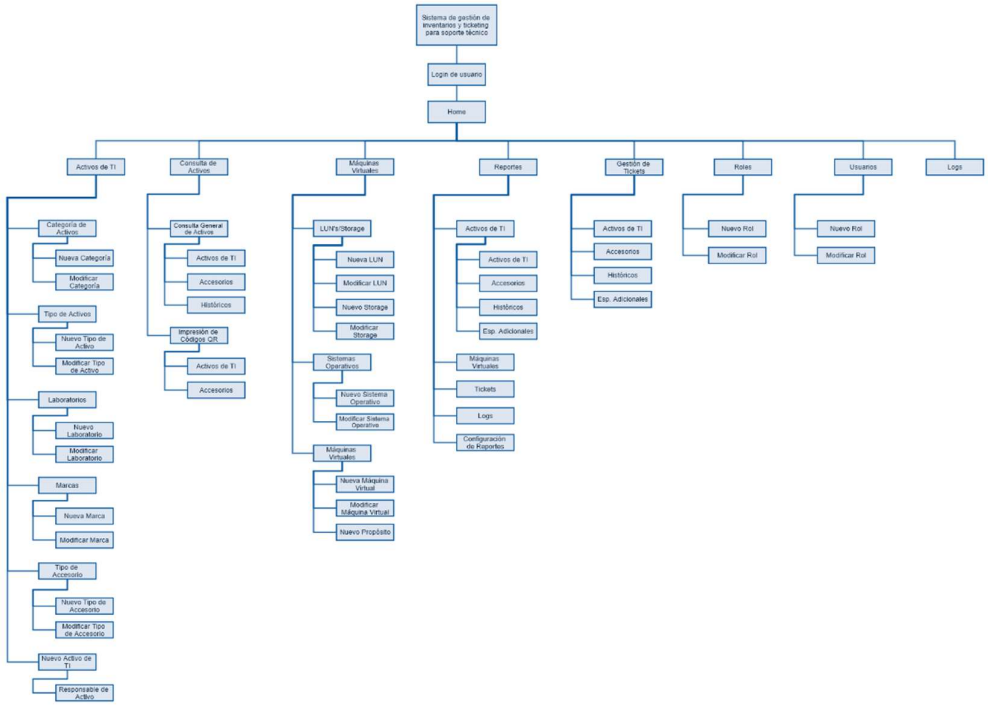


Figura 30. Diagrama de navegación de la aplicación web.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3. Código importante

3.2.3.1. Aplicación Web

En esta sección se describe el código fuente para las principales funcionalidades que cuenta la aplicación web, como son: comunicación con el Web Service, consulta, ingreso y actualización de activos de TI, y la generación de códigos QR.

3.2.3.1.1. Comunicación con el Web Service

Antes de hacer uso de alguno de los métodos del Web Service, es necesario solicitar un token de acceso. Para que un token pueda ser generado por el servicio, se debe enviar un Nick de usuario que se encuentre habilitado en la base de datos. Si cumple

con esta condición, será enviado un JSON con una cadena que contiene el token, caso contrario se envía un error 401 (No autorizado). El token será enviado en la cabecera "Header" de tipo "Authorization" en cada una de las peticiones realizadas en el servicio. Este token tiene un límite de tiempo de uso o caducidad que ha sido definido en 10 minutos.

En la Figura No. 31 se presentan las sentencias necesarias para iniciar la comunicación con el Web Service los mismos que se encuentran en la clase "ComunicacionServicio.cs", en donde en su constructor se define la URL de ubicación del Web Service y el tipo de dato que va a ser enviado.

Definición de parámetros para comunicación con el Web Service

```
public ComunicacionServicio(){
        base_URL = ConfigurationManager.AppSettings["URLWebServiceInventarios"];
        client_Service.DefaultRequestHeaders.Clear();
        client_Service.BaseAddress = new Uri(base_URL);
        client_Service.DefaultRequestHeaders.Accept.Add(new
        MediaTypeWithQualityHeaderValue("application/json"));
}
```

Figura 31. Configuración para comunicación con el Web Service.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 32 se presenta el método "ObtenerTokenTransacciones", el cual tiene como parámetro de ingreso una cadena que representa al Nick de la sesión actual del sistema. Luego se define el tipo de método HTTP, en este caso es POST, seguido del método que se encarga de generar el token desde el Web Service, el cual es "Token/ObtenerTokenTransacciones". Si el código de estado es "True", se serializa el JSON enviado a una cadena "String" y finalmente se retorna el token de autenticación seguido de la cadena "Bearer".

Obtención del token de transacciones

```
public string ObtenerTokenTransacciones(string NickUsuarioSesion){
       string tokenResult = string.Empty;
       var response =
       client_Service.PostAsJsonAsync("Token/ObtenerTokenTransacciones",
       NickUsuarioSesion).Result;
       if (response.IsSuccessStatusCode){
              var tokenJson = response.Content.ReadAsStringAsync().Result;
              tokenResult = JsonConvert.DeserializeObject<string>(tokenJson);
```

```
        }
        return string.Format("Bearer {0}",tokenResult);
}
```

Figura 32. Método para obtener el token de transacciones desde el web service.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.1.2. Inserción de un activo

Debido a que todo el proyecto está basado en una programación orientada a objetos (POO), cada módulo cuenta con una entidad y atributos correspondientes. En este caso, para los activos de TI se ha definido la entidad "Activos", la cual se muestra en la Figura No. 33 con algunos de sus atributos más importantes.

Clase que representa la entidad Activos

```
public class Activos{
        public int IdActivo { get; set; }
        public int IdMarca { get; set; }
        public string ResponsableActivo { get; set; }
        public string NombreActivo { get; set; }
        public string EstadoActivo { get; set; }
        public string ModeloActivo { get; set; }
        public string SerialActivo { get; set; }
        public DateTime FechaIngresoActivo { get; set; }
        ...
}
```

Figura 33. Entidad para activos de TI.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Para el envío y recepción de datos con el Web Service, se ha implementado una clase para cada entidad llamada "Mensajes<Entidad>", por ejemplo, para activos de TI la clase se llama "MensajesActivos.cs". Esta clase cuenta con cuatro parámetros que son: "OperacionExitosa", devuelve un valor Booleano indicando si la operación en la base de datos ya sea consulta, inserción o actualización ha sido o no realizada exitosamente; "MensajeError", es una cadena que indica el error proveniente de la base de datos; "ObjetoInventarios", es un objeto del tipo que representa la clase, por ejemplo, para la clase "MensajesActivos.cs" es un objeto tipo "Activos", el cual es utilizado únicamente en las consultas de un solo registro; finalmente "ListaObjetoInventarios", el cual es una lista tipo objeto, de igual manera el objeto se define dependiendo de la clase.

Clase para envío y recepción de datos para activos de TI

```
public class MensajesActivos{
        public bool OperacionExitosa { get; set; }
        public string MensajeError { get; set; }
        public Activos ObjetoInventarios { get; set; }
        public List<Activos> ListaObjetoInventarios { get; set; }
}
```

Figura 34. Clase de MensajesActivos.cs.
Elaborador por: Andres Chisaguano, Joel Ludeña.

A continuación, en la Figura No. 35 se presenta la clase "ActivosAccDatos.cs", la cual en su constructor se presenta la misma configuración inicial para comunicarse con el Web Service presentada en la Figura No. 31, pero en este caso recibe como parámetro de ingreso el Nick de usuario y se añade la sentencia en donde se añade el token de autenticación, para lo cual se utiliza el método de la Figura No. 32. Posteriormente este token es añadido a un "Header" de tipo "Authorization".

Definición de parámetros para comunicación con el Web Service en Activos.

```
public ActivosAccDatos(string NickUsuario_Sesion){
         ComunicacionServicio objComunicacionServicio = new
         ComunicacionServicio();
         client_Service.DefaultRequestHeaders.Clear();
         client_Service.BaseAddress = new Uri(ComunicacionServicio.base_URL);
         client_Service.DefaultRequestHeaders.Accept.Add(new
         MediaTypeWithQualityHeaderValue("application/json"));
         client_Service.DefaultRequestHeaders.Add("Authorization",
         objComunicacionServicio.ObtenerTokenTransacciones(NickUsuario_Sesion));
}
```

Figura 35. Configuración para comunicación con el Web Service en activos.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 35 se presenta el método que se encarga de enviar los datos de un activo para ser registrado en la base de datos. Se recibe como parámetro de ingreso un objeto tipo "Activos", luego se define el tipo de método HTTP seguido del método para ingresar un activo en la base de datos, el cual es "Activos/RegistrarActivo". Junto a la definición del método se coloca el parámetro de ingreso, en este caso el objeto tipo "Activos" y finalmente se retorna un objeto tipo "MensajesActivos".

Método para registrar un activo en la base de datos

```
public MensajesActivos RegistrarActivo(Activos infoActivo){
        MensajesActivos msjActivos = new MensajesActivos();
        var response = client_Service.PostAsJsonAsync("Activos/RegistrarActivo",
        infoActivo).Result;
        if (response.IsSuccessStatusCode){
                var ActivosJson = response.Content.ReadAsStringAsync().Result;
                msjActivos =
                JsonConvert.DeserializeObject<MensajesActivos>(ActivosJson);
        }
        return msjActivos;
}
```

Figura 36. Método para registro de activos en aplicación web.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 37 se presenta el método "NuevoActivo" del controlador "ActivosController.cs", el cual se encarga de recibir los datos del activo de TI desde la vista y envía los datos al método "RegistrarActivo" mostrada en la Figura No. 36. Básicamente, en este método se instancia a la clase "ActivosAccDatos.cs", en el cual se define el Nick de la sesión actual, luego se llama al método "RegistrarActivo" y se verificar si el atributo "OperacionExitosa" es "True".

Método del controlador para registrar un nuevo activo de TI

```
[HttpPost]
public ActionResult NuevoActivo(Activos infoActivo){
        string mensajesActivos = string.Empty;
        MensajesActivos msjActivos = new MensajesActivos();
        ActivosAccDatos objActivosAccDatos = new
        ActivosAccDatos((string)Session["NickUsuario"]);
        msjActivos = objActivosAccDatos.RegistrarActivo(infoActivo);
        if (msjActivos.OperacionExitosa){
                mensajesActivos = string.Format("El activo \"{0}\" ha sido
                registrado exitosamente.",infoActivo.NombreActivo);
                TempData["Mensaje"] = mensajesActivos;
        }else{
                mensajesActivos = string.Format("No se ha podido registrar el
                activo \"{0}\":
                {1}",infoActivo.NombreActivo,msjActivos.MensajeError);
                TempData["MensajeError"] = mensajesActivos;
        }
        return Json(msjActivos, JsonRequestBehavior.AllowGet);
}
```

Figura 37. Método de registro de activos en controlador.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.1.3. Actualización de activos

El método para el envío de datos para la modificación de un activo en la base de datos es similar al de registro de un activo de la Figura No. 38, ya que de igual

manera se envía un objeto tipo “Activos”. La diferencia es que este método recibe una variable Booleana para determinar si se actualiza todo el registro o solo su estado, en este caso si es “True” se actualizará solo el estado, caso contrario el resto de los atributos.

Método para actualizar un activo en la base de datos

```
public MensajesActivos ActualizarActivo(Activos infoActivo,bool actEstado){
        MensajesActivos msjActivos = new MensajesActivos();
        var response = client_Service.PostAsJsonAsync(actEstado ?
        "Activos/ActualizarEstadoActivo" : "Activos/ActualizarActivo",
        infoActivo).Result;
        if (response.IsSuccessStatusCode){
           var ActivosJson = response.Content.ReadAsStringAsync().Result;
           msjActivos =
           JsonConvert.DeserializeObject<MensajesActivos>(ActivosJson);
        }
        return msjActivos;
}
```

Figura 38. Método de actualización de activos en aplicación web.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 39 se presenta el método “ModificarActivo” del controlador “ActivosController.cs”, el cual se encarga de recibir los datos del activo de TI desde la vista y envía los datos al método “ActualizarActivo” mostrada en la Figura No. 38. Básicamente, en este método se instancia a la clase “ActivosAccDatos.cs”, en el cual se define el Nick de la sesión actual, luego se llama al método “ActualizarActivo” y se comprueba si el atributo “OperacionExitosa” es “True”, de ser este el caso, se define un mensaje exitoso, caso contrario se define un mensaje erróneo. Finalmente, se retorna en formato JSON el objeto “MensajesActivos” hacia la vista.

Método del controlador para actualizar un activo de TI

```
[HttpPost]
public ActionResult ModificarActivo(Activos infoActivo){
          string mensajesActivos = string.Empty;
          MensajesActivos msjActivos = new MensajesActivos();
          MensajesHistoricoActivos msjHistActivos = new
          MensajesHistoricoActivos();
          ActivosAccDatos objActivosAccDatos = new
          ActivosAccDatos((string)Session["NickUsuario"]);
          msjActivos = objActivosAccDatos.ActualizarActivo(infoActivo,false);
          if (msjActivos.OperacionExitosa)
          {
             mensajesActivos = string.Format("El activo con ID: {0} ha
             sido modificado correctamente.", infoActivo.IdActivo);
          }
```

```
            else
            {
                mensajesActivos = string.Format("No se ha podido actualizar el
                activo con ID: {0}:
                {1}",infoActivo.IdActivo,msjActivos.MensajeError);
            }
            return Json(msjActivos, JsonRequestBehavior.AllowGet);

}
```
Figura 39. Método de actualización de activos en controlador.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.1.4. Consulta de activos

En la Figura No. 40 se presenta el método que se encarga de obtener los activos registrados en la base de datos. El mismo recibe como parámetro de ingreso una cadena que define el tipo de función a llamar, este parámetro ha sido definido en tres opciones: “Comp”, para obtener todos los registros, “Hab”, para obtener solo los registros habilitados y “Nombres”, para obtener solo los nombres de los registros. Luego se define el tipo de método HTTP seguido del método para consultar los activos, el cual es “Activos/ObtenerActivos” seguido del tipo de función.

Método para consultar los activos en la base de datos

```
public MensajesActivos ObtenerActivos(string nombreFuncion){
            MensajesActivos msjActivos = new MensajesActivos();
            HttpResponseMessage response =
            client_Service.GetAsync(string.Format("Activos/ObtenerActivos{0}",nombr
            eFuncion)).Result;
            if (response.IsSuccessStatusCode){
                    var ActivosJson = response.Content.ReadAsStringAsync().Result;
                    msjActivos =
                    JsonConvert.DeserializeObject<MensajesActivos>(ActivosJson);
            }
            return msjActivos;
}
```
Figura 40. Método de consulta de activos en aplicación web.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 41 se define el método “ObtenerActivosComp” en el controlador “ActivosController.cs”, en el cual únicamente se hace la instancia a la clase “ActivosAccDatos.cs” para hacer uso del método de consulta previamente descrito y finalmente se retorna ese objeto en formato JSON hacia la vista.

Método del controlador para actualizar un activo de TI

```
public JsonResult ObtenerActivosComp(){
ActivosAccDatos objActivosAccDatos = new
```

```
        ActivosAccDatos((string)Session["NickUsuario"]);
        return Json(objActivosAccDatos.ObtenerActivos("Comp"),
        JsonRequestBehavior.AllowGet);
}
```

Figura 41. Método de consulta de activos en controlador.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.1.5. Generación de códigos QR

En la Figura No. 42 se presenta el método "GenerarCodigoQR", el cual se encarga de generar la imagen de un código QR en formato "Bitmap" a partir de una cadena "String". Para la generación de códigos QR se utilizó la librería "QRCoder", la cual se puede obtener desde Nuget.

Para obtener el código QR, se hace uso del método "CreateQrCode", que recibe como parámetros la cadena a codificar, que en este caso es el ID del código QR. Luego se debe definir el nivel de corrección de errores, el cual indica qué porcentaje del código QR puede corromperse antes de que el código ya no sea legible, estos pueden ser: sea L (7%), M (15%), Q (25%) o H (30%), en este caso se ha definido Q. Posteriormente, se establece el tamaño del código QR o número de pixeles por módulo, en este caso se ha definido en 6. Finalmente se retorna la imagen en formato Bitmap.

Método para generar un código QR

```
public Bitmap GenerarCodigoQR(string idCqr){
         QRCodeGenerator qrGenerator = new QRCodeGenerator();
         QRCodeData qrCodeData = qrGenerator.CreateQrCode(idCqr,
         QRCodeGenerator.ECCLevel.Q);
         QRCode qrCode = new QRCode(qrCodeData);
         Bitmap qrCodeImage = qrCode.GetGraphic(6);
         return qrCodeImage;
}
```

Figura 42. Método de generación de códigos QR.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.2. Web Service

En esta sección se describe el código fuente para las principales funcionalidades que cuenta el Web Service, como son: inserción, modificación y consulta de registros para la entidad "Marcas", y métodos HTTP.

3.2.3.2.1. Inserción de registros

Para las operaciones con la base de datos fue necesario el uso de la librería "Npgsql", la cual está optimizada para trabajar con PostgreSQL.

Clase que representa la entidad marcas

```
public class Marcas{
        public int IdMarca { get; set; }
        public string NombreMarca { get; set; }
        public string DescripcionMarca { get; set; }
        public bool HabilitadoMarca { get; set; }
}
```

Figura 43. Configuración inicial para comunicación con el Web Service.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Mientras que la clase utilizada para enviar los datos respectivos sobre una "Marca" se muestra en la siguiente Figura No. 44.

Clase para envío de datos para marcas

```
public class MensajesMarcas{
        public bool OperacionExitosa { get; set; }
        public string MensajeError { get; set; }
        public Marcas ObjetoInventarios { get; set; }
        public List<Marcas> ListaObjetoInventarios { get; set; }
}
```

Figura 44. Entidad de marcas.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 45 se presenta el método "RegistroMarca", el cual recibe un objeto tipo "Marca" y luego se define la sentencia propia de una inserción en PostgreSQL. Como se puede observar la cadena no se ve alterada, ya que se está usando parámetros de ingreso como son: "@nm", "@dm" y "@hm", las que pertenecen al "NombreMarca", "DescripcionMarca" y "HabilitadoMarca" respectivamente. Posteriormente se asigna el valor y el tipo de dato a cada parámetro. Si la operación se realizó sin problemas, el atributo "OperacionExitosa" será "True", caso contrario será "False".

Método de registro de marcas en la base de datos

```
public MensajesMarcas RegistroMarca(Marcas infoMarca){
           MensajesMarcas msjMarcas = new MensajesMarcas();
           try{
              NpgsqlTransaction tran = conn_BD.BeginTransaction();
                  using (NpgsqlCommand cmd = new NpgsqlCommand("insert into
                  dcicc_marca (nombre_marca,descripcion_marca,habilitado_marca)
```

```
                    VALUES (@nm,@dm,@hm)", conn_BD)){
                           cmd.Parameters.Add("nm",
                           NpgsqlTypes.NpgsqlDbType.Varchar).Value =
                           infoMarca.NombreMarca.Trim();
                           cmd.Parameters.Add("dm",
                           NpgsqlTypes.NpgsqlDbType.Varchar).Value =
                           infoMarca.DescripcionMarca;
                           cmd.Parameters.Add("hm",
                           NpgsqlTypes.NpgsqlDbType.Boolean).Value =
                           infoMarca.HabilitadoMarca;
                         cmd.ExecuteNonQuery();
                      }
                      tran.Commit();
                      conn_BD.Close();
                      msjMarcas.OperacionExitosa = true;
           }catch (Exception e){
                 msjMarcas.OperacionExitosa = false;
                 msjMarcas.MensajeError = e.Message;
           }
           return msjMarcas;
}
```

Figura 45. Método de inserción de marcas en el web service.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.2.2. Actualización de registros

El código de actualización de una marca es similar al de un registro de la Figura No. 46, la diferencia se encuentra en la sentencia SQL, además de que ahora se necesita de un ID para hacer posible la actualización.

Método para actualizar una marca en la base de datos

```
using (NpgsqlCommand cmd = new NpgsqlCommand("UPDATE dcicc_marca set nombre_marca
= @nm,descripcion_marca=@dm,habilitado_marca = @hm where id_marca = @im",
conn_BD)){
         cmd.Parameters.Add("nm", NpgsqlTypes.NpgsqlDbType.Varchar).Value =
         infoMarca.NombreMarca.Trim();
         cmd.Parameters.Add("dm", NpgsqlTypes.NpgsqlDbType.Varchar).Value =
         !string.IsNu infoMarca.DescripcionMarca;
         cmd.Parameters.Add("hm", NpgsqlTypes.NpgsqlDbType.Boolean).Value =
         infoMarca.HabilitadoMarca;
         cmd.Parameters.Add("im", NpgsqlTypes.NpgsqlDbType.Integer).Value =
         infoMarca.IdMarca;
         cmd.ExecuteNonQuery();
}
```

Figura 46. Método de actualización de marcas en el web service.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.2.3. Consulta de registros

La Figura No. 47 muestra el método “ObtenerMarcas”, el cual recibe como parámetro de ingreso una cadena que define el nombre de la función de la base de datos en la que se encuentra la consulta.

Una vez que se hace uso de la función, se llena los datos en un objeto de tipo "Marcas" en el orden en que se encuentra la consulta en SQL, para que luego cada objeto sea insertado en una lista.

Método para consultar las marcas en la base de datos

```
...
using (NpgsqlCommand cmd = new NpgsqlCommand(nombreFuncion, conn_BD)){
cmd.CommandType = CommandType.StoredProcedure;
        using (NpgsqlDataReader dr = cmd.ExecuteReader()){
        while (dr.Read()){
              Marcas objMarcas = new Marcas{
                      IdMarca = (int)dr[0],
                      NombreMarca = dr[1].ToString().Trim(),
                      DescripcionMarca = dr[2].ToString().Trim(),
                      HabilitadoMarca = (bool)dr[3]
               };
               lstMarcas.Add(objMarcas);
               }
        msjMarcas.ListaObjetoInventarios = lstMarcas;
        msjMarcas.OperacionExitosa = true;
        }
 }
 ...
```

Figura 47. Método de consulta de marcas en el web service.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.2.4. Métodos HTTP del Web Service

Para cada una de las entidades sobre las cuales se harán operaciones en la base de datos, ha sido definido un controlador. Por ejemplo, para el presente caso se tiene un controlador llamado "MarcasController.cs". Además, sobre cada controlador se define la ruta base sobre la cual serán utilizados sus métodos, en este caso se ha definido "Marcas". Por ejemplo, para consultar las marcas se debe colocar primero la ruta base "Marcas" y luego el nombre del método, "ObtenerMarcas".

Definición de ruta base y autorización

```
[Authorize(Policy = "Member")]
[Route("Marcas")]
public class MarcasContMarcaler : Controller
```

Figura 48. Configuración del controlador.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Referente a los métodos de inserciones o actualizaciones el tipo de método HTTP definido es POST, mientras que para las consultas es GET.

Para la inserción de una marca, en el método del Web Service se debe definir en primer lugar el tipo de método HTTP, además se define el nombre de la ruta con la cual se accederá al método, en este caso “RegistrarMarca”. Luego se especifican los parámetros de ingreso, en este caso un objeto tipo “Marcas”.

Método para registrar una marca en el Web Service

```
[HttpPost("RegistrarMarca")]
public MensajesMarcas RegistrarMarca([FromBody] Marcas infoMarca){
        MensajesMarcas msjMarcas = new MensajesMarcas();
        InsercionesMarcas objInsercionesMarcasBD = new InsercionesMarcas();
        msjMarcas = objInsercionesMarcasBD.RegistroMarca(infoMarca);
        return msjMarcas;
}
```

Figura 49. Método HTTP para registro de marcas.
Elaborador por: Andres Chisaguano, Joel Ludeña.

La Figura No. 50 muestra el método para actualizar una marca, en donde se llama al método “ActualizacionMarca” de la Figura No. 46 y se retorna un objeto tipo “MensajesMarcas”.

Método para actualizar una marca en el Web Service

```
[HttpPost("ActualizarMarca")]
public MensajesMarcas ActualizarMarca([FromBody] Marcas infoMarca){
        MensajesMarcas msjMarcas = new MensajesMarcas();
        ActualizacionesMarcas objActualizacionesMarcasBD = new
        ActualizacionesMarcas();
        msjMarcas = objActualizacionesMarcasBD.ActualizacionMarca(infoMarca);
        return msjMarcas;
}
```

Figura 50. Método HTTP para actualización de marcas.
Elaborador por: Andres Chisaguano, Joel Ludeña.

La Figura No. 51 muestra el método para consultar las marcas, en donde se llama al método “ObtenerMarcas” de la Figura No. 47, se define la función “consultamarca” de la base de datos y finalmente se retorna un objeto tipo “MensajesMarcas”.

Método para consultar marcas en el Web Service

```
[HttpGet("ObtenerMarcasComp")]
public MensajesMarcas ObtenerMarcasComp(){
        MensajesMarcas msjMarcas = new MensajesMarcas();
        ConsultasMarcas objConsultasMarcasBD = new ConsultasMarcas();
        msjMarcas = objConsultasMarcasBD.ObtenerMarcas("consultamarca");
        return msjMarcas;
}
```

Figura 51. Método HTTP para consulta de marcas.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.3. Aplicación móvil

En esta sección se puntualiza el código fuente de configuración y de funcionalidad que se utilizó para la aplicación móvil.

3.2.3.3.1. Archivos de configuración

Para el correcto funcionamiento de una aplicación móvil, es importante realizar ciertas configuraciones iniciales las cuales permiten acceder al hardware del dispositivo, incluir librerías externas o aprovechar las características del sistema operativo Android.

3.2.3.3.1.1. AndroidManifest.xml

Es el archivo de configuración inicial de un proyecto Android. En la Figura No. 52 se puede apreciar que se incluyen: permisos para la utilización de la cámara y servicios de red del dispositivo móvil, icono, nombre del proyecto, tema, actividad principal, dependencias externas y actividades secundarias.

Permisos para acceder al hardware del dispositivo móvil

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="ec.edu.ups.gestioninventarios">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.CAMERA" />
```

Nombre, icono, tema y llamada de la actividad principal

```
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_icono_dcicc"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".LoginActivity"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
```

Dependencia de datos externos

```
        <meta-data
            android:name="com.google.android.gms.vision.DEPENDENCIES"
            android:value="barcode" />
```

Llamada de las actividades secundarias

```
        <activity
            android:name=".Vistas.Activos.ActivosPrincipal"
            android:label="@string/title_activity_activos_principal"
            android:theme="@style/AppTheme.NoActionBar" />
```

Figura 52.Configuración inicial de la aplicación móvil
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.3.1.2. build.gradle

Este archivo permite definir configuraciones personalizadas para administrar el proceso de complicación y las librerías necesarias para aplicación móvil. En la Figura No. 53 se puede apreciar las siguientes configuraciones: versión de compilación, mínima versión de compilación (Android 6.0), ruta de recursos, repositorios y dependencias de compilación para librerías propias y externas.

Versiones de compilación y rutas de recursos

```
android {
    compileSdkVersion 26
    buildToolsVersion "25.0.2"
    defaultConfig {
        applicationId "ec.edu.ups.gestioninventarios"
        minSdkVersion 23
        targetSdkVersion 26
        versionName "1.0"
        multiDexEnabled true
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'),
'proguard-rules.pro'
        }
    }
    sourceSets { main { res.srcDirs = ['src/main/res', 'src/main/res/2'] } }
}
```

Repositorios para librerías externas

```
repositories{
    jcenter()
    maven {url "https://jitpack.io"}
}
```

Dependencias de compilación para librerías de Android y externas

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:26.0.0-alpha1'
    compile 'com.android.support:design:26.0.0-alpha1'
    testCompile 'junit:junit:4.12'
    compile 'com.google.android.gms:play-services-vision:9.8.0'
    compile 'com.github.PhilJay:MPAndroidChart:v3.0.3'
```

```
}
```

Figura 53.Configuraciones para la compilación de la aplicación móvil.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.3.2. Login de Usuario

Para que el usuario pueda acceder a los módulos de la aplicación, es necesario que las credenciales provistas estén registradas en el sistema, para lo cual se hace uso de los siguientes métodos de comunicación con el Web Service:

3.2.3.3.2.1. Verificación de usuario

Una vez que el usuario ha ingresado el Nick y contraseña, se procede a enviar dicha información al Web Service para realizar la verificación. Se hace uso del atributo "operacionExitosa" y los datos personales del usuario del objeto enviado por el Web Service. En la Figura No. 54 se aprecia la variable "conexión" y el método "loginInicio()" que son los que se encargan de realizar dicha función.

AutenticarUsuario.java
Métodos de conexión con el Web Service

```
public static final String
conexion="http://172.17.42.129/DCICC.WebServiceInventarios/";

public MensajesUsuarios loginInicio(String nick,String password){
    String METODO_WS="AccesoServicio/AutenticarUsuario";
    StrictMode.setThreadPolicy(new StrictMode.ThreadPolicy.Builder()
            .permitNetwork().build());
    URL url=null;
    HttpURLConnection conn=null;
    String json;

    MensajesUsuarios mensajesUsuarios = new MensajesUsuarios();
    try {
        url= new URL(conexion+METODO_WS);
        conn=(HttpURLConnection)url.openConnection();
        conn.setRequestMethod("POST");
        conn.setRequestProperty("Content-Type","application/json;charset=UTF-8");
        conn.setDoOutput(true);
        conn.setDoInput(true);
        conn.setConnectTimeout(10000);
        conn.setReadTimeout(20000);
        conn.connect();
         ...
```

Figura 54. Métodos para la conexión de la aplicación con el Web Service.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 55 se puede observar los métodos que se utilizan para el envío de información hacia el Web Service.

AutenticarUsuario.java
Envió de datos del usuario hacia el Web Service

```
JSONObject jsonObjectEnvio=new JSONObject();
jsonObjectEnvio.put("NickUsuario",nick);
jsonObjectEnvio.put("PasswordUsuario",password);
DataOutputStream os=new DataOutputStream(conn.getOutputStream());
BufferedWriter writer=
                    new BufferedWriter(new OutputStreamWriter(os,"UTF-8"));
writer.write(jsonObjectEnvio.toString());
...
```

Figura 55. Métodos para el envio de información al Web Service.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 56 se puede observar los métodos que se utilizan para la recepción de información desde el Web Service. Para hacer uso de la información, se debe almacenar en un objeto tipo "JSONObject".

AutenticarUsuario.java
Recepción de datos de usuario del Web Service

```
BufferedReader in= new BufferedReader(
                            new InputStreamReader(conn.getInputStream()));
String inputline;
StringBuffer response=new StringBuffer();
while((inputline=in.readLine())!=null){
       response.append(inputline);
}
json=response.toString();
JSONObject jsonObjectRetorno = null;
jsonObjectRetorno=new JSONObject(json);
...
```

Figura 56. Métodos para obtener información del Web Service.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Para almacenar la información del objeto enviado por el Web Service, se hace uso de clases que contienen las entidades correspondientes. En la Figura No. 57 se puede visualizar como se realiza la instancia de las clases.

AutenticarUsuario.java
Métodos para el llenado de las entidades

```
mensajesUsuarios.setOperacionExitosa(Boolean.parseBoolean(jsonObjectRetorno.optSt
ring("operacionExitosa")));
Usuarios usuario=new Usuarios();
JSONObject jsonUsuario=new
JSONObject(jsonObjectRetorno.optString("objetoInventarios"));
usuario.setIdUsuario(Integer.parseInt(jsonUsuario.optString("idUsuario")));
...
```

Figura 57. Métodos para instanciar las clases de entidades.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Para cada transacción se instancian entidades las cuales permiten mantener la información a lo largo de la sesión de usuario. En la Figura No. 58 se detalla la clase "MensajesUsuarios.java" la cual contiene toda la información del objeto "Usuario" provista por el Web Service.

Entidad Mensajes Usuarios

```
public class MensajesUsuarios {
    public boolean operacionExitosa;
    public String mensajeError;
    public Usuarios objetoInventarios;
    public boolean isOperacionExitosa() {return operacionExitosa;}
    public void setOperacionExitosa(boolean operacionExitosa) {
        this.operacionExitosa = operacionExitosa;
    }
    ...
}
```

Figura 58. Instancia de la clase MensajesUsuarios.java
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.3.2.2. Métodos para asignación de módulos

Una vez que los datos del usuario han sido almacenados, se realiza la comprobación del rol al cual pertenece con los métodos de la Figura No. 59. En la aplicación existen dos módulos para lo cual si el usuario tiene un rol perteneciente a "Activos de TI" se dará paso al módulo de Gestión de Activos y si el usuario tiene un rol perteneciente a "ticketing" se dará paso al módulo de Ticketing para soporte técnico.

LoginActivity.java
Condiciones para asignar el módulo al usuario.

```
if(mensaje.getObjetoInventarios().getNombreRol().equals("administrador")){
       this.myIntent = new Intent(LoginActivity.this, ActivosPrincipal.class);
       myIntent.putExtra("NICK_USUARIO", nick);
       myIntent.putExtra("CORREO_USUARIO",mensaje.getObjetoInventarios(
       myIntent.putExtra("OPERACION","LOGIN");
       startActivity(myIntent);
}else if(mensaje.getObjetoInventarios().getNombreRol().equals("docente")){
    this.myIntent = new Intent(LoginActivity.this, TicketingPrincipal.class);
       myIntent.putExtra("NICK_USUARIO_TICK", nick);
       myIntent.putExtra("CORREO_USUARIO_TICK",mensaje.getObjetoInventarios().
       getCorreoUsuario());
       myIntent.putExtra("OPERACION_TICK","LOGIN");
       startActivity(myIntent);
}
...
```

Figura 59. Método para determinar un módulo dependiendo del rol.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.3.3. Obtención de Token para Transacciones

Para realizar una consulta, actualización o inserción en cada módulo, es necesario obtener un token de autenticación desde el Web Service. En la Figura No. 60 se detalla el método en el cual se envía el Nick de usuario y se recibe el token correspondiente, además, se define el parámetro de autorización para cada petición.

AutenticarUsuario.java
Envió del Nick de usuario para obtener el token

```
DataOutputStream os=new DataOutputStream(conn.getOutputStream());
BufferedWriter writer=new BufferedWriter(new OutputStreamWriter(os,"UTF-8"));
writer.write("\""+nick+"\"");
```

Token de autorización para las transacciones.

```
token="Bearer "+response.toString();
```

Método para la autorización de un método del Web Service.

```
conn.setRequestMethod("POST");
conn.setRequestProperty("Content-Type","application/json;charset=UTF-8");
conn.setRequestProperty("Authorization",token);
...
```

Figura 60. Métodos para la obtención del token de autorización.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.3.3.4. Scanner para códigos QR.

Para propósitos del presente proyecto fue necesario instanciar un método que permita escanear el código QR de un activo o accesorio con la finalidad de obtener su información. En la Figura No. 61 se describen las instancias para definir el espacio de visualización del escáner, la detección de un código QR y la apertura de la cámara con sus propiedades.

ActivosCQR.java
Instancias para el detector de códigos QR

```
surfaceView = (SurfaceView) findViewById(R.id.camaraActivos);
barcodeDetector = new BarcodeDetector.Builder(this)
        .setBarcodeFormats(Barcode.QR_CODE)
        .build();
cameraSource = new CameraSource
        .Builder(this, barcodeDetector)
        .setRequestedPreviewSize(2048, 1024).setAutoFocusEnabled(true).build();
        ...
```

Figura 61. Instancia de los componentes para el scanner de códigos QR.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 62 se describe el método para la detección. Si el objeto "cqr" es diferente de 0, indica que se detectó un código QR. Este valor se almacena en una variable tipo String denominada "palabra" para enviarla en el método "datosActivosCQT" y realizar la consulta de la información respectiva.

ActivosCQR.java
Método para obtener el valor de un código QR.

```
barcodeDetector.setProcessor(new Detector.Processor<Barcode>() {
    Intent myIntent;
    @Override
    public void release() {
    }
    @Override
    public void receiveDetections(Detector.Detections<Barcode> detections) {
        SparseArray<Barcode> cqr=detections.getDetectedItems();
        if (cqr.size() !=0){
            String palabra=cqr.valueAt(0).displayValue;
                  MensajesActivos mensaje=
                  activos.datosActivosCQR(palabra,nick);
        }
    }
});
```

Figura 62. Método de detección de códigos QR.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.4. Implementación

Una vez concluida la etapa de desarrollo, se procedió a la implementación del sistema en una máquina virtual alojada en el servidor del Data Center de la Carrera de Ingeniería en Ciencias de la Computación. Para ello se describe a continuación los elementos que se utilizaron para desplegar el sistema en un ambiente de producción.

3.2.4.1. Máquina Virtual

Tabla 49. Características de la máquina virtual para la implementación.

Característica	Descripción
Sistema Operativo	Windows Server R2 2012
Disco Duro	100 GB
Memoria RAM	8 GB
Dirección IP	172.17.42.97
Nombre Máquina Virtual	VM_TESIS_DCHISAGUANO_I

Nota: Características de la máquina virtual en donde se implementó el sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.2.4.2. Software

Tabla 50. Características de software para la implementación.

Nombre	Versión
Visual Studio Community 2017	15.8
PostgreSQL	9.5
Internet Information Services (IIS)	10.0

Nota: Características del software en donde se implementó el sistema.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.3. Pruebas

En esta sección se realiza la ejecución de las pruebas unitarias, pruebas de funcionalidad, pruebas de carga y pruebas de stress, tanto para la aplicación Web como para la aplicación Android, con la finalidad de comprobar el funcionamiento y el rendimiento del sistema.

3.3.1. Pruebas unitarias

Las pruebas unitarias que se realizaron al sistema están enfocados al registro, actualización y consulta de las principales entidades como son: activos de TI, máquinas virtuales, tickets y usuarios. Las pruebas unitarias fueron realizadas mediante la herramienta "JUnit" para la aplicación Android y "TestTools" para la aplicación Web. Para cada caso se presentan los atributos principales de cada uno de los objetos que son la entrada para la prueba unitaria. A continuación, se muestran los resultados de los métodos para la autenticación de usuarios y la obtención del token de transacciones.

Tabla 51. Resultados de pruebas unitarias para Login.

Método	Objeto	Atributos	Tiempo (ms)	Estado
ObtenerTokenTransacciones	-	NickUsuario	577	OK
AutenticarUsuario	Usuario	NickUsuario, Password	152	OK

Nota: Resultados de pruebas unitarias.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Los resultados de los métodos de la entidad Activos son los siguientes:

Tabla 52. Resultados de las pruebas unitarias para Activos.

Método	Objeto	Atributos	Tiempo (ms)	Estado
RegistrarActivo	Activo	IdActivo	838	OK

ActualizarActivo		IdCQR IdTipoActivo IdLaboratorio IdMarca NombreActivo ModeloActivo SerialActivo FechaIngresoActivo, ResponsableActivo EstadoActivo.	652	OK
ObtenerActivos	-	TipoFuncion	190	OK

Nota: Resultados de pruebas unitarias.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Los resultados de los métodos de la entidad Máquinas Virtuales son los siguientes:

Tabla 53. Resultados de las pruebas unitarias para Máquinas Virtuales.

Método	Objeto	Atributos	Tiem-po (ms)	Estado
RegistrarMaqVirtual	MaqVirtuales	IdMaqVirtuales IdSistOperativos IdLUN UsuarioMaqVirtuales NombreMaqVirtuales PropositoMaqVirtuales DireccionIPMaqVirtuales DiscoMaqVirtuales RamMaqVirtuales HabilitadoMaqVirtuales.	2,119	OK
ActualizarMaqVirtual			702	OK
ObtenerMaqVirtuales	-	TipoFuncion	978	OK

Nota: Resultados de pruebas unitarias.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Los resultados de los métodos de la entidad Tickets son los siguientes:

Tabla 54. Resultados de las pruebas unitarias para Tickets.

Método	Objeto	Atributos	Tiempo (s)	Estado
IngresoNuevoTicket	Tickets	IdTicket IdUsuario IdResponsableUsuario IdLaboratorio EstadoTicket FechaAperturaTicket DescripcionTicket PrioridadTicket	0,185	OK
ActualizarTicket			1,6	OK
ObtenerTickets	-	TipoFuncion	2,6	OK

Nota: Resultados de pruebas unitarias.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Finalmente, los resultados de los métodos de la entidad Usuarios son los siguientes:

Tabla 55. Resultados para las pruebas unitarias para Usuarios.

Método	Objeto	Atributos	Tiempo (s)	Estado
RegistrarUsuario	Usuarios	IdUsuario	2,9	OK
ActualizarUsuario		IdRol NombresUsuario PasswordUsuario NickUsuario CorreoUsuario TelefonoUsuario HabilitadoUsuario	2,5	OK
ObtenerUsuarios	-	TipoFuncion	1,4	OK

Nota: Resultados de pruebas unitarias.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.3.2. Pruebas de funcionalidad

Las pruebas de funcionalidad en el sistema se han realizado basadas en los requisitos funcionales previamente definidos por los involucrados del proyecto. Para comprobar el cumplimiento de los requisitos funcionales de ambas aplicaciones, se detalla en la Tabla No. 56 y Tabla No. 57, el ID del requisito funcional, la descripción del proceso, observaciones y el estado de cada funcionalidad.

3.3.2.1. Aplicación Web

Tabla 56. Comprobación de los requisitos funcionales de la aplicación Web.

IDR	Requisito	Proceso en Aplicación	Observaciones	Estado
RF01	Login de Usuarios	1. Dirigirse a la página de inicio de la aplicación en la siguiente URL: http:/172.17.42.129/Inventarios/ 2. Ingresar las credenciales del usuario. 3. Clic en el botón “Login”.	No se pueden ingresar caracteres especiales, la contraseña debe contener más de 6 caracteres.	OK
RF02	Ingreso de nuevos activos de TI	1. Autenticarse en el sistema. 2. Clic en el menú lateral “Activos de TI”, en la opción “Nuevo Activo de TI”. 3. Ingresar los campos	El nombre del activo de TI debe escribirse en mayúsculas, no se permiten espacios ni ciertos	OK

		obligatorios del nuevo activo. 4. Ingresar las especificaciones adicionales del nuevo activo en caso de existir. 5. Presionar el botón “Siguiente”. 6. Finalmente, si el usuario lo requiere, imprimir el código QR generado.	caracteres especiales y debe ser único. Los campos de especificaciones adicionales no son obligatorios.	
RF04	Consulta general de los activos de TI.	1. Autenticarse en el sistema. 2. Clic en el menú lateral “Consulta de Activos”, en la opción “Consulta General de Activos”. 3. Aplicar los filtros de búsqueda en caso de requerirlos. 4. Visualizar los registros.	Se puede navegar entre pestañas para visualizar tanto los activos de TI, como también accesorios, especificaciones adicionales o activos que han sido dados de baja.	OK
RF08	Reportería	1. Autenticarse en el sistema. 2. Clic en el menú lateral “Reportes”. 3. Seleccionar el reporte que se desea generar entre las opciones: Activos de TI, Máquinas Virtuales, Tickets o Logs. 4. Aplicar los filtros necesarios en caso de requerirlos. 5. Clic en el botón Reporte PDF o Reporte Excel.	En el caso de que no existan datos registrados en una de las opciones del reporte, no se podrá generar ninguno de los dos tipos de reporte.	OK
RF11	Gestión de tickets	1. Autenticarse en el sistema. 2. Clic en el menú lateral “Gestión de tickets”. 3. En la tabla “Listado de Tickets”, dirigirse a una de las secciones de los tickets: “Abiertos”, “En Proceso”, “En Espera” o “Resuelto”. 4. Dirigirse a un ticket en	Los tickets “Resueltos” solo podrán ser visualizados, más no podrán ser gestionados. Los tickets “En espera” solo podrán ser trasladados a	OK

		particular, clic en el botón celeste para gestionar un ticket. 5. Ingresar o seleccionar la información requerida. 6. Clic en el botón "Modificar", seguido del botón "Confirmar" para actualizar el ticket.	"Resueltos". Los tickets solo pueden ser asignados una vez a un responsable.	

Nota: Análisis de pruebas de funcionalidad sobre la aplicación Web.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.3.2.2. Aplicación Android

Tabla 57. Comprobación de los requisitos funcionales de la aplicación Android.

IDR	Requisito	Proceso en Aplicación	Observaciones	Estado
RF01	Login de Usuarios	1. Abrir la aplicación instalada 2. Ingresar las credenciales del usuario y presionar el botón "Login".	No se pueden ingresar caracteres especiales, la contraseña debe contener más de 6 caracteres.	OK
RF03	Consulta de activos por código QR.	1. Autenticarse en la aplicación. 2. Desplegar el menú lateral y seleccionar la opción "Consultar Activo/Accesorio". 3. Enfocar el código QR a la cámara del dispositivo. 4. Se muestran los datos del activo.	Solo usuarios con rol de administración de activos de TI pueden ingresar a esta función. En caso de que el código QR no se encuentre registrado en la base de datos, se despliega un mensaje y se retorna a la página de inicio.	OK
RF09	Registro de Ticket (Activo/Accesorio)	1. Autenticarse en la aplicación. 2. Desplegar el menú lateral y seleccionar la opción "Incidencia de Activos de TI". 3. Enfocar el código QR a la cámara del dispositivo. 4. Se muestran los datos del activo.	Solamente los usuarios con rol docente o generador de tickets podrán registrar un ticket en la base de datos. El campo descripción de ticket es	OK

		5. Seleccionar la prioridad e ingresar la descripción. 6. Presionar el botón verde ubicado en la esquina superior derecha de la pantalla.	obligatorio.	
RF09	Registro de Ticket (General)	1. Autenticarse en la aplicación. 2. Desplegar el menú lateral y seleccionar la opción "Reportar Incidencia General". 3. Seleccionar el laboratorio, la prioridad e ingresar la descripción. 4. Presionar el botón verde ubicado en la esquina superior derecha de la pantalla.	El campo descripción de ticket es obligatorio. En caso de no existir un laboratorio, se mostrará un mensaje de notificación.	OK
RF10	Consulta de estado de ticket	1. Autenticarse en la aplicación. 2. Desplegar el menú lateral y seleccionar la opción "Consultar estado de tickets". 3. Se desplegará una pantalla con la información de los tickets reportados por el usuario.	Solamente se podrán visualizar los estados de los tickets generados por el usuario de la sesión actual.	OK

Nota: Análisis de pruebas de funcionalidad sobre la aplicación Android.
Elaborador por: Andres Chisaguano, Joel Ludeña.

3.3.3. Pruebas de carga y stress de la aplicación Web

Para llevar a cabo la ejecución de las pruebas se utilizó la herramienta "JMETER", en el cual se especifica el enlace URL sobre el cual se realiza la prueba, el tipo de petición, el número de peticiones, el periodo de tiempo, entre otros parámetros de entrada. Para la realización de las pruebas en los registros de códigos QR y activos de TI, fue necesario configurar un contador que fue agregado a campos con propiedad UNIQUE, con el fin de evitar errores de llave duplicada en la base de datos.

Para él envió de datos se definen entradas de tipo JSON dependiendo del objeto que se espera en el Web Service como por ejemplo, para el registro de un activo se envía un objeto tipo “Activos” ya descrita en la Figura No. 33 y para la recepción un objeto tipo “Mensajes<Entidad>”, como por ejemplo el que se muestra en la Figura No. 34.

3.3.3.1. Pruebas de carga

En el primer caso de cada prueba se realiza una evaluación con 30 peticiones concurrentes en 1 segundo sobre una determinada función, mientras que en el segundo caso se evalúan 50 peticiones concurrentes en 1 segundo. Esto con la finalidad de comprobar el rendimiento y comportamiento del sistema en escenarios o eventos que puedan producirse normalmente.

Caso 1

Tabla 58. Resultados del caso 1 de la prueba de carga de Login de usuarios.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
Login	HTTP POST	30	182	28	472	21,44	19,72

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 58 se puede observar que, en la autenticación con el sistema, se tiene una respuesta promedio de 0,182 segundos, en donde el tiempo máximo es de 0,472 segundos y el tiempo mínimo es de 0,028 segundos. En lo referente a los datos, fueron enviados un total de 21,44 KB por segundo y fueron recibidos 19,72 KB por segundo debido a la cantidad de información que contiene el JSON.

Tabla 59. Resultados del caso 1 de la prueba de carga de nuevo código QR.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoCQR	HTTP POST	30	176	19	520	18,95	8,89

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 59 se puede observar que, en el registro de un nuevo código QR, se tiene una respuesta promedio de 0,176 segundos, en donde el tiempo máximo es de 0,520 segundos y el tiempo mínimo es de 0,019 segundos. En lo referente a los datos, fueron enviados un total de 18,95 KB por segundo y fueron recibidos 8,89 KB por segundo debido a la cantidad de información que contiene el JSON.

Tabla 60. Resultados del caso 1 de la prueba de carga de nuevo activo.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoActivo	HTTP POST	30	1063	897	2046	10,66	14,10

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 60 se puede observar que, en el registro de un nuevo activo de TI, se tiene una respuesta promedio de 1,063 segundos, en donde el tiempo máximo es de 2,046 segundos y el tiempo mínimo es de 0,897 segundos. En lo referente a los datos, fueron enviados un total de 10,66 KB por segundo y fueron recibidos 14,1 KB por segundo debido a la cantidad de información que contiene el JSON.

Tabla 61. Resultados del caso 1 de la prueba de carga de nuevo ticket.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoTicket	HTTP POST	30	22	15	66	24,49	8,72

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 61 se puede observar que, en el registro de un nuevo ticket para soporte técnico, se tiene una respuesta promedio de 0,022 segundos, en donde el tiempo máximo es de 0,066 segundos y el tiempo mínimo es de 0,015 segundos. En lo referente a los datos, fueron enviados un total de 24,49 KB por segundo y fueron recibidos 8,72 KB por segundo debido a la cantidad de información que contiene el JSON.

Tabla 62. Resultados del caso 1 de la prueba de carga de consulta de activos.

Función	Tipo	Peticio-nes	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
ConsultaActivos	HTTP GET	30	220	40	483	16,90	1131,33

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 62 se puede observar que, en la consulta de activos de TI, se tiene una respuesta promedio de 0,22 segundos, en donde el tiempo máximo es de 0,483 segundos y el tiempo mínimo es de 0,04 segundos. En lo referente a los datos, fueron enviados un total de 16,90 KB por segundo y fueron recibidos 1131,33 KB por segundo debido a la cantidad de información que contiene el JSON.

Tabla 63. Resultados del caso 1 de la prueba de carga de consulta de tickets.

Función	Tipo	Peticio-nes	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
ConsultaTickets	HTTP GET	30	116	25	323	17,13	3892,80

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 63 se puede observar que, en la consulta de tickets, se tiene una respuesta promedio de 0,116 segundos, en donde el tiempo máximo es de 0,323 segundos y el tiempo mínimo es de 0,025 segundos. En lo referente a los datos, fueron enviados un total de 17,13 KB por segundo y fueron recibidos 3892,8 KB por segundo debido a la cantidad de información que contiene el JSON.

Caso 2

Tabla 64. Resultados del caso 2 de prueba de carga de Login de usuarios.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
Login	HTTP POST	50	32	25	102	32,14	29,56

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 50, en la Tabla No. 64 se tiene una respuesta promedio de 0,032 segundos, en donde el tiempo máximo es de 0,102 segundos y el

tiempo mínimo es de 0,025 segundos. En lo referente a los datos, fueron enviados un total de 32,14 KB por segundo y fueron recibidos 29,56 KB por segundo.

Tabla 65. Resultados del caso 2 de la prueba de carga de nuevo código QR.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoCQR	HTTP POST	50	592	187	1687	16,63	7,80

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 50, en la Tabla No. 65 se tiene una respuesta promedio de 0,592 segundos, en donde el tiempo máximo es de 1,687 segundos y el tiempo mínimo es de 0,187 segundos. En lo referente a los datos, fueron enviados un total de 16,63 KB por segundo y fueron recibidos 7,80 KB por segundo.

Tabla 66. Resultados del caso 2 de la prueba de carga de nuevo activo.

Función	Tipo	Peticio-nes	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoActivo	HTTP POST	50	44	31	107	38,85	51,45

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 50, en la Tabla No. 66 se tiene una respuesta promedio de 0,044 segundos, en donde el tiempo máximo es de 0,107 segundos y el tiempo mínimo es de 0,031 segundos. En lo referente a los datos, fueron enviados un total de 38,85 KB por segundo y fueron recibidos 51,45 KB por segundo.

Tabla 67. Resultados del caso 2 de la prueba de carga de nuevo ticket.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoTicket	HTTP POST	50	99	47	396	37,34	13,30

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 50, en la Tabla No. 67 se tiene una respuesta promedio de 0,099 segundos, en donde el tiempo máximo es de 0,396 segundos y el

tiempo mínimo es de 0,047 segundos. En lo referente a los datos, fueron enviados un total de 37,34 KB por segundo y fueron recibidos 13,3 KB por segundo.

Tabla 68. Resultados del caso 2 de la prueba de carga de consulta de activos.

Función	Tipo	Peticio-nes	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
ConsultaActivos	HTTP GET	50	118	45	908	26,69	1787,05

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 50, en la Tabla No. 68 se tiene una respuesta promedio de 118 segundos, en donde el tiempo máximo es de 0,908 segundos y el tiempo mínimo es de 0,045 segundos. En lo referente a los datos, fueron enviados un total de 26,69 KB por segundo y fueron recibidos 1,787 MB por segundo.

Tabla 69. Resultados del caso 2 de la prueba de carga de consulta de tickets.

Función	Tipo	Peticio-nes	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
ConsultaTickets	HTTP GET	50	505	55	1712	14,81	3364,89

Nota: Resultados de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 50, en la Tabla No. 69 se puede observar una respuesta promedio de 505 segundos, en donde el tiempo máximo es de 1,712 segundos y el tiempo mínimo es de 0,055 segundos. En lo referente a los datos, fueron enviados un total de 14,81 KB por segundo y fueron recibidos 3,364 MB por segundo.

Análisis de Resultados

Considerando la cantidad de información enviada y recibida en ambos casos, se puede decir que el sistema responde favorablemente a las peticiones, ya que en la mayoría de los casos el tiempo promedio no supera los 2 segundos en ejecutar la transacción, por lo que el cliente no percibiría ese tiempo durante la operación

realizada, además no se detectó ningún error durante la ejecución de las pruebas respectivas.

3.3.3.2. Pruebas de stress

Para la ejecución se las pruebas de stress se realizaron dos casos. En el primer caso de cada prueba se realiza una evaluación con 200 peticiones concurrentes en 1 segundo sobre una determinada función, mientras que en el segundo caso se evalúan 500 peticiones concurrentes en 1 segundo. Esto con la finalidad de comprobar el rendimiento y comportamiento del sistema en situaciones que puedan ser consideradas extremas, ya que, al tratarse de un sistema de gestión de inventarios, solo tendría acceso el personal autorizado, haciendo que dudablemente se produzca escenarios de este tipo.

Caso 1

Tabla 70. Resultados del caso 1 de la prueba de stress de Login de usuarios.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
Login	HTTP POST	200	2543	204	5022	23,24	21,38

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 70 se puede observar que, en la autenticación con el sistema, se tiene una respuesta promedio de 2,543 segundos, en donde el tiempo máximo es de 5,022 segundos y el tiempo mínimo es de 0,204 segundos. En lo referente a los datos, fueron enviados un total de 23,24 KB por segundo y fueron recibidos 21,38 KB por segundo debido a la cantidad de información que contiene el JSON.

Tabla 71. Resultados del caso 1 de la prueba de stress de nuevo código QR.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoCQR	HTTP POST	200	1581	162	3132	32,77	15,37

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 71 se puede observar que, en el registro de un nuevo código QR, se tiene una respuesta promedio de 1,581 segundos, en donde el tiempo máximo es de 3,132 segundos y el tiempo mínimo es de 0,162 segundos. En lo referente a los datos, fueron enviados un total de 32,77 KB por segundo y fueron recibidos 15,37 KB por segundo debido a la cantidad de información que contiene el JSON.

Tabla 72. Resultados del caso 1 de la prueba de stress de nuevo activo.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoActivo	HTTP POST	200	2115	142	4070	32,64	43,25

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 72 se puede observar que, en el registro de un nuevo código activo de TI, se tiene una respuesta promedio de 2,115 segundos, en donde el tiempo máximo es de 4,070 segundos y el tiempo mínimo es de 0,142 segundos. En lo referente a los datos, fueron enviados un total de 32,64 KB por segundo y fueron recibidos 43,25 KB por segundo debido a la cantidad de información que contiene el JSON.

Tabla 73. Resultados del caso 1 de la prueba de stress de nuevo ticket.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoTicket	HTTP POST	200	1486	151	3668	38,61	13,77

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 73 se puede observar que, en el registro de un nuevo ticket para soporte técnico, se tiene una respuesta promedio de 1,486 segundos, en donde el tiempo máximo es de 3,668 segundos y el tiempo mínimo es de 0,151 segundos. En lo referente a los datos, fueron enviados un total de 38,61 KB por segundo y fueron recibidos 13,77 KB por segundo debido a la cantidad de información que contiene el JSON.

Tabla 74. Resultados del caso 1 de la prueba de stress de consulta de activos.

Función	Tipo	Peticio-nes	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
ConsultaActivos	HTTP GET	200	8583	6731	11243	10,02	6685,80

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 74 se puede observar que, en la consulta de activos de TI, se tiene una respuesta promedio de 8,583 segundos, en donde el tiempo máximo es de 11,243 segundos y el tiempo mínimo es de 6,731 segundos. En lo referente a los datos, fueron enviados un total de 10,02 KB por segundo y fueron recibidos 6,685 KB por segundo. En este caso se puede notar un incremento tanto en el tiempo promedio de respuesta como en los bytes recibidos, ya que al realizar 50 peticiones se reciben aproximadamente 1,8 MB en un tiempo promedio de 0,5 segundos, mientras que con 200 peticiones se reciben 6,7 MB en un tiempo promedio de 8,6 segundos.

Tabla 75. Resultados del caso 1 de la prueba de stress de consulta de tickets.

Función	Tipo	Peticio-nes	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
ConsultaTickets	HTTP GET	200	2815	156	6149	18,56	20064,29

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 75 se puede observar que, en la consulta de activos de TI, se tiene una respuesta promedio de 2,815 segundos, en donde el tiempo máximo es de 6,149 segundos y el tiempo mínimo es de 0,156 segundos. En lo referente a los datos, fueron enviados un total de 18,56 KB por segundo y fueron recibidos 20,06 MB por segundo. En este caso se puede notar un incremento tanto en el tiempo promedio de respuesta como en los bytes recibidos, ya que al realizar 50 peticiones se reciben aproximadamente 3,4 MB en un tiempo promedio de 0,5 segundos, mientras que con 200 peticiones se reciben 20,06 MB en un tiempo promedio de 2,8 segundos.

Caso 2

Tabla 76. Resultados del caso 2 de la prueba de stress de Login de usuarios.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
Login	HTTP POST	500	6773	314	12860	23	21,16

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 500, en la Tabla No. 76 se tiene una respuesta promedio de 6,773 segundos, en donde el tiempo máximo es de 12,860 segundos y el tiempo mínimo es de 0,314 segundos. En lo referente a los datos, fueron enviados un total de 23 KB por segundo y fueron recibidos 21,16 KB por segundo. A comparación del primer caso, el tiempo promedio en realizar la comprobación de credenciales aumentó de 2,5 a 6,8 segundos.

Tabla 77. Resultados del caso 2 de la prueba de stress de nuevo código QR.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoCQR	HTTP POST	500	4597	343	8809	32,71	15,33

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 500, en la Tabla No. 77 se tiene una respuesta promedio de 4,597 segundos, en donde el tiempo máximo es de 8,809 segundos y el tiempo mínimo es de 0,343 segundos. En lo referente a los datos, fueron enviados un total de 32,71 KB por segundo y fueron recibidos 15,33 KB por segundo. A comparación del primer caso, el tiempo promedio en registrar un código QR aumentó considerablemente de 1,6 a 4,6 segundos.

Tabla 78. Resultados del caso 2 de la prueba de stress de nuevo activo.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoActivo	HTTP POST	500	4538	344	8507	41,77	55,35

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 500, en la Tabla No. 78 se tiene una respuesta promedio de 4,538 segundos, en donde el tiempo máximo es de 8,507 segundos y el tiempo mínimo es de 0,344 segundos. En lo referente a los datos, fueron enviados un total de 41,77 KB por segundo y fueron recibidos 55,35 KB por segundo. A comparación del primer caso, el tiempo promedio en registrar un activo de TI aumentó considerablemente de 2,1 a 8,8 segundos.

Tabla 79. Resultados del caso 2 de la prueba de stress de nuevo ticket.

Función	Tipo	Peticiones	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
NuevoTicket	HTTP POST	500	4408	222	8460	44,91	16,01

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 500, en la Tabla No. 79 se tiene una respuesta promedio de 4,408 segundos, en donde el tiempo máximo es de 8,460 segundos y el tiempo mínimo es de 0,222 segundos. En lo referente a los datos, fueron enviados un total de 44,91 KB por segundo y fueron recibidos 16,01 KB por segundo. A comparación del primer caso, el tiempo promedio en registrar un ticket aumentó considerablemente de 1,5 a 4,4 segundos.

Tabla 80. Resultados del caso 2 de la prueba de stress de consulta de activos.

Función	Tipo	Peticio-nes	T. Avg (ms)	T. Min (ms)	T. Max (ms)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
ConsultaActivos	HTTP POST	500	8715	1695	15171	18,01	12022,66

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 500, en la Tabla No. 80 se tiene una respuesta promedio de 8,715 segundos, en donde el tiempo máximo es de 15,171 segundos y el tiempo mínimo es de 1,695 segundos. En lo referente a los datos, fueron enviados un total de 18,0KB por segundo y fueron recibidos 12,02 MB por segundo. A

comparación del primer caso, el tiempo promedio en consultar todos los activos aumentó solamente 0,1 segundos.

Tabla 81. Resultados del caso 2 de la prueba de stress de consulta de tickets.

Función	Tipo	Peticio-nes	T. Avg (ms)	T. Min (ms)	T. Max (s)	Bytes Env. (KB/seg)	Bytes Recib. (KB/seg)
ConsultaTickets	HTTP POST	500	13864	6871	21,7	13,14	14200,69

Nota: Resultados de la prueba de stress.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Al aumentar el número de peticiones a 500, en la Tabla No. 81 se tiene una respuesta promedio de 13,864 segundos, en donde el tiempo máximo es de 21,665 segundos y el tiempo mínimo es de 6,871 segundos. En lo referente a los datos, fueron enviados un total de 13,14 KB por segundo y fueron recibidos 14,20 MB por segundo. A comparación del primer caso, el tiempo promedio en consultar todos los activos aumentó notablemente de 2,8 segundos a 13,9 segundos.

Análisis de Resultados

Considerando el número de peticiones y datos enviados, se puede concluir que el sistema cumple con las expectativas al ser evaluado en condiciones relativamente extremas para este proyecto, ya que, en la mayoría de los casos al realizar 500 peticiones concurrentes, el tiempo promedio de respuesta no supera los 10 segundos, además no se registró ningún tipo de error en el servidor durante la ejecución de las pruebas en ninguno de los casos presentados.

También cabe mencionar que el tiempo promedio que le lleva al sistema procesar las consultas es mayor al de los registros, ya que, al realizar 500 peticiones, el tiempo promedio en registrar activos, códigos QR y tickets es de 4,5 segundos aproximadamente, mientras que en las consultas de activos y tickets el tiempo promedio es de 11,3 segundos, dejando una diferencia de 6,8 segundos. Este dato puede variar debido a varios factores, como el número de registros que se encuentran

en la base de datos en el caso de las consultas o la cantidad de datos enviados para un determinado registro en el caso de las inserciones en la base de datos.

3.3.4. Pruebas de carga y consumo de recursos de la aplicación Android

3.3.4.1. Pruebas de carga

Para determinar los recursos de red que consume la aplicación Android al momento de ejecutarse y realizar las peticiones de consulta e ingreso de información, se utilizó el software "APPDYNAMICS" el cual permite visualizar los datos estadísticos en tiempo real del consumo de recursos. Se plantearon dos casos de prueba en el que se aumentará la carga para determinar el comportamiento de la aplicación móvil.

Caso 1

Se realizó 52 peticiones concurrentes entre las siguientes transacciones: Login de usuario, consulta y modificación de activos, consulta de laboratorios y registro de tickets de activos durante un tiempo de 2 minutos.

Pruebas de carga de la aplicación movil del caso 1.

Name	Crashes	Requests	Requests Per Minute	Network Request Time (ms)	HTTP Errors per Minute	Network Errors per Minute
Ecuador	-	52	17	434	0	0

Figura 63. Pruebas de carga para 52 peticiones.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 63 se puede observar que el tiempo total de respuesta fue de 434ms, procesando 17 peticiones por minuto. Además, se obtuvo 0% de errores en la red y en las peticiones HTTP.

Gráficas estadisticas de peticiones, errores de red y tiempos de respuesta del caso 1.

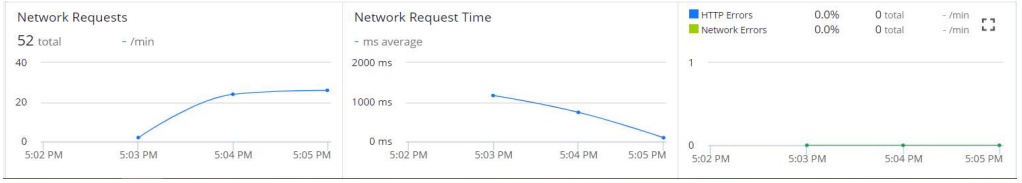


Figura 64. Resultados gráficos de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Gráfica estadística de número de peticiones por minuto para el caso 1.

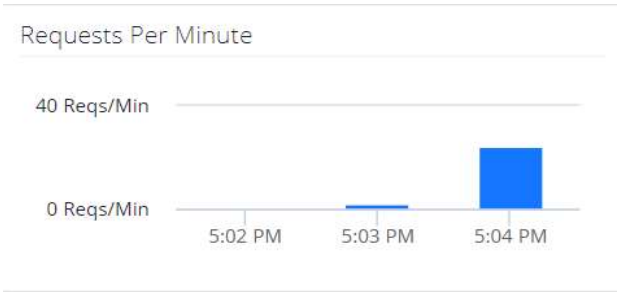


Figura 65. Número de peticiones por minuto de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Caso 2

Se realizó 128 peticiones concurrentes entre las siguientes transacciones: Login de usuario, consulta y modificación de activos, consulta de laboratorios y registro de tickets de activos durante un tiempo de 3 minutos.

Pruebas de carga de la aplicación movil del caso 2.

Name	Crashes	Requests	Requests Per Minute	Network Request Time (ms)	HTTP Errors per Minute	Network Errors per Minute
Ecuador	-	128	43	755	0	0

Figura 66. Pruebas de carga para 128 peticiones.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Figura No. 66 se puede observar que el tiempo total de respuesta fue de 755ms, procesando 43 peticiones por minuto. Además, se obtuvo 0% de errores en la red y en las peticiones HTTP.

Gráficas estadisticas de peticiones, errores de red y tiempos de respuesta del caso 2.

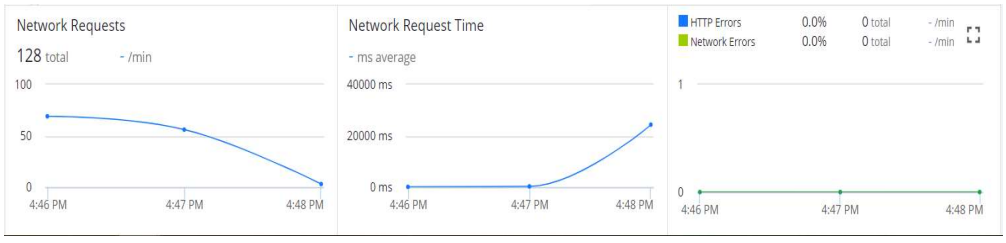


Figura 67. Resultados gráficos de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Gráfica estadística de número de peticiones por minuto para el caso 2.

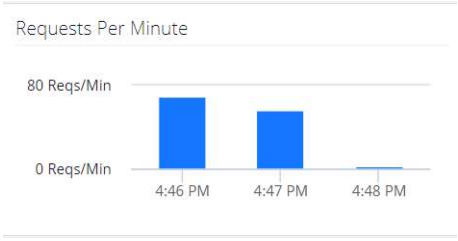


Figura 68. Número de peticiones por minuto de la prueba de carga.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Análisis de resultados

Con la información que proporcionan los dos casos de pruebas, se puede concluir que la aplicación móvil puede manejar una cantidad superior a 128 transacciones en un tiempo promedio de 595 ms, sin tener errores de red o perdidas de peticiones HTTP.

3.3.4.2. Pruebas de consumo de recursos

Para determinar los recursos de red, CPU y memoria RAM que ocupa la aplicación móvil al momento de ejecutarse y realizar las transacciones, se utilizó la herramienta "Monitor ADB" de Android Studio la cual muestra en tiempo real gráficos estadísticos del consumo de recursos.

Memoria RAM

En la Figura No. 69 se aprecia que el consumo máximo de consumo de memoria RAM del dispositivo es de 35,40MB.

Gráfica de consumo de memoria RAM de la aplicación móvil.

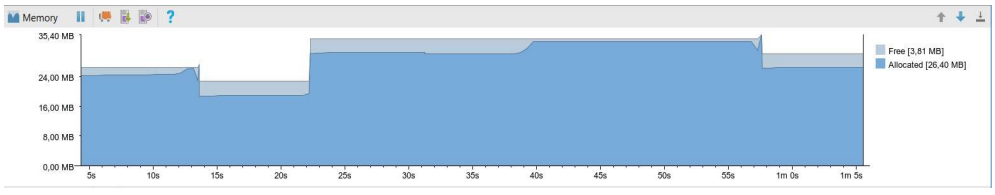


Figura 69. Consumo de memoria RAM de la aplicación móvil.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Tabla 82. Consumo de memoria RAM por operación.

Operación	Consumo de Memoria RAM
Inicio de la aplicación	24,00 MB
Login de usuario	25,00 MB
Pantalla de inicio	20,00 MB
Escáner de código QR	30,00 MB
Consultas	35,40 MB
Actualización/Inserción	26,00 MB

Nota: Valores de consumo de memoria RAM por operación.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 82 se puede observar un aproximado de la cantidad de memoria RAM que ocupan las diferentes operaciones, siendo en las consultas el valor más alto con 35,40 MB debido a que se debe enviar y recibir parámetros del Web Service.

Rendimiento del CPU

En la Figura No. 70 se aprecia que el porcentaje máximo de CPU que se ocupa para la aplicación es de 40%.

Gráfica de consumo de CPU de la aplicación móvil.

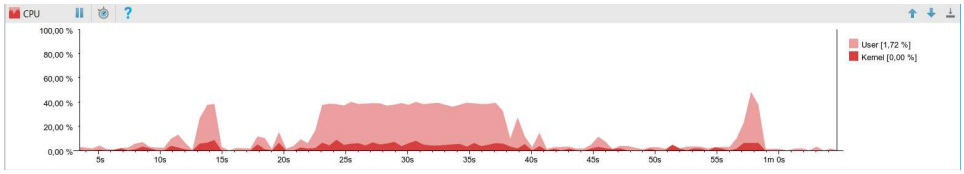


Figura 70. Consumo de CPU de la aplicación móvil.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Tabla 83. Consumo de CPU por operación.

Operación	Consumo de CPU
Inicio de la aplicación	1%
Login de usuario	1%
Pantalla de inicio	35%
Escáner de código QR	40%
Consultas	35%
Actualización/Inserción	40%

Nota: Porcentajes de consumo de CPU por operación.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 83 se puede observar un aproximado del porcentaje de CPU que ocupan las diferentes operaciones, siendo el escáner de código QR la más alta con

40% debido a que es necesario activar la cámara del dispositivo para cumplir con dicha función.

Consumo de Red

En la Figura No. 71 se aprecia que la aplicación móvil tiene una tasa de recepción máxima de datos de 49,73 KB/s y transmisión máxima de datos de 13,97 KB/s.

Gráfica de consumo de red de la aplicación móvil.

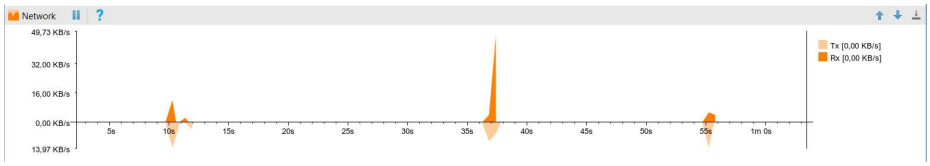


Figura 71. Consumo de recursos de red de la aplicación móvil.
Elaborador por: Andres Chisaguano, Joel Ludeña.

Tabla 84.Consumo de recursos de red por operación.

Operación	Recepción	Transmisión
Login de usuario	15,00 KB/s	12,00 KB/s
Consulta por código QR	48,00 KB/s	10,00 KB/s
Actualización/Inserción	5,00 KB/s	13,97 KB/s

Nota: Valores de consumo de recursos de red por operación.
Elaborador por: Andres Chisaguano, Joel Ludeña.

En la Tabla No. 84 se puede observar un aproximado de los valores de recepción y transmisión que se genera cuando se realizan las diferentes operaciones, siendo la tasa de recepción más alta la consulta por código QR, debido a que se solicita la información del código escaneado y la tasa de transmisión más alta la actualización o inserción, debido a que se envía la información necesaria para poder almacenarla.

Análisis de resultados

Con la información que proporcionan las gráficas de rendimiento, se puede concluir que la aplicación no consume una cantidad excesiva de recursos del dispositivo móvil. La cantidad de memoria RAM que se ocupa es 35,40 MB, el porcentaje máximo de CPU que se consumió es de un 40%, para procesos que se requieran usar el hardware del dispositivo, la tasa de recepción 48,00 KB/s y la tasa de transmisión es 13,97 KB/s, valores que no afectan al desempeño del dispositivo.

CONCLUSIONES

- Realizar el levantamiento y análisis de los procesos con los que cuenta el Data Center y Laboratorios del ICC facilitó la comprensión de las actividades de gestión que se realizan actualmente.
- El conocimiento, resultado del análisis de los procesos será la base para que los futuros proyectos que se desarrollen sigan un mismo estándar y facilitará la comunicación del flujo del proceso con los nuevos colaboradores cuando se requiera.
- De los procesos analizados, se puede concluir que ninguno se encuentra automatizado completamente y cada proceso se lleva por separado. Esta es una desventaja al momento de mantener la información centralizada, ya que cada proceso genera sus propios datos y no hay una conexión entre ellos, provocando que la gestión sea desorganizada y difícil de manejar.
- Para automatizar el proceso de control de inventarios y ticketing para soporte técnico, se tomó en cuenta el flujo de las actividades descritas en el manual de procesos previamente elaborado, lo cual facilitó que entender la lógica del negocio sea fácil y no haya la urgencia de acudir constantemente a solicitar información adicional sobre las actividades del proceso, ahorrando tiempo tanto a los interesados en el proyecto como a los desarrolladores del mismo.
- Con la automatización del proceso de ticketing para soporte técnico, se tendrá una constancia de las incidencias reportadas por los usuarios, la fecha en la que se reportó, el responsable asignado y el estado en el que se encuentra su requerimiento. Con esta información se evitará que los requerimientos reportados pasen por alto y se podrá dar fe al usuario de que su solicitud fue atendida para mejorar la satisfacción y confianza del mismo.

- Para realizar las operaciones CRUD con la base de datos, se creó un Web Service el cual actúa como intermediario entre las aplicaciones y la base de datos. Esta es una ventaja, debido a que no se tuvo que desarrollar una arquitectura independiente para la aplicación Web y para la aplicación móvil Android, debido a que las dos se comunican a través del Web Service para manejar la misma información y así mantener la consistencia y seguridad de los datos.
- Para aumentar la seguridad del sistema se utilizó las librerías "JWT" tokens, para que solamente los usuarios registrados en el sistema puedan consumir los métodos implementados en el Web Service. El token que se genera es único y tiene un tiempo de caducidad predefinido, además, es la única manera para que los métodos entreguen los datos que los usuarios solicitan.
- Con la implementación de un servicio REST se logró la encapsulación de datos, haciendo que el Web Service sea el único que entregue y reciba los datos por parte del usuario sin que haya apremio de que se conozca la estructura interna de la base de datos. Además, al hacer uso de formatos JSON y XML para el manejo de datos, permite la flexibilidad para el intercambio de información entre aplicaciones independientemente del lenguaje en el que fue programado.
- Al momento de realizar el registro del código QR no fue necesario almacenar los bytes de la imagen en la base de datos, debido a que, gracias a librería "QRCoder" solo es necesario establecer la cadena que se va a codificar y el tamaño de la imagen del código, reduciendo así la carga de almacenamiento y transferencia de datos.
- La utilización de la metodología de desarrollo "XP" fomentó la comunicación directa y constante con el cliente. Con cada iteración desarrollada, se presentaba al cliente una determinada funcionalidad del sistema para así conocer los posibles

cambios que se puedan presentar. Este proceso permitió que los tiempos de análisis y construcción del sistema se acorten, facilitando la integración de los módulos modificados y cumpliendo los requisitos establecidos por el cliente.

- El uso de la librería "Newtonsoft" facilitó el proceso de serialización y deserialización de los datos provenientes de los archivos JSON del Web Service, ya que solo es necesario una sola línea de código en la cual se define el tipo de dato que se desea procesar. Esta función es de mucha utilidad cuando se tienen listas u objetos en el mismo archivo.
- El uso de la API "Vision" de Google para Android, fue el que permitió que se pueda detectar los códigos QR pertenecientes a los Activos de Ti, haciendo uso únicamente de la funcionalidad Barcode Reader. Para acceder a las funcionalidades que presenta la API, no es necesario descargar las librerías, solo basta con hacer referencia a la página de los repositorios de "Google Play Services Vision", de donde se pueden obtener directamente los métodos para el tratamiento de imágenes como detección de rostros, códigos de barras y QR, texto, etc.
- Durante la implementación del escáner de códigos QR en la aplicación móvil se encontró que el evento de la detección del código se activaba una y otra vez hasta recibir los datos del Web Service, lo que producía una caída del sistema o lentitud en el dispositivo móvil. Para solucionar este inconveniente, se utilizó una variable temporal en la cual se guarda el valor anterior del código escaneado; si el valor de la variable temporal es igual al nuevo valor no se producen más llamadas a los métodos de comunicación con el Web Service, caso contrario se procede a realizar la solicitud y limpiar la variable temporal.

RECOMENDACIONES

- Se recomienda que los futuros proyectos de automatización de los procesos del Data Center se guíen en el manual de procesos que se encuentra en los anexos del proyecto. Además, se debe utilizar los formatos de la reportería que ya están establecidos.
- Para una futura versión de la aplicación web, se recomienda añadir la funcionalidad para generar alertas que permitan decidir o tomar decisiones sobre los activos que ya cumplieron con su vida útil.
- Para futuras versiones de la aplicación móvil con el objetivo de que no haya restricción de ejecución en diferentes dispositivos, se recomienda implementar una solución para iOS.
- Continuando con el proyecto desarrollado, se recomienda que se implemente un módulo para la gestión de mantenimientos preventivos, correctivos y mesa de ayuda para las incidencias, guiándose en el proceso de gestión de tickets para soporte técnico ya establecido.
- Se recomienda que se implemente un módulo para la asignación dinámica de horarios, préstamo de laboratorios y activos de TI, basándose en las actividades que se encuentran en la documentación de procesos.

LISTA DE REFERENCIAS

Aranda Software. (2018). *LA IMPORTANCIA DE UN INVENTARIO ACTUALIZADO*. Obtenido de https://arandasoft.com/la-importancia-de-un-inventario-actualizado/

BBVAOPEN4U. (23 de Marzo de 2016). API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos. España.

Bizagi. (2018). Obtenido de Bizagi BPMN Modeler: https://www.bizagi.com/es/productos/bpm-suite/modeler

Bizagi. (2018). *Modelado para ejecución*. Obtenido de http://help.bizagi.com/bpm-suite/es/index.html?modelado_para_ejecucion.htm

Can, L. (16 de Junio de 2015). Análisis y estudio del código QR y su aplicación en centros de información. Salamanca, España.

Canchala, A. (2016). *UML, ejemplo sencillo sobre Modelado de un Proyecto*. Obtenido de https://msdn.microsoft.com/es-es/library/bb972214.aspx

DataCenterMarket. (18 de Noviembre de 2015). Las soluciones de DCIM seducen a las grandes cuentas. Obtenido de http://www.datacentermarket.es/tendencias-tic/noticias/1085361032809/soluciones-dcim-seducen-grandes-cuentas.1.html

Developers Android. (25 de Abril de 2018). *Conoce Android Studio*. Obtenido de https://developer.android.com/studio/intro/?hl=es-419

Gómez, J. M. (2014). Modelo de datos. Clasificación. En J. M. Gómez, *Diseño de bases de datos relacionales* (págs. 15-16). Ediciones Paraninfo, S.A.

Gustavo, C. (2017). *Sistema Automatizado de Control de Inventarios* . Obtenido de https://www.workana.com/es/job/sistema-automatizado-de-control-de-inventarios

IBM. (2018). *Tipos de aplicaciones móviles*. Obtenido de https://www.ibm.com/support/knowledgecenter/es/SS8H2S/com.ibm.mc.doc/dev_source/references/dev_about_app_types.htm

INCIBE. (29 de Diciembre de 2016). Inventario de activos y gestión de la seguridad en SCI. España. Obtenido de https://www.certsi.es/blog/inventario-activos-y-gestion-seguridad-sci

Jiménez, F. (Febrero de 2017). El Data Center es imprescindible en la Transformación Digital. *ITCONIC*, 38-39. Obtenido de https://www.itconic.com/data-center-imprescindible-la-transformacion-digital/

Joskowicz, J. (2008). Reglas y Prácticas en eXtreme Programming. Vigo, Pontevedra, España. Obtenido de http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf

Junta de Andalucía. (15 de Febrero de 2013). Patrón Modelo Vista Controlador. Andalucía, Sevilla, España. Obtenido de http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/122

JWT. (2018). Introduction to JSON Web Tokens. Obtenido de https://jwt.io/introduction/

Letelier, P., & Penadés, M. C. (2012). Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). Valencia, España. Obtenido de http://roa.ult.edu.cu/bitstream/123456789/477/1/masyxp.pdf

López, J. (2018). *Infraestructura de TI adquirida para los nuevos laboratorios.* Quito.

Macias, S. (21 de Marzo de 2018). Construyendo una Web API REST segura con JSON Web Token en .NET.

Maldonado, D. (10 de Junio de 2018). *¿Qué es Infraestructura de TI y cuáles son sus componentes?* Obtenido de http://www.icorp.com.mx/blog/infraestructura-de-ti-componentes/

Maldonado, D. (09 de Febrero de 2018). Los beneficios de un sistema de gestión de incidentes. Obtenido de http://www.icorp.com.mx/blog/los-beneficios-sistema-gestion-incidentes/

Martín, S. (25 de Octubre de 2017). ¿Sabes si tu empresa necesita utilizar un software de inventario? Madrid, Madrid, España. Obtenido de https://integriaims.com/software-de-inventario/

Mendoza, A. (24 de Enero de 2018). Herramientas de Control de Activos de TI: ¿Para qué sirven? Madrid, España. Obtenido de http://www.gb-advisors.com/es/control-de-activos-de-ti/

Microsoft. (19 de Julio de 2015). Introducción al lenguaje C# y .NET Framework. Estados Unidos. Obtenido de https://docs.microsoft.com/es-es/dotnet/csharp/getting-started/introduction-to-the-csharp-language-and-the-net-framework

Microsoft. (22 de Junio de 2017). Licencias y Desarrollo en Visual Studio Community. Obtenido de https://social.msdn.microsoft.com/Forums/es-ES/1a018c22-4504-4218-a9aa-3c2d326aa18c/licencias-y-desarrollo-en-visual-studio-community?forum=vsgenerales

Microsoft. (2018). ASP.NET y Visual Studio para Web. Obtenido de https://msdn.microsoft.com/es-es/library/dd566231(v=vs.120).aspx

Microsoft. (03 de Octubre de 2018). Introducción a ASP.NET MVC 5. Obtenido de https://docs.microsoft.com/es-es/aspnet/mvc/overview/getting-started/introduction/getting-started

Microsoft. (01 de 06 de 2018). Introducción al IDE de Visual Studio. Obtenido de https://docs.microsoft.com/es-es/visualstudio/ide/visual-studio-ide

Pacio, G. (4 de Mayo de 2013). DCIM mucho más que una Herramienta de Gestión. Obtenido de http://www.datacentershoy.com/2013/05/dcim-mucho-mas-que-una-herramienta-de.html

postgresql. (2017). *About PostgreSQL* . Obtenido de https://www.postgresql.org/

R2Docuo. (15 de Marzo de 2018). Cómo crear un Sistema de Ticketing. Obtenido de https://www.r2docuo.com/es/blog/como-crear-un-sistema-de-ticketing

Roth, D., Anderson, R., & Luttin, S. (15 de Noviembre de 2018). Introducción a ASP.NET Core.

ServiceTonic. (2018). ITIL v3. Gestión de Incidencias. España. Obtenido de https://www.servicetonic.es/itil/itil-v3-gestion-de-incidencias/

Servisoftcorp. (9 de mayo de 2019). *Definición y cómo funcionan las aplicaciones móviles*. Obtenido de https://www.servisoftcorp.com/definicion-y-como-funcionan-las-aplicaciones-moviles/

Silberschatz, A., Korth, H., & Sudarshan, S. (2002). *FUNDAMENTOS DE BASES DE DATOS* (Cuarta ed.). Madrid: McGRAW-HILL.

Unitag. (2018). ¿Qué es un código QR? Obtenido de https://www.unitag.io/es/qrcode/what-is-a-qrcode

Universidad Católica Los Ángeles Chimbote. (2017). Metodología de Desarrollo de Software. Chimbote, Perú. Obtenido de https://www.uladech.edu.pe/images/stories/universidad/documentos/2018/metodologia-desarrollo-software-v001.pdf

White, A. S., & Miers, D. (2009). *BPMN - Guía de referencia y modelado.* Florida: Future Strategias Inc.

Young, A. d. (2017). *Acerca de campos de tickets*. Obtenido de https://support.zendesk.com/hc/es/articles/203661506-Acerca-de-campos-de-tickets

ANEXOS

Para revisar los anexos del presente proyecto, por favor diríjase al CD.

Anexo A: Manual de procesos del Data Center y laboratorios del ICC.

Anexo B: Manual técnico de configuración del sistema.

Anexo C: Manual de usuario.

Anexo D: Especificación de requisitos de usuario

Anexo E: Diagramas y diccionario de datos.

Anexo F: Diagramas UML del sistema.

Anexo G: Prototipos navegacionales del sistema.

Anexo H: Pruebas del sistema.