UNIVERSIDAD NACIONAL DEL ALTIPLANO

FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

SISTEMA WEB INTEGRADO PARA LA OFICINA DE CIRCULACIÓN TERRESTRE DE LA DIRECCIÓN REGIONAL DE TRANSPORTES Y COMUNICACIONES PUNO

TESIS

PRESENTADA POR:

Bach. Mayda Chipana Mamani

Bach. Rosa Maria Gomez Sacari

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ESTADÍSTICO E INFORMÁTICO

PUNO – PERÚ

2018

UNIVERSIDAD NACIONAL DEL ALTIPLANO

FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

SISTEMA WEB INTEGRADO PARA LA OFICINA DE CIRCULACIÓN TERRESTRE DE LA DIRECCIÓN REGIONAL DE TRANSPORTES Y COMUNICACIONES PUNO

TESIS PRESENTADA POR:

Bach. MAYDA CHIPANA MAMANI
Bach. ROSA MARIA GOMEZ SACARI

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ESTADÍSTICO E INFORMÁTICO

APROBADA POR:

PRESIDENTE DEL JURADO	:	**Dr. BERNABÉ CANQUI FLORES**
PRIMER MIEMBRO	:	**M.Sc. ERNESTO NAYER TUMI FIGUEROA**
SEGUNDO MIEMBRO	:	**Ing. RONALD MAMANI MAYTA**
DIRECTOR DE TESIS	:	**Dr. JUAN REYNALDO PAREDES QUISPE**

Área	:	**Informática**
Tema	:	**Sistema de Información y Base de Datos**
Fecha de sustentación	:	**28/06/2018**

DEDICATORIA

Al ser más supremo a Dios, por permitirnos llegar a este momento, por concedernos vida y sabiduría y por darnos la fuerza dia tras dia y guiarnos en cada momento de nuestras vidas.

Con todo mi cariño y amor para mis queridos padres Raúl y Luisa, por todos sus esfuerzos y sacrificios que hicieron por mí, por su apoyo incondicional, sus valiosos consejos y estar siempre conmigo y permitirme el haber llegado hasta este momento tan importante de mi formación profesional.

A mis hermanos Abraham y Elías por su constante apoyo y con quienes sé que puedo contar para todos mis logros.

Rosa Maria G. S.

Con todo mi cariño y amor a mi madre Maritza gracias por todo tu esfuerzo, apoyo y por la confianza que depositaste en mí, gracias porque siempre has estado a mi lado y permitirme el haber llegado hasta este momento tan importante de mi formación profesional.

A mi hermano Edward por apoyarme siempre y con quien sé que puedo contar para todos mis logros.

Mayda Ch. M.

AGRADECIMIENTO

A la Facultad de Ingeniería Estadística e Informática de la Universidad Nacional del Altiplano - Puno, por cobijarnos en sus aulas estos cinco años de formación.

A los catedráticos de la Escuela Profesional de Ingeniería Estadística e Informática, por compartir sus conocimientos con sus estudiantes y contribuir en la formación profesional, por absolver cada uno de nuestras dudas, por su paciencia y calma en las sesiones de aprendizaje, nuestro cariño, respeto y admiración por cada uno de ellos.

Un agradecimiento muy grande también a los jurados Dr. Bernabe Canqui Flores, M. Sc. Ernesto Nayer Tumi Figueroa, Ing. Ronald Mamani Mayta y en especial a nuestro director de tesis Dr. Juan Reynaldo Paredes Quispe, quienes participaron en nuestra formación profesional.

Y a todas aquellas personas y amigos, que de una y otra forma estuvieron a nuestro lado apoyándonos en todo momento.

¡A todos mil gracias!

MAYDA CHIPANA MAMANI

ROSA MARIA GOMEZ SACARI

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ACRONIMOS

MTC	Ministerio de Transportes y Comunicaciones
DRTC	Dirección Regional de Transportes y Comunicaciones
ASP	Páginas Activas del Servidor
UML	Lenguaje Unificado de Modelado
CLR	Entorno Común de Ejecución
CTS	Sistema Común de Tipos
MVC	Modelo Vista Controlador
XML	Lenguaje de Marcado Extensible
ADO	Componentes de Acceso a Datos
API	Interfaz de Programación de Aplicaciones
IO	Entrada/Salida
VB	Visual Basic
DLL	Biblioteca de Vínculos Dinámicos
EXE	Archivo Ejecutable
ECMA	Asociación Europea de Fabricantes de Ordenadores
HTTP	Protocolo de Transferencia de Hipertexto
REST	Transferencia de Estado Representacional
CORS	El Intercambio de Recursos de Origen Cruzado
RAD	Desarrollo Rápido de Aplicaciones
BCL	Biblioteca de Clases Base
SDK	Kit de Desarrollo de Software

RESUMEN

Los procesos y tramites en la Oficina de Circulación Terrestre de la Dirección Regional de Transportes Puno tienen un tiempo de atención prolongada esto, sin tener en cuenta la extensión de la cola y el desorden que se genera, para reducir el tiempo es necesario implementar un sistema web integrado por lo cual se plantea el siguiente objetivo: Desarrollar un Sistema Web Integrado para la Oficina de Circulación Terrestre. Para el desarrollo del sistema web se ha empleado la metodología SCRUM que es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente en equipo, y obtener el mejor resultado posible del proyecto y para el modelado del sistema web se utilizó la metodología UML. Las conclusiones de la investigación son: Se diseñó e implemento un sistema web integrado utilizando la metodología ágil SCRUM, así como para el modelado se ha empleado el lenguaje de Modelamiento Unificado (UML), el desarrollo de dicho sistema mejora la calidad de atención. Se validó el funcionamiento del sistema web integrado para los procesos y tramites de la Oficina de Circulación Terrestre, con la ficha de evaluación ISO - 9126 en una escala del 27 - 135 puntos, dando como resultado 108 puntos, que significa que el sistema web integrado cumple con los requisitos de calidad para su funcionamiento. El tiempo de atención redujo notablemente con la implementación del sistema web integrado para los procesos y tramites de la Oficina de Circulación Terrestre de la Dirección Regional de Transportes y Comunicaciones Puno.

Palabras claves: Circulación Terrestre, Comunicación, Integración, Sistema, Web.

ABSTRACT

The processes and procedures in the Office of Land Traffic of the Puno Regional Transport Directorate have a prolonged attention time this, without taking into account the extension of the queue and the disorder that is generated, to reduce the time it is necessary to implement a system integrated web, for which the following objective is proposed: To develop an Integrated Web System for the Office of Land Traffic. For the development of the web system the SCRUM methodology has been used, which is a process in which a set of good practices are applied on a regular basis to work collaboratively as a team, and obtain the best possible result of the project and for the modeling of the web system. the UML methodology was used. The conclusions of the research are: An integrated web system was designed and implemented using the agile SCRUM methodology, as well as modeling using the Unified Modeling language (UML), the development of said system improves the quality of attention. The operation of the integrated web system for the processes and procedures of the Office of Land Traffic was validated, with the evaluation form ISO - 9126 on a scale of 27 - 135 points, resulting in 108 points, which means that the integrated web system meets the quality requirements for its operation. The time of attention reduced notably with the implementation of the integrated web system for processes and procedures of the Office of Land Traffic of the Regional Directorate of Transport and Communications Puno.

Key words: Land Circulation, Communication, Integration, System, Web.

CAPÍTULO I

INTRODUCCIÓN

La Dirección Regional de Circulación Terrestre como oficina que brinda servicio de trámite de brevetes, así como la obtención de nuevos brevetes, trámite de duplicados, denuncia y des habilitación de brevetes perdidos, pero a la presente fecha no cuenta con sistemas informáticos que controlen el trámite de los documentos ingresados, si ponemos el hecho de que las oficinas requieren estar interconectadas para el registro de documentos y el estado final de los mismos a través del seguimiento mediante un software.

El desarrollo de un sistema web integrado permitirá obtener una mejora en el tiempo de trámite, así como un control más detallado de los documentos ingresados de las personas que realizan tramites. Para el desarrollo del software se ha hecho uso de la metodología ágil SCRUM, así como para el modelado se ha empleado el lenguaje de Modelamiento Unificado para la diagramación de las distintas etapas en el software y sobre todo en el procedimiento que toman las oficinas bajo los distintos trámites.

1.1. Planteamiento del Problema

En la actualidad en la Dirección Regional de Transportes Puno, donde se realizan varios trámites así como la obtención de nuevos brevetes, duplicados, denuncias y des habilitación de brevetes perdidos entre otros , el tiempo de atención tiene una larga duración, sin tener en cuenta la extensión de la cola y el desorden que se genera, el desarrollo de un sistema web integrado ayudará significativamente a los trabajadores, llevar una buena atención de procesos y tramites, permitirá obtener una mejora en el tiempo de atención de procesos y trámites .

1.2. Formulación Del Problema

¿De qué manera mejorará la atención de los procesos y tramites en la Oficina de Circulación Terrestre de la Dirección Regional de Transportes Puno 2018?

1.3. Objetivos de la Investigación

1.3.1. Objetivo General

Desarrollar un Sistema Web Integrado para la Oficina de Circulación Terrestre de la Dirección Regional de Transportes y Comunicaciones Puno.

1.3.2. Objetivos Específicos

- Analizar y diseñar el sistema web integrado para la Oficina de Circulación Terrestre de la Dirección Regional de Transportes y Comunicaciones Puno.
- Validar la implementación del sistema con el ISO 9126, para el uso y

fiabilidad del software.

- Comparar los tiempos de atención de procesos y trámites documentarios antes y después de la implementación del sistema web integrado.

1.4. Hipótesis de la investigación

El Desarrollo de Sistema Web Integrado para los procesos y tramites; mejorará el tiempo de atención en la Oficina de Circulación Terrestre.

1.5. Justificación de la Investigación

El presente proyecto de investigación es de gran interés que responde a la necesidad de contar con un nuevo sistema web integrado para la Oficina de Circulación Terrestre de la Dirección Regional de Transportes y Comunicaciones Puno.

El uso de las Tecnologías de Información contribuirá en optimizar el proceso de varios trámites, mediante la rápida identificación sin tener que demorar en la escritura de nombres complicados o extensos que generan demoras en el proceso, mediante el ingreso o lectura del Documento Nacional de Identidad DNI, se procederá a la generación del trámite.

1.6. Delimitación de la Investigación

El proyecto de sistema web integrado para la Oficina de Circulación Terrestre de la Dirección Regional de Transportes y Comunicaciones Puno solo abarca la sede Puno.

CAPÍTULO II

REVISIÓN DE LITERATURA

2.1. Marco Teórico

2.1.1. Antecedentes de la Investigación

Herrera (2016), menciona que la implementación de un Sistema de Información para Instituto de Informática de la Universidad Nacional del Altiplano - PUNO - 2016, la permite la atención rápida y oportuna a las solicitudes de los estudiantes; además permite al personal administrativo realizar las actividades de manera automatizada y eficiente. Tal como se demostró en el resultado de la prueba estadística de comparación de medias para datos apareados, como (−5.89) < Zt (−1.645). Lo que indica que el tiempo promedio de atención es mucho mejor con la implementación de una aplicación web permitió que los estudiantes y docentes puedan acceder a los distintos procesos que se realizan en la secretaria del Instituto de Informática de la Universidad Nacional del Altiplano Puno 2016. El resultado a través de la encuesta demostró que el 79% de los estudiantes afirma que se ha reducido el tiempo de atención de una manera eficiente mediante la implementación

del Sistema de Información para el Instituto de Informática de la Universidad Nacional del Altiplano Puno 2016 "SIPII".

Jihuallanca (2016), logró implementar un sistema para los procesos de búsqueda y consulta de perfiles usando la tecnología web, el desarrollo de dicho sistema mejora la calidad de servicio en la Unidad de Pensiones y Liquidaciones de la Universidad Nacional del Altiplano Puno.

Hay Diferencia significativa en el análisis del tiempo de demora entre el antes y el después de la implementación del sistema en la búsqueda de información. El tiempo de demora redujo notablemente con la implementación del sistema de consulta de perfiles de la Unidad de Pensiones y Liquidaciones de la Universidad Nacional del Altiplano Puno. Se realizó un análisis estadístico utilizando la prueba t-student para datos relacionados. Lo que nos indica que el tiempo promedio de búsqueda de perfiles es mucho mejor.

Vilca & Alferez (2014), menciona que la aplicación web de tramite documentario mejora la accesibilidad a la información de los usuarios del edificio administrativo de la UNA-Puno, donde la prueba de post prueba supero a la de pre prueba, 11.72 > 7.96 respectivamente. Se realizó el análisis y diseño de la aplicación web, en base a procesos del negocio de tramite documentario, con lo cual se concluye que la aplicación mejoro la visualización de los documentos en el Edificio Administrativos. El diseño, normalización y posterior implementación de la base de datos utilizando el gestor MySql, permitió la agilización de la búsqueda de la información en el Edificio Administrativo. Obteniendo los requerimientos a través del levantamiento de información en las reuniones sostenidas con el personal involucrado en los

procesos de negocio de las diferentes áreas del Edificio Administrativos y los resultados de las pruebas, se concluyó que la aplicación web vincula dichas áreas, mejorando y agilizando el flujo del trámite documentario.

Montenegro (2015), menciona que la investigación permite concluir que, el tiempo de consulta de los docentes y estudiantes del Instituto Superior Publico Pedagógico de Juliaca en el año 2015, antes de la implementación del sistema web se tardaba más en la consulta y el préstamo de un libro. Luego de la implementación del sistema web de biblioteca en el Instituto Superior Publico Pedagógico de Juliaca, el tiempo de consulta de los docentes y estudiantes fue más rápido por la cual los docentes y estudiante concurren más a la biblioteca del Instituto Superior Publico Pedagógico de Juliaca para realizar préstamo de los libros y consulta de los libros existente en la biblioteca. Efectivamente la implementación del sistema web de biblioteca mejoro notablemente en la consulta de los libros y tesis de la biblioteca del Instituto Superior Publico Pedagógico de Juliaca en el año 2015 y podemos decir que influye bastante en el tiempo de consulta y el uso de los mismos libros ya que por falta de conocimiento de los mismo los usuarios no hacían el uso de la biblioteca de manera permanentemente.

RENIEC (2012), sistema integrado de trámite documentario – SITD permite un gran ahorro en el uso de papel y de tinta de impresión, ya que todos los documentos que ingresan a la institución y que circulan por función, entre las diversas unidades orgánicas están digitalizados y son direccionados hacia las personas que lo van atender. El SITD genera ahorros en tiempo y recursos que pueden ser destinados hacia otras tareas prioritarias, así como el menor

gasto de papel y tóner para la impresión, apoya las actividades orientadas a la conservación del medio ambiente.

2.2. Marco Conceptual

2.2.1. Ministerio de Transportes y Comunicaciones

El Ministerio de Transportes y Comunicaciones (MTC) es el órgano del Estado Peruano responsable del desarrollo de los sistemas de transporte, la infraestructura de las comunicaciones y telecomunicaciones. EL servicio de trámites del Ministerio de Transportes y Comunicaciones. Tiene la finalidad de facilitar las gestiones que los ciudadanos requieran hacer en todos los sectores del MTC. **(http://www.mtc.gob.pe)**

2.2.2. Oficina de Circulación Terrestre

La Dirección General de Transporte Terrestre es un órgano de línea de ámbito nacional encargado de normar el transporte y tránsito terrestre; regular y autorizar, la prestación de servicios de transporte terrestre por carretera y servicios complementarios, así como del tránsito terrestre. **(http://www.mtc.gob.pe)**

Sus funciones específicas son:

- Proponer y ejecutar las políticas orientadas a la administración de los servicios de transporte terrestre de personas y de mercancías.
- Proponer proyectos de normas, reglamentos y demás disposiciones relacionadas con las actividades de transporte y tránsito terrestre.
- Otorgar autorizaciones para la prestación de servicios de transporte

terrestre de personas y de mercancías de ámbito nacional e internacional y sus servicios complementarios.

- Conducir la gestión y mantener actualizados los registros administrativos nacionales relacionados al transporte y tránsito terrestre por carretera, en coordinación con los gobiernos regionales y locales según corresponda.
- Mantener un sistema estándar de licencias de conducir; normar, coordinar y fiscalizar el proceso de otorgamiento de éstas a nivel nacional y emitir licencias de conducir en el ámbito de su competencia.
- Mantener un sistema estándar de homologación, certificación, verificación y revisiones técnicas de vehículos; así como normar su operación.
- Desarrollar las actividades orientadas a promover la educación y seguridad vial, de competencia del Ministerio.
- Producir estadísticas relacionadas al transporte y tránsito terrestre de personas y mercaderías en su ámbito de competencia.
- Participar en representación del Ministerio como organismo nacional competente de Transporte Terrestre, en eventos nacionales e internacionales sobre transporte y tránsito terrestre.

2.2.3. Atención o Gestión Documentaria

La aplicación de un programa de gestión documental permite un incremento exponencial de la productividad empresarial, ya que facilita la ubicación y el manejo de la información además que reduce en gran medida, el exceso de documentos que generalmente se conservan en las organizaciones y que no son importantes para la misma **(Ibai 2005),** comenta algunas de estas ventajas de la siguiente forma:

- Reducción del tiempo de consulta de un documento en papel.
- Reducción del tiempo de consulta de documentos electrónicos.
- Reducción de los costes de archivado
- Reducción de la recuperación de un documento.
- Acceso concurrente a un documento.
- Mejora de atención a los clientes.
- Reducción de costes legales.
- Reducción de costes de acceso a la documentación.
- Posibilidad de integrarse con subsistemas de gestión documental específicos.
- Incremento en la satisfacción de los usuarios internos.

2.2.4. Trabajo Colaborativo

Se define como las aportaciones que hace una persona a sus compañeros de equipo (uno o dos personas) en cuanto a experiencias, comentarios, sugerencias y reflexiones sobre el trabajo que se ha desarrollado cada miembro del equipo, a su vez espera que sus compañeros de equipo contribuyan en el mismo sentido. Para después transformar el trabajo individual el producto más eficiente. **(https://es.wikipedia.org)**

El trabajo colaborativo consiste en una forma de organización de pequeños grupos de trabajo dado por alumnos o personas que desean desempeñar una labor en conjunto, este tipo de metodología didáctica parte de una nueva propuesta en la que la responsabilidad en el aprendizaje es cosa del grupo que desempeña el trabajo propuesto y no tanto en el profesorado. De esta manera, la labor debe fluir de una forma continuada

con las interacciones del grupo en sí. **(Onetti, 2011)**

En relación con la definición del aprendizaje en grupo**,** lo plasma no como una innovación en la educación sino, como una perspectiva metodológica que tienen un largo recorrido en el mundo de la educación. Trata de una concepción de cómo se aprende, es decir, como una forma de aprender en la que se analiza la cooperación del grupo con sus características específicas. **(Clemente,1997)**

2.2.5. Metodologías Ágiles

Este enfoque nace como respuesta a los problemas que puedan ocasionar las metodologías tradicionales y se basa en dos aspectos fundamentales, retrasar las decisiones y la planificación adaptativa. Basan su fundamento en la adaptabilidad de los procesos de desarrollo.

Las metodologías ágiles de desarrollo están especialmente indicadas en proyectos con requisitos poco definidos o cambiantes.

2.2.5.1. Metodología de Desarrollo Ágil SCRUM

SCRUM es una metodología ágil de desarrollo de software para la gestión de proyectos basados en la iteración y entregas incrementales de un producto o servicio, es una manera de afrontar los proyectos de creación de aplicaciones de forma iterativa, rápida y eficaz. Se basa en trabajar con equipos pequeños multidisciplinares, formados por un número de personas que suele ir de dos a ocho. Se apuesta por el talento frente a la estimación de tiempos tradicional, que dicho sea de paso no es válida en desarrollo de software dado que los tiempos de desarrollo estimados por las vías

tradicionales rara vez se cumplen durante la vida de un proyecto.

Es un proceso donde se aplica las buenas prácticas para el trabajo colaborativo y en equipo, que permite obtener mejores resultados de un proyecto, garantizando así la correcta comprensión de las actividades, tareas asignadas a cada equipo. Al ser un trabajo en equipo, se aprovecharía al máximo la participación de cada integrante, generando un ambiente de colaboración y apoyo, donde son tenidas en cuenta cada opinión, generando en cada individuo el sentido de pertenencia e importancia dentro de los procesos de la organización. **(http://www.scrumguides.org)**

Es la colección de procesos para la gestión de proyectos, que permite centrarse en la entrega de valor para el cliente y la potenciación del equipo para lograr su máxima eficiencia, dentro de un esquema de mejora continua. **(Díaz, 2009)**

Características de SCRUM, especialmente indicada para proyectos con un rápido cambio de requisitos. El desarrollo de software se realiza mediante iteraciones, denominadas Sprints, con una duración de 30 días. El resultado de cada Sprint es un incremento ejecutable que se muestra al cliente. La realización de reuniones a lo largo del proyecto, entre ellas se destaca la diaria del equipo de desarrollo, con una duración aproximada de 15 minutos y con miras a la coordinación e integración de actividades. **(Canós, 2003)**

SCRUM propone las siguientes tres fases

➢ **Fase de Planificación**

Planeación: Se define el equipo, herramientas, el sistema de desarrollo y se crea el product backlog con la lista de requerimientos conocidos junto con sus prioridades y se estima el esfuerzo necesario para llevarlo acabo.

Diseño Arquitectónico: Se define la arquitectura del producto que permita implementar los requerimientos.

➢ **Fase de Desarrollo:** Es la parte ágil, donde el sistema se desarrolla en Sprints.

➢ **Fase de Finalización:** Incluye integración, testing y documentación. Indica la implementación de todos los requerimientos, quedando el product backlog vacío. **(Gonzales & Méndez, 2010)**

Figura N° 01: Roles, artefactos y eventos principales de SCRUM.

Fuente: (Gonzales & Méndez, 2010).

En SCRUM intervienen 3 roles fundamentalmente

➢ **Product Owner:** Es la persona que toma las decisiones, y es la que realmente conoce el negocio del cliente y su visión del producto. Se encarga de escribir las ideas del cliente, las ordena por prioridad y las coloca en el Product Backlog.

➢ **ScrumMaster:** Es el encargado de comprobar que el modelo y la metodología funciona. Eliminará todos los inconvenientes que hagan en el proceso no fluya e interactuará con el cliente y con los gestores.

➢ **Developer Team:** Suele ser un equipo pequeño de una 5- 9 personas y tienen autoridad para organizar y tomar decisiones para conseguir su objetivo. Está involucrado en la estimación del esfuerzo de las tareas del Backlog.

En SCRUM intervienen 3 artefactos

➢ **Pila del producto**: (Product Backlog) lista de requisitos de usuario, que a partir de la visión inicial del producto crece y evoluciona durante el desarrollo.

➢ **Pila del sprint**: (Sprint Backlog) lista de los trabajos que debe realizar el equipo durante el sprint para generar el incremento previsto.

➢ **Incremento**: Resultado de cada sprint.

En SCRUM intervienen 5 eventos

➢ **Sprint**: Es el núcleo central que genera el pulso de avance por tiempos prefijados (Time Boxing). Se denomina sprint a cada ciclo o iteración de

trabajo que produce una parte del producto terminada y funcionalmente operativa (incremento)

➢ **Reunión de Planificación del Sprint**: Reunión de trabajo previa al inicio de cada sprint en la que se determina cuál va a ser el objetivo del sprint y las tareas necesarias para conseguirlo.

➢ **Scrum Diario*:*** Breve reunión diaria del equipo, en la que cada miembro responde a tres cuestiones:

- El trabajo realizado el día anterior.
- El que tiene previsto realizar.
- Cosas que puede necesitar o impedimentos que deben eliminarse para poder realizar el trabajo.

Cada persona actualiza en la pila del sprint el tiempo o esfuerzo pendiente de sus tareas, y con esta información se actualiza a su vez el gráfico con el que el equipo monitoriza el avance del sprint (Burn-Down).

➢ **Revisión del sprint*:*** Análisis e inspección del incremento generado, y adaptación de la pila del producto si resulta necesario.

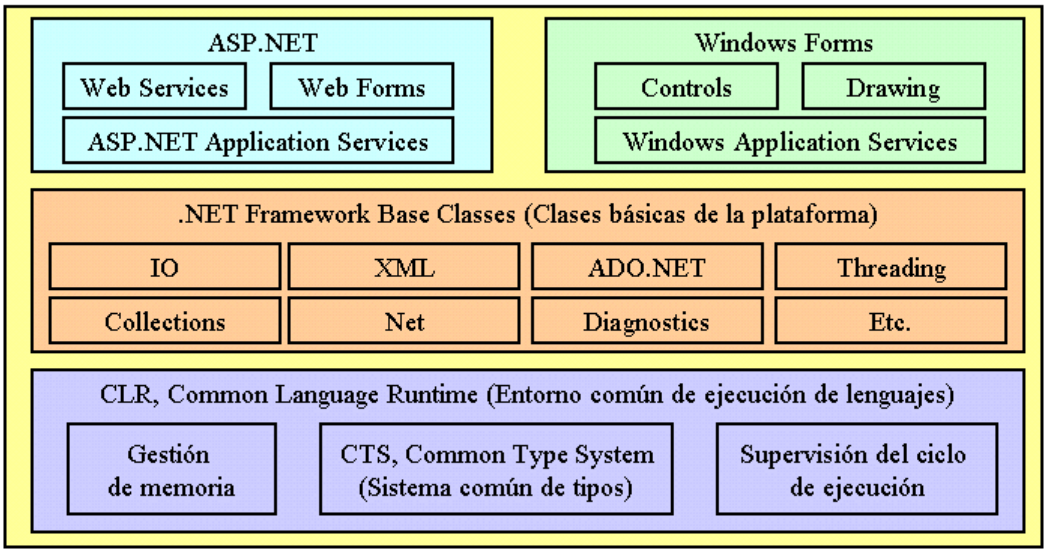
➢ **Retrospectiva del sprint:** Revisión de lo sucedido durante el Sprint. Reunión en la que el equipo analiza aspectos operativos de la forma de trabajo y crea un plan de mejoras para aplicar en el próximo sprint.

2.2.6. +Net Framework

NET Framework constituye la plataforma y elemento principal sobre el que se asienta Microsoft.NET. De cara al programador, es la pieza fundamental de todo este nuevo modelo de trabajo, ya que proporciona las herramientas y servicios que necesitará en su labor habitual de desarrollo.

NET Framework permite el desarrollo de aplicaciones a través del uso de un conjunto de herramientas y servicios que proporciona, y que pueden agruparse en tres bloques principales: el Entorno de Ejecución Común o Common Language Runtime (CLR a partir de ahora); la jerarquía de clases básicas de la plataforma o .NET Framework Base Classes; y el motor de generación de interfaz de usuario, que permite crear interfaces para la web o para el tradicional entorno Windows, así como servicios para ambos entornos operativos. **(Nuñez, 2002)**

Figura N° 02: Esquema de componentes .NET Framework.



Fuente: (Nuñez, 2002).

En la base del entorno de ejecución, se encuentra el CLR, que constituye el núcleo de NET Framework, encargándose de la gestión del código en cuanto a su carga, ejecución, manipulación de memoria, seguridad, etc.

En el nivel intermedio, se sitúa la jerarquía de clases básicas del entorno de ejecución, que constituyen un sólido API de servicios a disposición del programador, para multitud de tareas como, gestión del sistema de ficheros, manipulación multihebra, acceso a datos, etc.

Finalmente, en el nivel superior, encontramos las clases que permiten el diseño del interfaz de usuario de nuestras aplicaciones. Si necesitamos desarrollar aplicaciones para Internet, utilizaremos ASP.NET, que nos provee de todo lo necesario para crear aplicaciones para la Red: web forms, web services, etc.

Y no piense el programador tradicional de Windows, que todo en .NET Framework es programación para Internet. La plataforma no se ha olvidado de este colectivo de programadores, que necesitan desarrollar programas para este sistema operativo, y pone a su disposición los denominados Windows Forms, la nueva generación de formularios, con características avanzadas y muy superiores a las del motor de generación de formularios de VB 6.

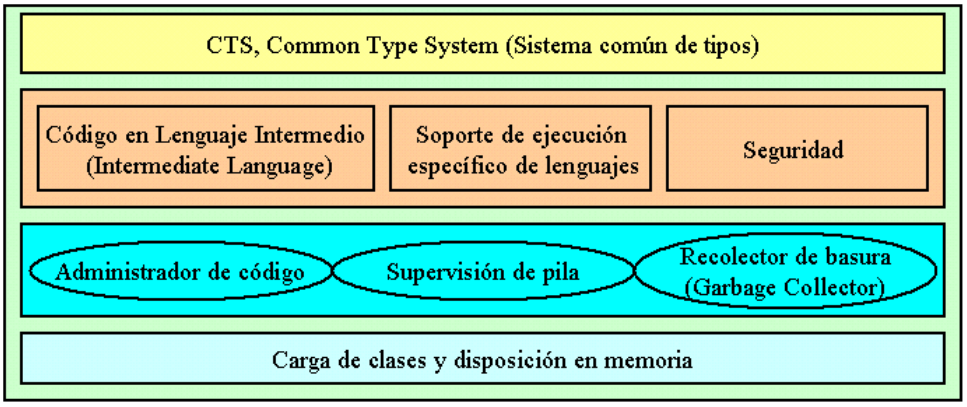
2.2.6.1. El CLR (Common Language Runtime)

El Entorno de Ejecución Común de Lenguajes o CLR (Common Language Runtime), representa el alma de .NET Framework y es el encargado de la ejecución del código de las aplicaciones.

A continuación, se enumeran algunas de las características de este componente de la plataforma:

- Proporciona un desarrollo de aplicaciones más sencillo y rápido gracias a que gran parte de las funcionalidades que tradicionalmente debía de crear el programador, vienen implementadas en el entorno de ejecución.
- Administra el código en tiempo de ejecución, en todo lo referente a su carga, disposición en memoria, recuperación de memoria no utilizada a través de un recolector de memoria, etc.
- Implementa características de gestión a bajo nivel (administración de memoria, por ejemplo), que en ciertos lenguajes eran labor del programador.
- Proporciona un sistema común de tipos para todos los lenguajes del entorno.
- Gestiona la seguridad del código que es ejecutado.
- Dispone de un diseño abierto a lenguajes y herramientas de desarrollo creadas por terceros fabricantes.
- Facilita enormemente la distribución e instalación de aplicaciones, ya que en teoría es posible instalar una aplicación simplemente copiando los ficheros que la componen en uno de los directorios del equipo en el que se vaya a ejecutar, eliminando los temibles conflictos de versiones entre librerías, problema conocido también con el nombre de *Infierno de las DLL* o *DLL Hell*.

Figura N° 03: Esquema de la organización interna del CLR.



Fuente: (Nuñez, 2002).

En los siguientes apartados, haremos una descripción de los elementos y características más destacables del CLR, que permitan al lector obtener una visión global del mismo, y de las ventajas de escribir programas para este entorno de ejecución.

2.2.6.2. El CTS (Common Type System)

El Sistema Común de Tipos o CTS (Common Type System), es el mecanismo del CLR que permite definir el modo en que los tipos serán creados y manipulados por el entorno de ejecución de .NET Framework.

Entre las funcionalidades que comprende, podemos destacar la integración de código escrito en diferentes lenguajes; optimización del código en ejecución; un modelo de tipos orientado a objeto, que soporta múltiples lenguajes; y una serie de normas que aseguran la intercomunicación entre objetos.

El sistema común de tipos (CTS a partir de ahora), como hemos indicado, permite definir o diseñar el modo en cómo el código de la aplicación

será ejecutado, pero no se encarga directamente de su ejecución; dicho de otro modo, el CTS le dice al CLR cómo quiere que sea ejecutado el código.

Un ejemplo de las ventajas del CTS, consiste en que desde un lenguaje como VB.NET, podemos instanciar un objeto de una clase escrita en otro lenguaje como C#; y al hacer una llamada a uno de los métodos del objeto, no es necesario realizar conversiones de tipos en los parámetros del método, funcionando todo de forma transparente.

2.2.6.3. Metadata (Metadatos)

Durante el diseño de .NET Framework, se hizo evidente que una aplicación necesitaba, además de su propio código ejecutable, información adicional sobre la propia aplicación, que pudiera ser utilizada por el entorno de ejecución para funcionalidades diversas. Para resolver este problema, se optó por incluir toda esta información complementaria dentro de la propia aplicación. A la información que va incluida en la aplicación pero que no forma parte del código ejecutable se le denomina metadatos, y con esta técnica obtenemos aplicaciones o componentes auto-descritos.

Los metadatos son creados por el compilador del lenguaje utilizado en cada caso y grabados dentro del fichero resultante (EXE o DLL) en formato binario, siendo el CLR el encargado de recuperarlos en el momento que los necesite. Algunos de los datos proporcionados por los metadatos de una aplicación son la descripción del ensamblado (trataremos los ensamblados posteriormente) junto a su versión, clave y tipos que lo componen (clases, interfaces, etc.).

2.2.6.4. Soporte Multi-Lenguaje

Uno de los puntos clave del CLR es que está diseñado para soportar múltiples lenguajes, permitiendo así unos elevados niveles de integración entre los mismos. Con tal motivo, .NET Framework proporciona los siguientes lenguajes con sus correspondientes compiladores para la escritura de aplicaciones:

- VB.NET.
- C#.
- C++ con Extensiones Administradas.
- JavaScript
- TypeScript

Por integración de lenguajes podemos definir algo tan poderoso como el hecho de escribir una clase en C#, y heredar de dicha clase desde VB.NET. Esto permite formar grupos de trabajo heterogéneos, en los que cada integrante del grupo, puede escribir el código de una aplicación en el lenguaje de su preferencia. Gracias a que el entorno de ejecución es común, y el código compilado no pasa directamente a código ejecutable puro, sino a un código intermedio (lo veremos más adelante), podemos crear nuestros programas en el lenguaje con el que nos sintamos más cómodos en cuanto a sintaxis y prestaciones, por ejemplo VB.NET; con la ventaja de que la velocidad de ejecución será muy parecida a la obtenida habiendo escrito el código en otro lenguaje en principio más rápido como C++ o C#.

2.2.7. Tecnología Asp.Net

La tecnología ASP .NET perteneciente a la plataforma de desarrollo de Microsoft, denominada .NET Framework o plataforma .NET. ASP .NET es la nueva versión de las páginas activas de servidor, más conocidas como Active Server Pages (ASP).

ASP .NET ofrece toda una nueva forma de desarrollar aplicaciones basadas en el entorno de Internet/Intranet, esta forma nueva de trabajar incluye una serie de novedades que no sólo son las correspondientes a una siguiente versión de ASP, sino que son las que se desprenden también de la nueva plataforma ofrecida por Microsoft, es decir, la plataforma .NET **(Nuñez, 2002).**

La plataforma .NET ofrece una serie de herramientas y tecnologías necesarias para construir y desarrollar aplicaciones Web, así pues, las páginas ASP .NET se van a ejecutar dentro del entorno de ejecución que nos facilita el .NET Framework. Podríamos decir que ASP .NET es una parte de la plataforma .NET, y es esta parte la que se va a tratar en el presente texto.

El código intermedio es una característica común que poseen todas las tecnologías englobadas en la estrategia .NET de Microsoft, a la que lógicamente pertenece ASP .NET. Las páginas ASP .NET cuando reciben la primera petición se compilan automáticamente a un lenguaje intermedio que es conocido como Common Language Runtime, es decir, es un lenguaje común al que compilan todos los lenguajes que utilicemos en nuestras páginas ASP .NET, generando el mismo código, ya sea Visual Basic .NET, C#. Gracias a esta característica podemos obtener grandes ventajas en lo que

a rendimiento en tiempo de ejecución se refiere, ya que la compilación de las páginas sólo se produce en la primera petición que se realiza sobre la página, o bien cuando el código fuente de la misma se ha modificado y necesita por lo tanto actualizarse, además el resultado de esta compilación permanece en caché para poder ser reutilizada.

El cambio que aporta ASP .NET es sobre todo en la forma de desarrollar aplicaciones Web, es decir, la forma en la que vamos a utilizar los distintos componentes y servicios que nos ofrece ASP .NET, y también cambia la forma de programar, como muestra diré que podemos tratar los eventos de cliente desde código de servidor, pudiendo crear formularios Web al estilo de las aplicaciones típicas de Visual Basic.

ASP .NET soporta diversos lenguajes de programación, entre los que se encuentra C#, los otros lenguajes que soporta directamente ASP .NET son Visual Basic .NET (VB7), como ya hemos dicho en este texto vamos a emplear C#, ya que es el nuevo lenguaje que ofrece Microsoft y pretende ser el lenguaje están para el .NET Framework. Apunto como dato curioso que el 90% de la herramienta de desarrollo Visual Studio .NET, es decir, la versión que nos ofrece Microsoft de Visual Studio para desarrollar en la plataforma .NET, ha sido desarrollado con el lenguaje C#, además este nuevo lenguaje está siendo sometido a estandarización por parte de ECMA, la misma entidad de normalización que llevó a cabo la estandarización de JavaScript

2.2.8. Net Core 2.0

NET Core 2.0 implementa .NET Standard 2.0, lo que facilita el uso del código compartido con los otros elementos de .NET Framework para web, siendo unos de las principales características el soporte multiplataforma en un solo paquete integrado, el dotnet corre en Windows, Linux y Mac OS. Los beneficios más importantes de la orientación de .NET Core 2.0 son las posibles mejoras de rendimiento, los beneficios de admitir .NET Standard 2.0 y el soporte para más plataformas.

.NET Core es la nueva generación de plataformas de desarrollo de Microsoft. Está escrita desde cero para hacerla multi-plataforma, ofrecer un rendimiento y escalabilidad sin precedentes, facilitar su trabajo en la nube... Y además es de código abierto (Open Source). Supone el mayor cambio en las tecnologías de desarrollo web de Microsoft desde sus inicios, y es el futuro a medio plazo.

Por encima de .NET Core se construyen otras bibliotecas de más alto nivel, especializadas en tareas concretas de desarrollo. Así, disponemos de ASP.NET Core y ASP.NET Core MVC. La primera se encarga de las funcionalidades transversales necesarias para el desarrollo web: infraestructura, cacheado, logging, autenticación, configuración, globalización. ASP.NET Core MVC funciona por encima de la anterior y nos proporciona el marco de trabajo para poder crear aplicaciones web con arquitectura Modelo-Vista-Controlador, con todo lo que ello supone.

Aunque ASP.NET Core MVC está basado conceptualmente en el “tradicional” ASP.NET MVC 5 y ambas comparten conceptos similares, son

muy distintas. Core MVC introduce nuevas características para facilitar el desarrollo de aplicaciones web bajo este paradigma.

Es decir: aunque ya conozcas ASP.NET MVC 5, esta nueva plataforma es muy diferente, e implica que deberás aprender desde cero ASP.NET Core, y actualizar muchos de los conocimientos y técnicas de MVC 5.

Los principales temas tratados a lo largo de sus más de 200 lecciones son los siguientes:

- Fundamentos y puesta en marcha de .NET Core, ASP.NET Core
- La arquitectura modular: pipeline, middlewares, paquetes, entornos de ejecución, startup Inyección de dependencias.
- Servicios de infraestructura: sesiones, caching, autenticación y seguridad, depuración, logging y trazas, gestión de archivos estáticos, settings/configuración, routing a bajo nivel.
- Creación de módulos middleware personalizados

Fundamentos de ASP.NET Core MVC

- La vista a fondo, herramientas, posibilidades y técnicas para la capa de presentación: tipologías, sintaxis razor, helpers, tag helpers, view components, inyección de dependencias en vistas...
- Programación de APIs HTTP y Servicios REST, CORS, herramientas para prueba de APIs
- Acceso a servicios HTTP/REST desde lado cliente y desde apps .NET
- Interacción entre cliente y servidor con AJAX en Core MVC
- Internacionalización, despliegue, testing de aplicaciones Core MVC,

páginas de error personalizadas.

➢ Aplicación de ejemplo BlogMachineCore: desarrollarás por tu cuenta y con el apoyo de José María Aguilar una aplicación real completa donde podrás ver implementadas la mayor parte de las técnicas explicadas y con la que afianzarás los conocimientos.

2.2.9. Lenguaje C#

Es el nuevo lenguaje de propósito general diseñado por Microsoft para su plataforma .NET. Sus principales creadores son Scott Wiltamuth y Anders Hejlsberg, éste último también conocido por haber sido el diseñador del lenguaje Turbo Pascal y la herramienta RAD Delphi.

Aunque es posible escribir código para la plataforma .NET en muchos otros lenguajes, C# es el único que ha sido diseñado específicamente para ser utilizado en ella, por lo que programarla usando C# es mucho más sencillo e intuitivo que hacerlo con cualquiera de los otros lenguajes ya que C# carece de elementos heredados innecesarios en .NET. Por esta razón, se suele decir que C# es el lenguaje nativo de .NET

La sintaxis y estructuración de C# es muy parecida a la de C++ o Java, puesto que la intención de Microsoft es facilitar la migración de códigos escritos en estos lenguajes a C# y facilitar su aprendizaje a los desarrolladores habituados a ellos. Sin embargo, su sencillez y el alto nivel de productividad son comparables con los de Visual Basic.

Un lenguaje que hubiese sido ideal utilizar para estos menesteres es Java, pero debido a problemas con la empresa creadora del mismo Sun,

Microsoft ha tenido que desarrollar un nuevo lenguaje que añadiese a las ya probadas virtudes de Java las modificaciones que Microsoft tenía pensado añadirle para mejorarlo aún más y hacerlo un lenguaje orientado al desarrollo de componentes.

En resumen, C# es un lenguaje de programación que toma las mejores características de lenguajes preexistentes como Visual Basic, Java o C++ y las combina en uno solo. El hecho de ser relativamente reciente no implica que sea inmaduro, pues Microsoft ha escrito la mayor parte de la BCL usándolo, por lo que su compilador es el más depurado y optimizado de los incluidos en el *.NET Framework SDK.*

2.2.10. Modelo Vista Controlador

Modelo Vista Controlador (MVC) es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Se trata de un modelo muy maduro y que ha demostrado su validez a lo largo de los años en todo tipo de aplicaciones, y sobre multitud de lenguajes y plataformas de desarrollo, la presentación y las acciones basadas en la información ingresada por el usuario quedan separados bajo estos tres componentes. **(Mancini, 2003)**

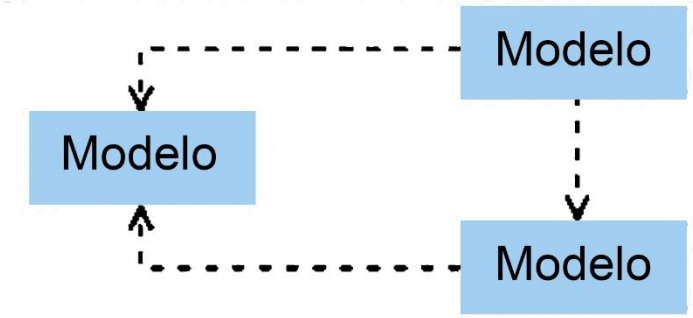
- **El Modelo:** En este ámbito se gestionan las comunicaciones entre el dominio de datos y dominio de aplicación atendiendo las consultas sobre su estado (realizadas con frecuencia desde la Vista) así como a las instrucciones de cambio de estado (usualmente desde el Controlador).

➢ **La Vista:** Este ámbito maneja la visualización de la información en un formato adecuado para el usuario y su interacción.

➢ **El Controlador:** Que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno. Este ámbito funciona interpretando las acciones del usuario sea por el teclado o el mouse, informando al modelo y/o a la vista sobre los cambios a realizarse en cada ámbito.

Como uno de los beneficios bajo este diseño destaca el soporte a múltiples vistas de una misma aplicación al mismo tiempo, aprovechando un único modelo de datos. La incorporación de nuevas vistas (por ejemplo, para dispositivos de plataformas diversas) no altera de sobremanera el comportamiento del modelo. En contraparte, adoptando este patrón trae consigo una fuerte dependencia hacia los eventos en la interfaz de usuario, incrementando la complejidad en la programación y control de tales acciones según las reglas de negocio. Asimismo, la codificación el modelo debe efectuarse tomando en cuenta la vista, para así evitar escenarios en los cuales un modelo al manejar múltiples cambios en el dominio pudiera sobrecargar a la vista con solicitudes de actualización, en tanto algunas vistas ralentizarían su ejecución quedando inoperativas ante tales sobrecargas.

Figura N° 04: Las interacciones en el patrón MVC.



Fuente: Patrón de arquitectura MVC (Mancini, 2003).

Las vistas y los controladores dependen del modelo, pero el modelo no depende ni de la vista ni del controlador. Esto permite que el modelo se pueda construir y probar independientemente de la presentación visual de la aplicación. Además, se pueden mostrar varias vistas de los mismos datos simultáneamente.

2.2.11. Lenguaje de Modelado Unificado UML

Para el desarrollo de software orientado a objetos no basta usar un lenguaje orientado a objetos. También se necesitará realizar un análisis y diseño orientado a objetos. El modelamiento visual es la clave para realizar el análisis. Desde los inicios del desarrollo de software han existido diferentes metodologías para hacer esto del modelamiento, pero sin lugar a duda, el Lenguaje de Modelamiento Unificado (UML) puso fin a la guerra de metodologías.

2.2.11.1. Definición de UML

UML (Unified Modeling Language) es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema software

orientado a objetos. UML entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas Concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables. La estandarización de un lenguaje de modelado es invaluable, ya que es la parte principal del proceso de comunicación que requieren todos los agentes involucrados en un proyecto informático. Si se quiere discutir un diseño con alguien más, ambos deben conocer el lenguaje de modelado y no así el proceso que se siguió para obtenerlo.

El UML es un lenguaje de modelado y no un método. La mayor parte de los métodos consisten, al menos en principio, en un lenguaje y en un proceso para modelar. El lenguaje de modelado es la notación (principalmente gráfica) de que se valen los métodos para expresar los diseños. El proceso es la orientación que nos dan sobre los pasos a seguir para hacer el diseño. El lenguaje de modelado es la parte más importante del método, es la clave para la comunicación; para poder analizar un diseño se necesita comprender el lenguaje de modelado; no el proceso que se siguió para lograr el diseño. **(Valles, 2010)**

2.2.11.2. Características del UML

- Desplegar los límites de un sistema, sus principales funciones mediante casos de uso y actores.
- Representar la estructura estática de un sistema usando diagramas de clases.
- Modelar los límites de un objeto con diagramas de estados.

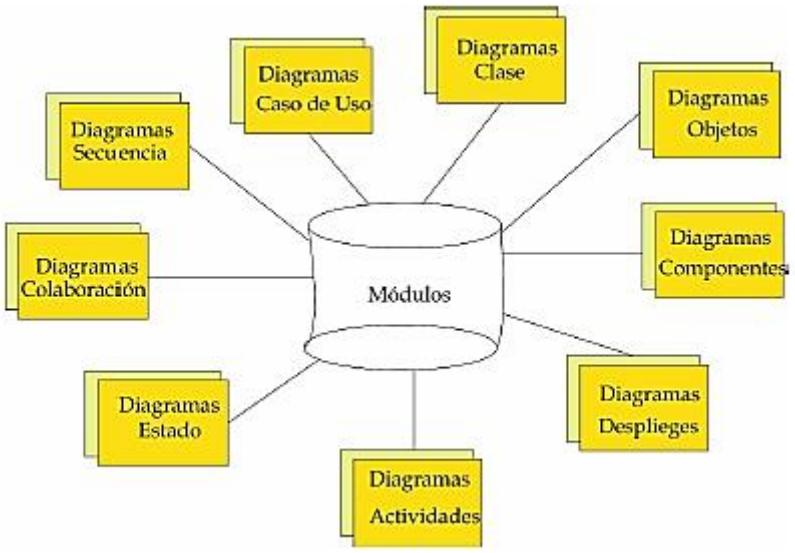
➢ Mostrar la arquitectura de la implementación física con diagramas componentes y de emplazamiento o despliegue.

2.2.11.3. Objetivos del UML

Los diagramas se utilizan para dar diferentes perspectivas del problema

según lo que nos interesa representar en un determinado momento, vale decir que en algunos casos no es necesario representar los nueve diagramas.

Figura N° 05: Objetivos del UML.



Fuente: (Valles, 2010).

2.2.11.4. Diagrama de Clases

El propósito de este diagrama es el de representar los objetos fundamentales del sistema, es decir los que percibe el usuario y con los que espera tratar para completar su tarea en vez de objetos del sistema o de un

modelo de programación. Un diagrama de clases está compuesto por los siguientes elementos:

➢ **Clase:** Es la unidad básica que encapsula toda la información de un Objeto (un objeto es una instancia de una clase). A través de ella podemos modelar el entorno en estudio. Los atributos o características de las clases pueden ser de tres tipos, según el grado de comunicación y visibilidad de ellos con el entorno, estos son:

Tabla N° 01: Tipos de atributos de la clase.

Nombre	Descripción
Públicos (+)	Indican que el atributo será visible tanto fuera como dentro de la clase, es decir, es accesible desde todos lados.
Privados (-)	Indican que el atributo solo será accesible desde dentro de la clase (solo sus métodos lo pueden acceder).
Protegidos (#)	Indican que el atributo no será accesible desde afuera de la clase, pero si podrá ser accesado por métodos de la clase además de la subclases que se deriven.

Fuente: Elaboración propia.

➢ **Relaciones entre clases:** Una relación es un término general que abarca los tipos específicos de conexiones lógicas que se pueden encontrar en los diagramas de clases y objetos. UML presenta las siguientes relaciones.

➢ **Herencia:** Indica que una subclase hereda los métodos y atributos especificados por una Súper Clase, por ende la Subclase además de poseer

sus propios métodos y atributos, poseerá las características y atributos visibles de la Súper Clase.

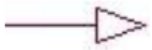
- **Agregación**: Es un tipo especial de asociación que representa una relación estructural entre las clases donde el llamado agregado indica el todo y el componente es una parte del mismo.

- **Composición:** Es un tipo de agregación donde la relación de posesión es tan fuerte como para marcar otro tipo de relación.

- **Asociación**: Relación estructural que describe un conjunto de conexiones entre objetos de forma bidireccional.

- **Dependencia:** Es una relación de uso, es decir una clase usa a otra, que la necesita para su cometido. Se representa con una flecha discontinua que va desde la clase utilizadora a la clase utilizada. Con la dependencia se muestra que un cambio en la clase utilizada puede afectar el funcionamiento de la clase utilizadora, pero no al contrario.

Tabla N° 02: Símbolo de relaciones de clases.

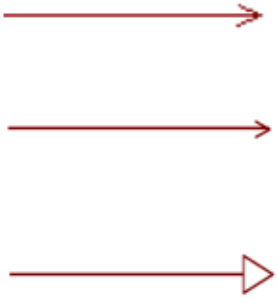
Nombre	Símbolo
Herencia	
Agregación	
Composición	
Asociación	
Dependencia	

Fuente: Elaboración Propia.

2.2.11.5. Diagrama de Casos de Uso

Un caso Diagrama de Casos de Uso puede existir tanto a nivel del Modelo de Negocio como en el nivel de Modelo de Construcción del Software. A nivel de Negocio muestra el Caso de Uso del Negocio relacionado con los actores internos y externos de negocio. A nivel de Sistema muestra la funcionalidad total del Sistema Software que se construye. El Diagrama de Casos de Uso a nivel de Sistema permite definir los privilegios del Sistema por actor, teniendo en cuenta aspectos de auditoría al considerar el módulo de identificación, como obligatorio.

Tabla N° 03: Elementos de casos de uso.

Nombre	Símbolo	Descripción
actor		Es un rol que un usuario juega con respecto al sistema. Es importante destacar el uso de la palabra rol, pues con esto se especifica que un Actor no necesariamente representa a una persona en particular, sino más bien la labor que realiza frente al sistema.
Caso de uso		Es una operación y/o tarea específica que se realiza tras una orden de algún agente externo, sea desde una petición de un actor o bien desde la invocación desde otro caso de uso.
Relaciones		**Asociación:** Es el tipo de relación más básica que indica la invocación desde un actor o caso de uso a otra operación. **Dependencia:** Es una forma muy particular de relación entre clases, en la cual una clase depende de otra, es decir, se crea. **Generalización:** Este tipo de relación es uno de los más utilizados, cumple una doble función dependiendo de su estereotipo, que puede ser de Uso o de Herencia.

Fuente: Elaboración Propia.

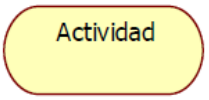
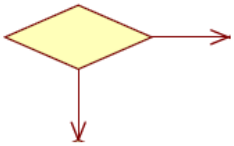
2.2.11.6. Diagrama de Objetos

Muestra un conjunto de objetos (instancias de las clases) y sus relaciones. Modelan las instancias de elementos contendidos en los diagramas de clases, es decir las ocurrencias de cada elemento que constituye una clase, a cada uno de estos elementos se les llama objetos. Son como fotos instantáneas de los diagramas de clases. **(Valles, 2010)**

2.2.11.7. Diagrama de Actividades

Muestra la realización de operaciones para conseguir un objetivo. Presentan una visión simplificada de lo que ocurre en un proceso, mostrando los pasos que se realizan. Los diagramas de actividad, son una extensión de los diagramas de estado. Los diagramas de estado resaltan los estados y muestran las actividades que dan lugar a cambios de estado, mientras que los diagramas de actividad, resaltan justamente las actividades. Comúnmente los diagramas de actividad se utilizan en dos formas. En el modelado de flujos de trabajo, haciendo hincapié en las actividades tal y como son vistas por los actores que colaboran con el sistema, esto es modelando procesos de negocios. En el modelado de una operación, utilizando los diagramas de actividad como diagramas de flujo para mostrar detalles de un algoritmo, haciendo amplio uso de las condiciones y modelado de procesos concurrentes. **(Valles, 2010)**

Tabla N° 04: Elementos de diagrama de actividades.

Nombre	Símbolo	Descripción
Acción		Nodo de Actividad primitiva ejecutable de asignación o computación.
Nodo de Inicio		Nodo de control que indica el inicio de un flujo de control cuando una actividad es invocada.
Nodo fin de actividad		Nodo de control que indica el fin de datos los flojos dentro de una muestra el fin de la actividad.
Flujo de control		Eje de actividades para flujo de control que conecta dos acciones. Usado para indicar secuencia.
Nodo de decisión		Nodo de control que selecciona entre dos o más flujos de salida.

Fuente: Elaboración propia.

2.2.11.8. Diagrama de Secuencia

Un diagrama de Secuencia muestra una interacción ordenada según la secuencia temporal de eventos. En particular, muestra los objetos participantes en la interacción y los mensajes que intercambian ordenados según su secuencia en el tiempo. El eje vertical representa el tiempo, y en el eje horizontal se colocan los objetos y actores participantes en la interacción, sin un orden prefijado. Cada objeto o actor tiene una línea vertical, y los mensajes se representan mediante flechas entre los distintos objetos. El tiempo fluye de arriba abajo. Se pueden colocar etiquetas (como restricciones de tiempo, descripciones de acciones, etc.) bien en el margen izquierdo o bien junto a las transiciones o activaciones a las que se refieren.

2.2.11.9. Diagrama de Colaboración

Un Diagrama de Colaboración muestra una interacción organizada basándose en los objetos que toman parte en la interacción y los enlaces entre los mismos (en cuanto a la interacción se refiere). A diferencia de los Diagramas de Secuencia, los Diagramas de Colaboración muestran las relaciones entre los roles de los objetos. La secuencia de los mensajes y los flujos de ejecución concurrentes deben determinarse explícitamente mediante números de secuencia. Los diagramas de colaboración permiten mostrar mejor como se vinculan los objetos, a cambio de hacer más difícil observar el orden de ejecución, pues enumeran los mensajes en lugar de mostrar al tiempo como una dimensión, tal como lo hacen los diagramas de secuencia. **(Valles, 2010)**

2.2.11.10. Diagrama de Estados

Un Diagrama de Estados muestra la secuencia de estados por los que pasa bien un caso de uso, bien un objeto a lo largo de su vida, o bien todo el sistema. En él se indican qué eventos hacen que se pase de un estado a otro y cuáles son las respuestas y acciones que genera. En cuanto a la representación, un diagrama de estados es un gráfico cuyos nodos son estados y cuyos arcos dirigidos son transiciones etiquetadas con los nombres de los eventos. Capturan los cambios de estado que sufren los objetos en respuesta a eventos. Los diagramas de clases y de objetos correspondientes, sólo muestran los aspectos estáticos, pero no muestran como son afectados los objetos cuando ocurre algo. Sin embargo, estos comportamientos tienen que implementarse mediante software y representarlos en algún sitio, asegura que los desarrolladores no adivinen el

comportamiento y produzcan software que satisfaga los requerimientos.

2.2.11.11. Diagrama de Componente

Los diagramas de componentes permiten visualizar las partes de un sistema, mostrando las diversas formas en que pueden ensamblarse para construir ejecutables. Un diagrama de componentes muestra las dependencias entre componentes físicos de software, tales como archivos de código fuente, binarios, de configuración, de instalación y desinstalación, ejecutables, tablas, etc. Los diagramas de componentes modelan la vista estática de los sistemas, es decir sólo los componentes y sus conexiones y no como funcionan.

2.2.11.12. Diagrama de Despliegue

El diagrama de despliegue, modela la topología del hardware sobre el cual correrá nuestra aplicación y nos indica en donde se ejecutará cada uno de nuestros componentes; muestra las relaciones físicas entre los componentes de software y el hardware de nuestro sistema. Los diagramas de despliegue muestran la forma en que físicamente lucirá nuestro sistema, sólo deben mostrarse los nodos y componentes que utilizarán en su versión ejecutable. El término original para estos diagramas es deployment diagram que en nuestro idioma ha sido traducido como diagramas de distribución, emplazamiento, implantación o despliegue. **(Valles, 2010)**

2.2.12. Pruebas o Métricas del Software

Las métricas de software tienen un papel decisivo en la obtención de un producto de alta calidad, porque determinan mediante estadísticas basadas en la experiencia, el avance del software y el cumplimiento de parámetros requeridos. Siempre habrá elementos cualitativos para la creación de software. El problema estriba en que la valoración cualitativa puede no ser suficiente. Un ingeniero del software necesita criterios objetivos para guiarse en el diseño de datos, de la arquitectura, de las interfaces y de los componentes. El verificador necesita una referencia cuantitativa que le ayude en la selección de los casos de prueba y de sus objetivos. Las métricas técnicas facilitan una base para que el análisis, diseño, codificación y prueba puedan ser conducidos más objetivamente y valorados más cuantitativamente.

La palabra métrica, es muy común asociarla con las palabras medición y medida, aunque estas tres son distintas. La medición “es el proceso por el cual los números o símbolos son asignados a atributos o entidades en el mundo real tal como son descritos de acuerdo a reglas claramente definidas”.

La prueba de software es un elemento crítico para la garantía de calidad del software y representa una visión final de las especificaciones del diseño y la codificación.

El objetivo es diseñar pruebas que sistemáticamente saquen a la Luz diferentes clases de errores, haciéndolos con la menor cantidad de tiempo y esfuerzo. Las normas que puedan servir para objetos de prueba son:

➢ La prueba es un proceso de ejecución de un programa con la intensión de descubrir un error.

➢ Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces.

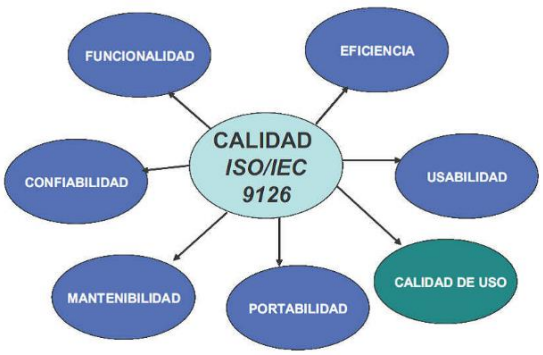
➢ Una prueba tiene éxito si se descubre un error no detectado hasta entonces.

2.2.13. Norma de Evaluación de ISO/IEC 9126

Esta norma internacional fue publicada en 1992, la cual es usada para la evaluación de la calidad de software, la norma ISO/IEC 9126 permite especificar y evaluar la calidad del software desde diferentes criterios asociados con adquisición, requerimientos, desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento de la calidad y auditoria de software.

El objetivo de esta norma ISO/IEC 9126 es proporcionar un modelo de calidad que sirve como elemento central en un proceso de evaluación. El modelo de calidad que propone la norma puede aplicarse a cualquier tipo de software incluido el desarrollo para el ámbito educativo. **(Borbón, 2013)**

Figura N° 06: Norma de evaluación ISO/IEC 9126.



Fuente: Estándar ISO/IEC 9126.

Características Generales

a. Funcionalidad: Es la capacidad del software de cumplir y proveer las funciones para satisfacer las necesidades explícitas e implícitas cuando es utilizado en condiciones específicas se divide en 5 criterios:

- **Adecuación:** La capacidad del software para proveer un adecuado conjunto de funciones que cumplan las tareas y objetivos especificados por el usuario.

- **Exactitud:** La capacidad del software para hacer procesos y entregar los resultados solicitados con precisión o de forma esperada.

- **Interoperabilidad:** La capacidad del software de interactuar con uno o más sistemas específicos.

- **Seguridad:** La capacidad del software para proteger la información y los datos de manera que los usuarios o los sistemas no autorizados no puedan acceder a ellos para realizar operaciones, y la capacidad de aceptar el acceso a los datos de los usuarios o sistemas autorizados

- **Conformidad de la funcionalidad:** La capacidad del software de cumplir los estándares referentes a la funcionalidad.

b. Confiabilidad: Es la capacidad para asegurar un nivel de funcionamiento adecuado cuando es utilizado en condiciones específicas, se divide en 4 criterios:

- **Madurez:** La capacidad que tiene el software para evitar fallas cuando encuentra errores. Ejemplo, la forma como el software advierte al usuario

cuando realiza operaciones en la unidad de diskett vacia, o cuando no encuentra espacio suficiente el disco duro donde esta almacenando los datos.

- **Tolerancia a errores:** La capacidad que tiene el software para mantener un nivel de funcionamiento en caso de errores.

- **Recuperabilidad:** La capacidad que tiene el software para restablecer su funcionamiento adecuado y recuperar los datos afectados en el caso de una falla.

- **Conformidad de la fiabilidad:** La capacidad del software de cumplir a los estándares o normas relacionadas a la fiabilidad.

c. Usabilidad: Es la capacidad del software de ser entendido, aprendido, y usado en forma fácil y atractiva. La usabilidad está determinada por los usuarios finales y los usuarios indirectos del software, dirigidos a todos los ambientes, a la preparación del uso y el resultado obtenido, se divide en 5 criterios:

- **Entendimiento:** La capacidad que tiene el software para permitir al usuario entender si es adecuado, y de una manera fácil como ser utilizado para las tareas y las condiciones particulares de la aplicación. En este criterio se debe tener en cuenta la documentación y de las ayudas que el software entrega.

- **Aprendizaje:** La forma como el software permite al usuario aprender su uso. También es importante considerar la documentación.

- **Operabilidad:** La manera como el software permite al usuario operarlo y controlarlo.

- **Atracción:** La presentación del software debe ser atractiva al usuario. Esto se refiere a las cualidades del software para hacer más agradable al usuario, ejemplo, el diseño gráfico.

- **Conformidad de uso:** La capacidad del software de cumplir los estándares o normas relacionadas a su usabilidad.

d. Eficiencia: La eficiencia del software es la forma del desempeño adecuado, de acuerdo al número de recursos utilizados según las condiciones planteadas, se divide en 3 criterios:

- **Comportamiento de tiempos:** Los tiempos adecuados de respuesta y procesamiento, el rendimiento cuando realiza su función en condiciones específicas. Ejemplo, ejecutar el procedimiento más complejo del software y esperar su tiempo de respuesta, realizar la misma función, pero con más cantidad de registros.

- **Utilización de recursos:** La capacidad del software para utilizar cantidades y tipos adecuados de recursos cuando este funciona bajo requerimientos o condiciones establecidas. Ejemplo, los recursos humanos, el hardware, dispositivos externos.

- **Conformidad de eficiencia:** La capacidad que tiene el software para cumplir con los estándares o convenciones relacionados a la eficiencia.

e. Capacidad de mantenimiento: Es la cualidad que tiene el software para ser modificado. Incluyendo correcciones o mejoras del software, a cambio en el entorno, y especificaciones de requerimientos funcionales, se divide en 5 criterios:

➢ **Capacidad de ser analizado:** La forma como el software permite diagnósticos de deficiencias o causas de fallas, o la identificación de partes modificadas.

➢ **Cambiabilidad:** La capacidad del software para que la implementación de una modificación se pueda realizar, incluye también codificación, diseño y documentación de cambios.

➢ **Estabilidad:** La forma como el software evita efectos inesperados para modificaciones del mismo.

➢ **Facilidad de prueba:** La forma como el software permite realizar pruebas a las modificaciones sin poner el riesgo los datos.

➢ **Conformidad de facilidad de mantenimiento:** La capacidad que tiene el software para cumplir con los estándares de facilidad de mantenimiento.

f. Portabilidad: La capacidad que tiene el software para ser trasladado de un entorno a otro. Esta referido por los siguientes sub atributos, se divide en 5 criterios:

➢ **Adaptabilidad:** Es como el software se adapta a diferentes entornos especificados (hardware o sistemas operativos) sin que implique reacciones negativas ante el cambio. Incluye la escalabilidad de capacidad interna (Ejemplo: Campos en pantalla, tablas, volúmenes de transacciones, formatos de reporte, etc.).

➢ **Facilidad de instalación:** La facilidad del software para ser instalado en un entorno específico o por el usuario final.

- **Coexistencia:** La capacidad que tiene el software para coexistir con otro o varios software, la forma de compartir recursos comunes con otro software o dispositivo.

- **Reemplazabilidad**: La capacidad que tiene el software para ser remplazado por otro software del mismo tipo, y para el mismo objetivo. Ejemplo, la reemplazabilidad de una nueva versión es importante para el usuario, la propiedad de poder migrar los datos a otro software de diferente proveedor.

- **Conformidad de portabilidad:** La capacidad que tiene el software para cumplir con los estándares relacionados a la portabilidad.

g. Calidad de uso: Es la calidad del software que el usuario final refleja, la forma como el usuario final logra realizar los procesos con satisfacción, eficiencia y exactitud, se divide en 4 criterios:

- **Eficacia:** La capacidad del software para permitir a los usuarios finales realizar los procesos con exactitud e integridad.

- **Productividad:** La forma como el software permite a los usuarios emplear cantidades apropiadas de recursos, en relación a la eficacia lograda en un contexto específico de uso. Para una empresa es muy importante que el software no afecte a la productividad del empleado

- **Seguridad:** Se refiere al que el Software no tenga niveles de riesgo para causar daño a las personas, instituciones, software, propiedad intelectual o entorno. Los riesgos son normalmente el resultado de deficiencias en la funcionalidad, fiabilidad, usabilidad o facilidad de mantenimiento.

➢ **Satisfacción:** La satisfacción es la respuesta del usuario a la interacción con el software, e incluye las actitudes hacia el uso del mismo. A continuación, se describe un cuadro donde podemos resumir las características y cada uno de sus atributos, este cuadro le ayudara a visualizar el proceso de evaluación.

2.3. Definición de Términos Básicos

➢ **Sistema**

Conjunto de componentes interrelacionados e interactuantes para llevar a cabo una misión conjunta. Admite ciertos elementos de entrada y produce ciertos elementos de salida en un proceso organizado.

➢ **Sistema Web**

También conocidos como "Aplicaciones Web", son aquellos que están creados e instalados, no sobre una plataforma o sistemas operativos (Windows, Linux) sino que se alojan en un servidor en Internet o sobre una intranet (red local). Su apariencia es muy equivalente a páginas Web, que normalmente vemos. Realmente los sistemas web tienen funcionalidades muy poderosas que resuelven o brindan respuestas a casos particulares. Los Sistemas Web se pueden utilizar desde cualquier navegador y por supuesto independientemente del sistema operativo. (**Baez, 2012)**

➢ **Integración**

Integración es el acto de unir, incorporar y/o entrelazar partes para que forme parte de un todo.

➢ **Circulación terrestre**

Es aquel cuyas redes (carreteras, caminos, ferrocarriles) se extienden por la superficie de la tierra. La gran mayoría de transportes terrestres se realizan sobre ruedas que podrían ser automóviles, autobuses, motocicletas, camiones, etc.

➢ **Documento**

Medios evidentes de prueba, siendo insustituibles cuando así lo dispone la ley en determinadas circunstancias y condiciones, lo cual se debe a que es el testimonio humano existente y permanente que mantiene el vínculo con el pasado, señalando cómo ocurrieron los hechos y se manifestaron externamente como todo otro documento en entidades públicas y privadas**. (Calvo, 2009)**

➢ **Trámite documentario**

Permite tener el control de la ubicación física de la documentación que llega, fluye y se genera en una organización. Función de realizar tramite progresivo que se le hace al escrito en papel u otro tipo de soporte con que se prueba o acredita una cosa, como un título, una profesión, un contrato. **(Layout, 2009)**

➢ **Reincidencia**

Se denomina reincidencia a la repetición de un cierto vicio.

- **Aceleración**

Es el proceso en donde la velocidad del tiempo cambia.

- **Implementación**

En desarrollo de Sistemas Informáticos, la implementación es la etapa donde efectivamente se programa el sistema.

CAPÍTULO III

MATERIALES Y MÉTODOS

3.1. Ubicación Geográfica del Estudio

El desarrollo del presente trabajo se llevó acabo en la Oficina de Circulación Terrestre de la Dirección Regional de Transportes Puno.

3.2. Población y Muestra del Estudio

3.2.1. Población

La población para efectos de la investigación del proyecto presentado lo constituyen 130 usuarios que realizaron procesos y tramites en la Oficina de Circulación Terrestre de la Dirección Regional de Transportes Puno.

3.2.2. Muestra

Se tomará el método de muestreo no probabilístico a criterio del investigador en este sentido la muestra estará conformada por 130 usuarios que realizaron procesos y tramites en la Oficina de Circulación Terrestre de la Dirección Regional de Transportes Puno.

3.3. Operacionalización de Variables

Tabla N° 05: Operacionalización de variables.

VARIABLES	INDICADORES	INDICE
Variable independiente	Diseño de interfaz del sistema	Muy bueno Bueno Regular
Funcionamiento del sistema web integrado	Servicio que ofrece el sistema	Eficiente Medianamente eficiente Deficiente
	Acceso al sistema	Bueno Regular Pesimo
Variable dependiente Tiempo de atención	Tiempo de atención de procesos y tramites	Minutos de atención

Fuente: Elaboración propia.

3.4. Métodos de Recopilación de Datos

La recolección de datos para el presente trabajo de investigación se obtuvo a través de la encuesta a los usuarios que realizaron procesos y tramites en la Oficina de Circulación Terrestre de la Dirección Regional de Transportes Puno.

Figura N° 07: Diseño de investigación.

Fuente: Elaboración propia.

Para la recopilación de datos de usabilidad del sistema se utilizó los instrumentos que fueron aplicados en los usuarios del sistema de información en la Universidad Nacional del Altiplano, como:

- La ficha de evaluación del ISO-9126 que ve la calidad de software.
- Encuesta al usuario.
- Encuesta al administrador.

3.5. Método de Análisis de Datos

Para el análisis del resultado de la encuesta respecto al tiempo de demora antes y después de la implementación del sistema se utilizó la prueba Z – diferencia de promedios.

i. Planteamiento de hipótesis

H_o : No hay diferencia significativa en el tiempo promedio de procesos y tramites antes y después de implementar el sistema web integrado.

H_1 : El tiempo promedio de procesos y tramites antes de la implementación del sistemas es mayor al tiempo promedio después de la implementación del sistema web integrado.

CAPÍTULO IV

RESULTADOS Y DISCUSIÓN

4.1. Presentación de Resultados

4.1.1. Análisis

4.1.1.1. Ámbito del Problema

El Sistema web integrado para la Oficina de Circulación Terrestre de la Dirección Regional de Transportes Puno, se desarrolló principalmente para mejorar la atención a los usuarios, mediante el control de procesos y trámites de los documentos ingresados, al mismo tiempo mejorar la gestión administrativa mediante la interconexión de oficinas para el registro de documentos y el estado final de los mismos a través del seguimiento mediante el software. La creciente demanda por parte de los usuarios en estos últimos años ha originado al personal administrativo mayor carga laboral, una excesiva cola y un desorden por parte de los usuarios la cual requiere de nuevas tecnologías para agilizar la atención de los procesos y tramites.

Lo expuesto anteriormente fueron las evidencias para la implementación

de un Sistema web integrado para brindar una atención más eficiente y oportuna.

4.1.2. Análisis de Requisitos del Sistema

El desarrollo del software no consiste únicamente en la codificación, aunque sea una de las fases más importantes, sino que existen múltiples tareas que deben ser desarrolladas si se desea un sistema robusto y de fácil mantenimiento y con garantías de finalización en el tiempo y costo estimados.

Para determinar las tareas es necesario asegurar un proceso de desarrollo de software, adaptándolo según se estime conveniente al proyecto en el cual se aplique. Una de estas tareas es permitir conocer en todo momento el estado en el que se encuentra el sistema durante el desarrollo.

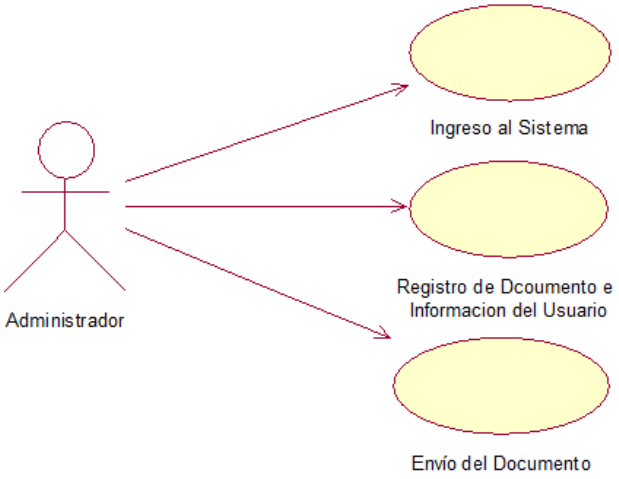
4.2. Modelamiento de la Plataforma Mediante UML

Para entender el modelamiento en el diseño de la plataforma se ha utilizado el lenguaje unificado de Modelado UML, la misma que nos ha permitido obtener los diagramas de casos de uso y secuencia.

4.3. Modelado de Requisitos

4.3.1. Diagrama de Caso de Uso

Figura N° 08: Caso de Uso General de Sistema.



Fuente: Elaboración Propia.

Mediante este diagrama representaremos al sistema en su forma general.

Casos de uso: Ingresar al sistema, registro de documentos e información del usuario, envió de documento hacia otra oficina.

Actores: Administrador del sistema.

Descripción en forma general de los casos de uso

1. Ingresar al sistema: Acceso del Administrador al sistema con un nombre de usuario y una contraseña.

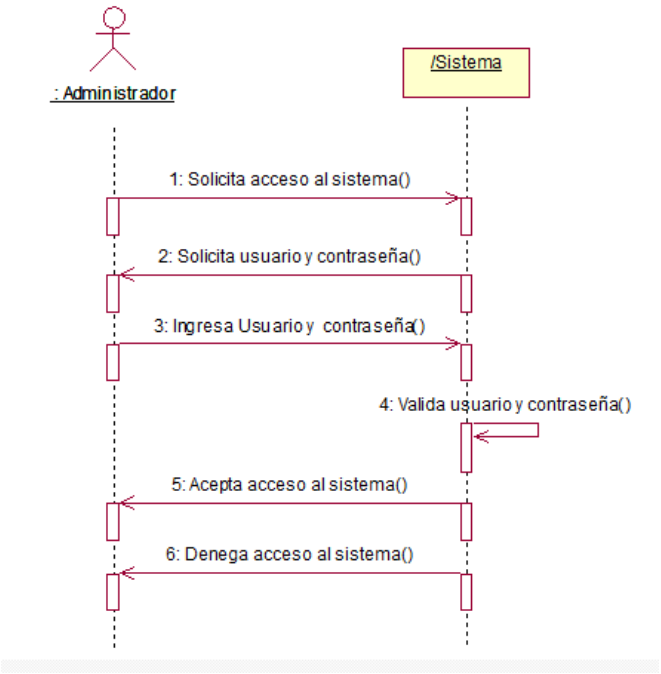
2. Registro de Documentos e Información del usuario: Registra información del documento e información del usuario que realizara el tramite documentario.

3. Envió de documento: Envía la información del documento a la oficina correspondiente.

4.4. Diagrama de Secuencia

Interfaz para Ingreso al Sistema: El operador solicita acceso al sistema, el sistema solicita usuario y contraseña, el operador ingresa el usuario y contraseña, luego el sistema valida los datos ingresados y en efecto el sistema acepta o deniega el acceso

Figura N° 09: Diagrama de secuencia de ingreso al sistema.

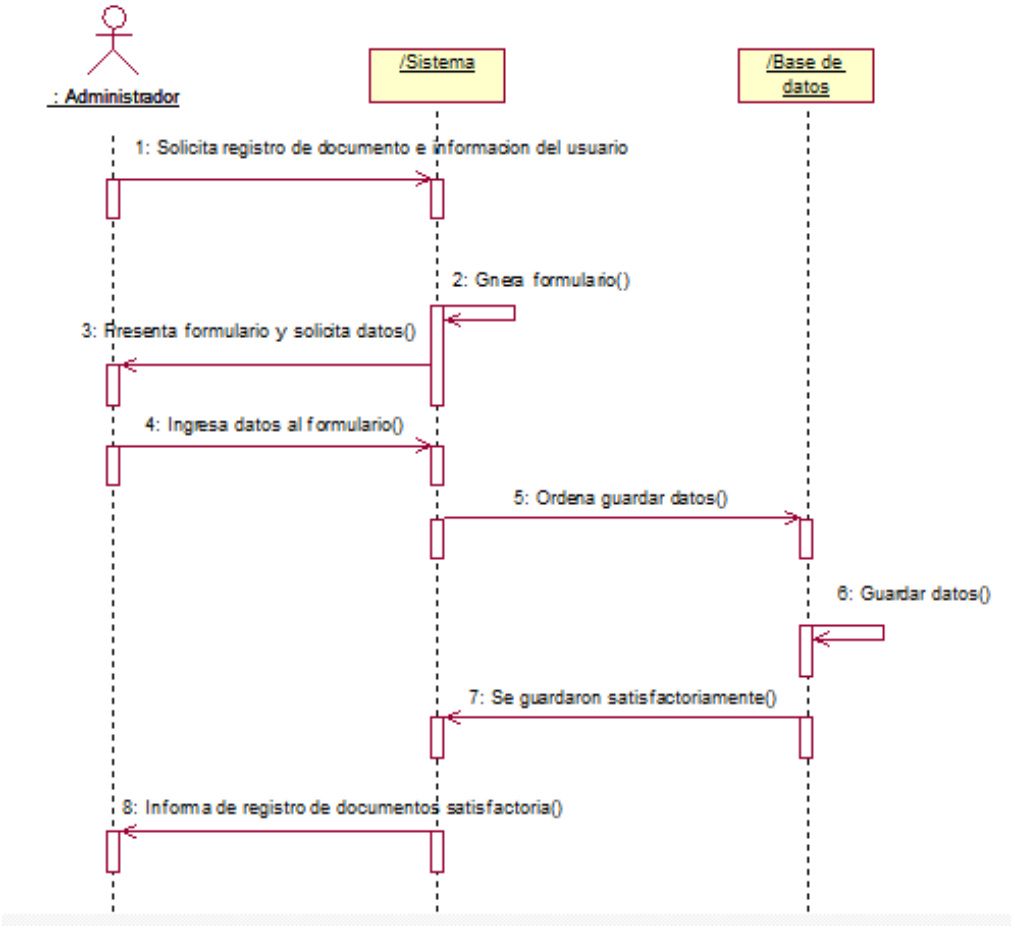


Fuente: Elaboración Propia.

Interfaz para Registro de Documentos e información del usuario: El administrador solicita registro de documentos e información del usuario, el sistema genera un formulario y a su vez solicita datos, el administrador ingresa datos al sistema y este los guarda en la Base de Datos, la Base de Datos

informa al sistema que se guardaron satisfactoriamente, y el sistema informa al administrador que el registro se realizó de forma satisfactoria.

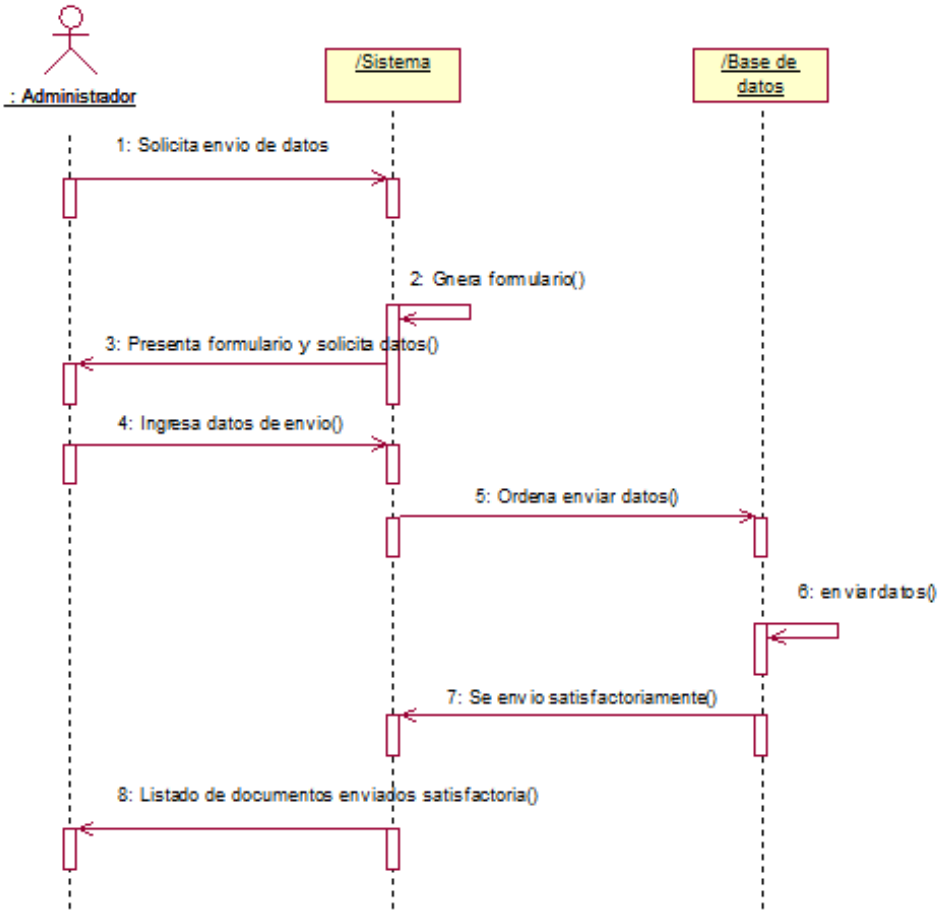
Figura N° 10: Diagrama de secuencia de registro de Documentos e información del usuario.



Fuente: Elaboración Propia.

Interfaz Envío de Documentos: El administrador solicita envío de documentos, el sistema genera un formulario y a su vez presenta y solicita datos, el administrador ingresa datos de envío al sistema y este los envía a la Base de Datos, la Base de Datos informa al sistema que se envió satisfactoriamente, y el sistema informa al administrador que el envío se realizó de forma satisfactoria.

Figura N° 11: Diagrama de secuencia envío de documentos.

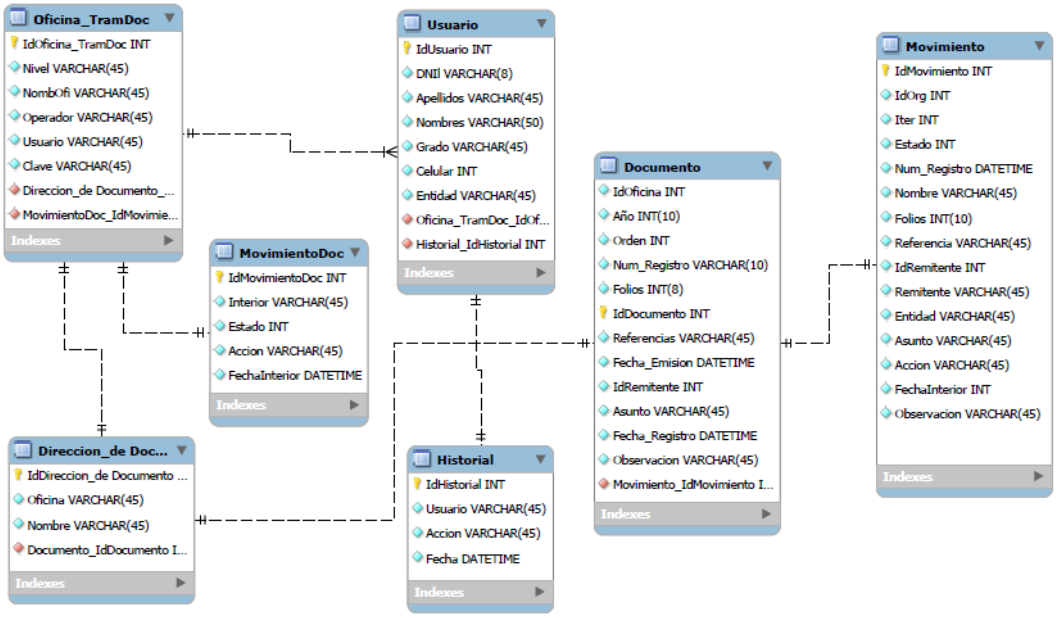


Fuente: Elaboración Propia.

4.5. Diseño de la Base de Datos

El diseño de la Base de Datos es fundamental para empezar con el desarrollo de cualquier sistema informático y así asegurar el óptimo funcionamiento del mismo.

Figura N° 12: Diagrama de Base de Datos de Trámite Documentario.



Fuente: Elaboración Propia

4.6. Pruebas del Software

4.6.1. Prueba de Software ISO – 9126

Después de aplicar la ficha de evaluación ISO-9126 a los 13 Administradores del sistema implementado.

Tabla N° 06: Decisiones ISO- 9126.

CLASIFICACIÓN	INTERVALO	DECISIÓN
A) Inaceptable	[27 - 54 >	
B) Mínimamente aceptable	[54 - 81 >	
C) Aceptable	[81 - 95 >	
D) Cumple los requisitos	[95 - 122 >	108
F) Excede los requisitos	[122 – 135]	

Fuente: Tablas de validación de software.

Según los resultados el promedio de 112, 102, 111, 115, 99, 119, 98, 114, 100, 115, 106, 104 y 112 nos resulta 108, indicando que cumple con los requisitos según ISO-9126.

Decisión:

De acuerdo a los resultados de la calidad de software se concluyó que el sistema web integrado para los procesos y trámites en la Oficina de Circulación Terrestre de la Dirección Regional de Transportes Puno, cumple los requisitos con un promedio de 108 puntos de un total de 135 puntos que se considera en la tabla de decisiones del ISO-9126.

4.7. Gráficos de Satisfacción de Usuario

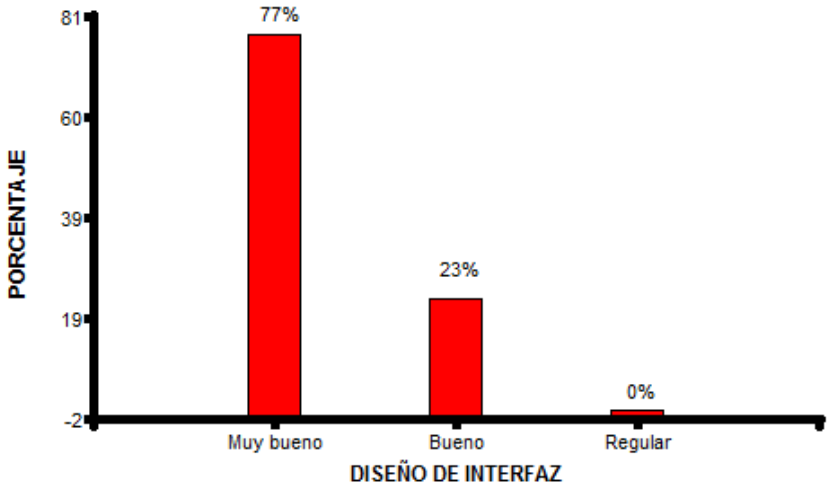
Resultados del Diseño de la Interfaz del Sistema Web

Tabla N° 07: Diseño de interfaz.

DISEÑO DE INTERFAZ DEL SISTEMA WEB	FRECUENCIA	PORCENTAJE (%)
Muy bueno	10	77%
Bueno	3	23%
Regular	0	0%

Fuente: Elaboración del grupo.

Gráfico N° 04: Diseño de interfaz.



Fuente: Elaboración del grupo.

De acuerdo a la encuesta aplicada a los administradores de la oficina de circulación terrestre se observa que el 77% de los administradores considera que el diseño de interfaz del sistema web es Muy bueno y el 23% del total considera que el diseño de interfaz del sistema web es bueno.

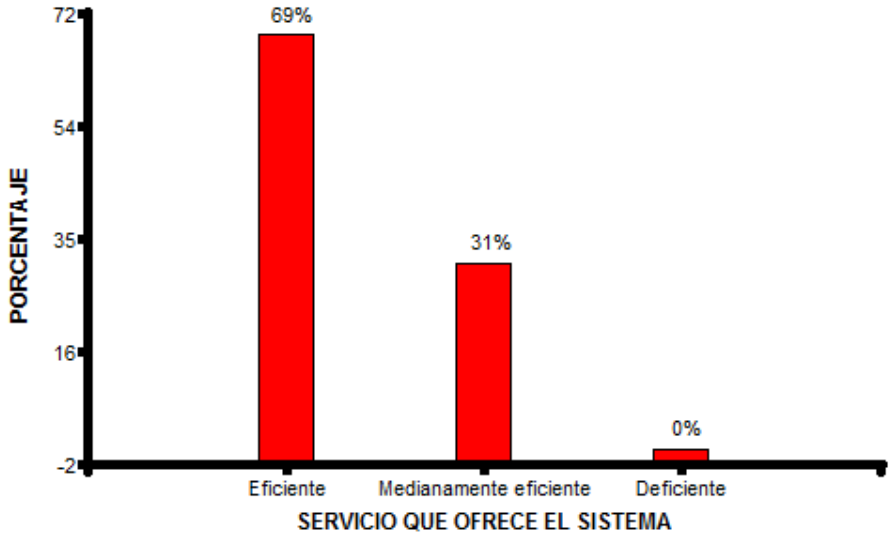
Resultados de Servicio que Ofrece el Sistema Web

Tabla N° 08: Servicio que ofrece el sistema.

SERVICIO QUE OFRECE EL SISTEMA	FRECUENCIA	PORCENTAJE (%)
Eficiente	9	69%
Medianamente eficiente	4	31%
Deficiente	0	0%

Fuente: Elaboración del grupo.

Gráfico N° 05: Servicio que ofrece el sistema.



Fuente: Elaboración del grupo.

De acuerdo a la encuesta aplicada a los administradores de la oficina de circulación terrestre se observa que el 69% de los administradores considera que el Servicio que ofrece el sistema web es eficiente y el 31% del total considera que el Servicio que ofrece el sistema web medianamente eficiente.

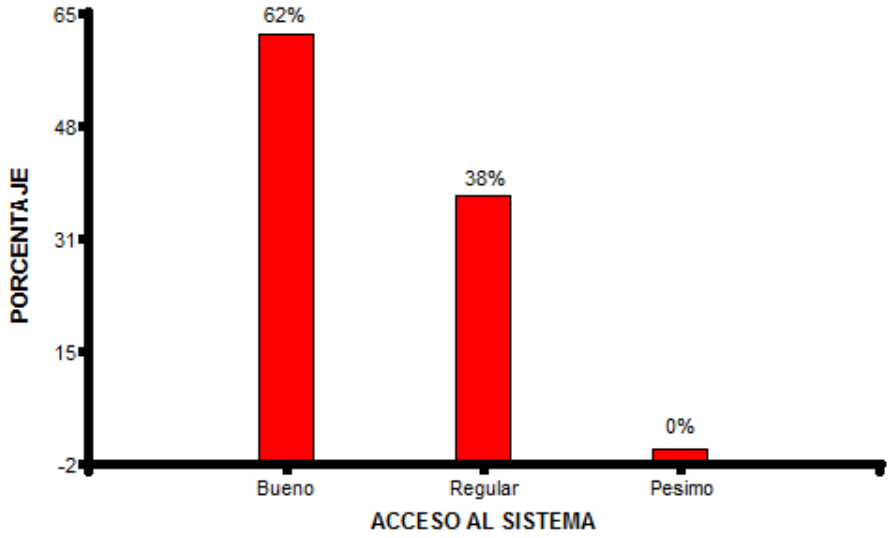
Resultados del Acceso al Sistema

Tabla N° 09: Acceso al sistema.

ACCESO AL SISTEMA	FRECUENCIA	PORCENTAJE (%)
Bueno	8	62%
Regular	5	38%
Pésimo	0	0%

Fuente: Elaboración del grupo.

Gráfico N° 06: Acceso al sistema.



Fuente: Elaboración del grupo.

De acuerdo a la encuesta aplicada a los administradores de la oficina de circulación terrestre se observa que el 62% de los administradores considera que el Acceso al sistema web es bueno y el 38% del total considera que el Acceso al sistema web es regular.

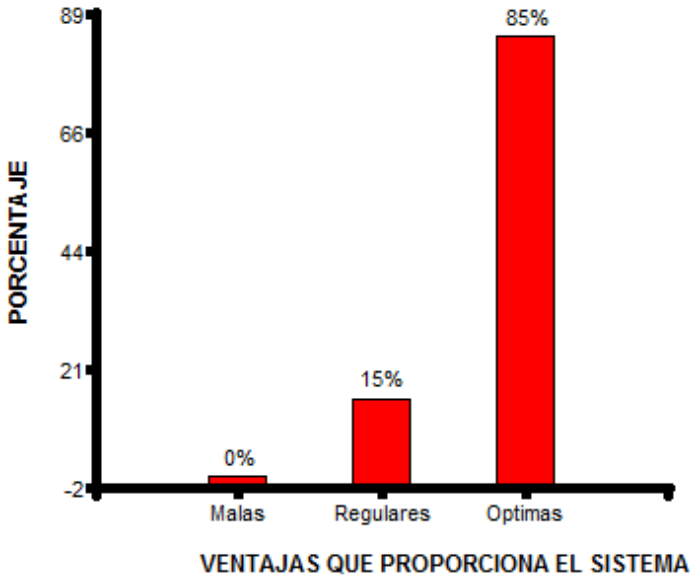
Resultados que Ventajas le Proporciona el Sistema Web

Tabla N° 10: Ventajas que proporciona el sistema web.

VENTAJAS QUE PROPORCIONA EL SISTEMA	FRECUENCIA	PORCENTAJE (%)
Malas	0	0%
Regulares	2	15%
Optimas	11	85%

Fuente: Elaboración del grupo.

Gráfico N° 07: Ventajas que proporciona el sistema web.



Fuente: Elaboración del grupo.

De acuerdo a la encuesta aplicada a los administradores de la oficina de circulación terrestre se observa que el 85% de los administradores considera que las ventajas que proporciona el sistema web son óptimas y el 15% del total considera que las ventajas que proporciona el sistema web son regulares.

4.8. Contraste de Hipótesis

Hipótesis Nula (H_o):

Desarrollo del Sistema Web Integrado para los procesos y trámites no reduce el tiempo de atención en la Oficina de Circulación Terrestre.

Hipótesis Alterna (H_a):

Desarrollo del Sistema Web Integrado para los procesos y trámites reduce el tiempo de atención en la Oficina de Circulación Terrestre.

H_o: $\mu_1 = \mu_2$

H_a: $\mu_1 > \mu_2$

Nivel de significancia

Se considera un nivel de 95% de confiabilidad y una significancia del α (0.05)

Prueba estadística

$$\mathbf{Z_c} = \frac{\bar{D}}{\frac{S_D}{\sqrt{n}}}$$

$$\bar{\mathbf{D}} = \frac{\sum_{i=1}^{n} D_i}{n} \; ; \; \mathbf{D_i} = x_1 - x_2 \; ; \; \mathbf{S_D} = \frac{\sum_{i=1}^{n} (D_i - \bar{D})^2}{n-1}$$

Donde:

$\mathbf{Z_c}$ **:** Z calculada.

$\bar{\mathbf{D}}$ **:** Promedio diferencial de la diferencia de los resultados antes y después de la implementación del Sistema web.

$\mathbf{n}$ **:** Tamaño de la muestra del grupo de estudio.

$\mathbf{S_D}$ **:** Varianza de la diferencia (antes y después).

$\mathbf{x_1}$ **:** Tiempo de atención antes de la implementación del Sistema web.

$\mathbf{x_2}$ **:** Tiempo de atención después de la implementación del Sistema web.

$\bar{D}$ = 70.49 ; S_D = 297.46 ; Z_c = 2.70

Cuadro N° 01: Prueba Z para diferencia de medias del tiempo antes y después de la implementación del Sistema Web Integrado.

Obs(1)	Obs(2)	N	media(dif)	Media(1)	Media(2)	p(Unilateral D)
Antes	Despues	130	70.49	89.62	19.12	<0.0001

Fuente: Elaboración en Infostat.

Regla de decisión

Como el P **(0.0001)** es muy inferior o menor al nivel de significancia α (0.05) entonces se rechaza la H_o.

Interpretación

Con un nivel de confianza del 95%, el desarrollo del Sistema Web Integrado para los procesos y trámites reduce el tiempo de atención en la Oficina de Circulación Terrestre.

CAPÍTULO V

CONCLUSIONES

1. Se diseñó e implemento un sistema web integrado utilizando la metodología ágil SCRUM, así como para el modelado se ha empleado el lenguaje de Modelamiento Unificado (UML), el desarrollo de dicho sistema mejora la calidad de atención en la Oficina de Circulación Terrestre de la Dirección Regional de Transportes y Comunicaciones Puno.

2. Se validó el funcionamiento del sistema web integrado para los procesos y tramites de la Oficina de Circulación Terrestre de la Dirección Regional de Transportes y Comunicaciones Puno, con la ficha de evaluación ISO – 9126 en una escala del 27 - 135 puntos, dando como resultado 108 puntos, que significa que el sistema web integrado cumple con los requisitos de calidad para su funcionamiento.

3. El tiempo de atención redujo notablemente con la implementación del sistema web integrado para los procesos y tramites de la Oficina de Circulación Terrestre de la Dirección Regional de Transportes y Comunicaciones Puno.

CAPÍTULO VI

RECOMENDACIONES

- Implementar un sistema web con la metodología SCRUM para comprobar si hay diferencia entre el presente trabajo y otros que puedan beneficiar a las oficinas del sector transporte.

- Se sugiere implementar un sistema similar en las demás instituciones públicas y privadas donde se realizan procesos y trmites, para reducir el tiempo de atención al usuario.

- Para el desarrollo del software de pequeñas magnitudes tanto en costo y documentación, se recomienda la utilización de metodologías agiles tal como SCRUM.

CAPÍTULO VII

REFERENCIAS

Baez (2012), *Sistemas Web*. Disponible en http://www.knowdo.org/knowledge/39-sistemas-web.

Borbón (2013), *Norma de Evaluación ISO/IEC 9126.* Disponible en http://actividad reconocimiento-301569-8.blogspot.com/2013/03/norma-de-evaluacion-isoiec -9126.html.

Calvo (2009), *Concepto de documento.*

Canós (2003), *Principales características de SCRUM.*

Clemente (1997), *Trabajo Colaborativo.*

Díaz (2009), *Metodología SCRUM.*

Gonzales & Mendez (2010), *Método SCRUM aplicado al sistema de gestión de seguridad de la información,* Tesis de pre grado, IUPG-Colombia 2016.

Herrera (2016), *Sistema de información para el Instituto de Informática de la Universidad Nacional del Altiplano Puno*, Tesis de pre grado UNA-Puno 2016.

Ibai (2005), *Atención o Gestión Documentaria.*

Jihuallanca (2016), *Sistema de Consulta de la Unidad de Pensiones y Liquidaciones de la Universidad Nacional del Altiplano Puno*, Tesis de pre grado UNA-Puno 2016.

Layout (2009), *Tramite documentario*.

Mancini (2003), *Modelo Vista Controlador.*

Montenegro (2015), *Sistema Web de Biblioteca Para el Instituto de Educación Superior Pedagógico Público de Juliaca*, Tesis de pregrado UNA-Puno 2015.

Nuñez (2002), *Desarrollo de Aplicaciones para Internet con Asp .Net.*

Onetti (2011), *Trabajo Colaborativo.*

Página Web, *Definición de Ministerio de Transportes y Comunicaciones, Oficina de Circulación Terrestre.* Disponible en http://www.mtc.gob.pe.

Página Web, *Metodología de Desarrollo Ágil SCRUM*. Disponible en http://www. scrumguides.org.

RENIEC (2012), *Sistema Integrado de Trámite Documentario – SITD.*

Valles (2010), *Definición UML*.

Vilca & Alferez (2014), *Aplicación Web de Tramite Documentario para la Mejora y Agilización de Trámite en el Edificio Administrativo de la Universidad Nacional del Altiplano Puno*, Tesis de pre grado UNA-Puno 2014.

Wikipedia, *Trabajo Colaborativo*. Disponible en https://es.wikipedia.org.

ANEXOS

ANEXO 1. Ficha de Evaluación de la Calidad del Producto Estándar ISO 9126.

INDICADORES	PUNTUACIÓN				
	1	2	3	4	5
1. FUNCIONALIDAD					
Adecuación: la capacidad del producto software para proporcionar un conjunto apropiado de funciones para tareas específicas y objetivos de los usuarios.					
Exactitud: la capacidad del producto software para proporcionar los resultados o efectos correctos y con el grado de precisión acordado.					
Interoperabilidad: la capacidad del producto software para interactuar con uno o más sistemas especificados					
Seguridad: referido a la capacidad del producto software para proteger la información y los datos					
Conformidad: la capacidad del producto software para adaptarse a los estándares, convenciones o regulaciones en leyes y prescripciones relativos a la funcionalidad					
2. FIABILIDAD					
Madurez: la capacidad del producto software para evitar fallos provocados por errores en el software.					
Tolerancia a fallos: la capacidad del producto software para mantener un nivel de rendimiento determinado en caso de defectos en el software o incumplimiento de su interfaz.					
Recuperabilidad: la capacidad del producto software para restablecer un determinado nivel de rendimiento y recuperar los datos afectados directamente en caso de ocurrir un fallo.					
Conformidad: la capacidad del producto software para adaptarse a estándares, convenciones y regulaciones referidas a la fiabilidad.					
3. USABILIDAD					
Comprensibilidad: la capacidad del producto software para permitir al usuario que entienda si el software es adecuado, y como debe utilizarse para determinadas tareas y bajo ciertas condiciones de uso.					
Facilidad de aprendizaje: la capacidad del producto software para permitir al usuario aprender su aplicación.					
Atracción: la capacidad del producto software para atraer al usuario.					
Conformidad: la capacidad del producto software para adaptarse a estándares, convenciones, guías de estilo y regulaciones relacionadas con la usabilidad.					
Operabilidad: la capacidad del producto software para permitir que el usuario lo opere y lo controle.					
4. EFICIENCIA					
Comportamiento temporal: la capacidad del producto software para proporcionar tiempos de respuesta y de procesamiento apropiados cuando realiza sus funciones bajo condiciones determinadas.					
Utilización de recursos: la capacidad del producto software para utilizar cantidades y tipos de recursos apropiados cuando el software realiza su función bajo determinadas condiciones.					
Conformidad: la capacidad del producto software para adaptarse a estándares o convenciones relacionadas con la eficiencia.					
5. MANTENIMIENTO					
Analizabilidad: Capacidad del producto software de diagnosticar sus deficiencias o causas de fallos, o de identificar las partes que deben ser modificadas.					
Cambiabilidad: Capacidad del producto software de permitir implementar una modificación especificada. La implementación incluye los cambios en el diseño, el código y la documentación.					
Estabilidad: Capacidad del producto software de evitar los efectos inesperados de las modificaciones.					
Facilidad de prueba: Capacidad del producto software de permitir validar las partes modificadas.					
Conformidad: Capacidad del producto software de cumplir los estándares o convenciones relativas a la mantenibilidad.					
6. PORTABILIDAD					
Adaptabilidad: la capacidad del producto software para ser adaptado para ambientes determinados sin realizar acciones o aplicar medios, más que los proporcionados para este propósito para el software considerado.					
Facilidad de instalación: la capacidad del producto software para ser instalado en un ambiente determinado. **Coexistencia:** la capacidad del producto software para coexistir con otro software independiente en un ambiente común compartiendo recursos.					
Reemplazabilidad: la capacidad del producto software para ser utilizado en lugar de otro producto de software para el mismo propósito en el mismo ambiente.					
Conformidad: la capacidad del producto software para adaptarse a estándares relacionados con la portabilidad.					
SUB TOTALES					
TOTAL					

Indicador Cualitativo	Valor
Deficiente	1
Malo	2
Regular	3
Bueno	4
Muy bueno	5

Clasificación	Intervalo	Decisión
A) Inaceptable	[27 - 54 >	
B) Mínimamente aceptable	[54 - 81 >	
C) Aceptable	[81 - 95 >	
D) Cumple los requisitos	[95 - 122 >	
F) Excede los requisitos	[122 – 135]	

Fuente: Tablas de validación

ANEXO 2. Datos de Tiempo de Atención

A continuación, se muestra los cálculos realizados.

Tiempo	Antes	Después	D_i	$D_i - \overline{D}$	$(D_i - \overline{D})^2$
1	95	20	75	4.508	20.319
2	100	18	82	11.508	132.427
3	70	19	51	-19.492	379.950
4	90	18	72	1.508	2.273
5	95	20	75	4.508	20.319
6	80	18	62	-8.492	72.119
7	120	20	100	29.508	870.704
8	80	18	62	-8.492	72.119
9	100	20	80	9.508	90.396
10	80	20	60	-10.492	110.089
11	90	18	72	1.508	2.273
12	100	20	80	9.508	90.396
13	120	20	100	29.508	870.704
14	120	18	102	31.508	992.735
15	60	18	42	-28.492	811.812
16	95	20	75	4.508	20.319
17	85	19	66	-4.492	20.181
18	65	18	47	-23.492	551.889
19	60	18	42	-28.492	811.812
20	80	19	61	-9.492	90.104
21	70	18	52	-18.492	341.965
22	90	18	72	1.508	2.273
23	120	18	102	31.508	992.735
24	65	20	45	-25.492	649.858
25	70	20	50	-20.492	419.935
26	100	18	82	11.508	132.427
27	70	19	51	-19.492	379.950
28	100	20	80	9.508	90.396
29	90	20	70	-0.492	0.242
30	90	18	72	1.508	2.273
31	65	20	45	-25.492	649.858
32	90	18	72	1.508	2.273
33	100	18	82	11.508	132.427
34	80	20	60	-10.492	110.089
35	100	18	82	11.508	132.427
36	70	20	50	-20.492	419.935
37	65	18	47	-23.492	551.889
38	95	18	77	6.508	42.350
39	90	20	70	-0.492	0.242
40	100	19	81	10.508	110.412
41	90	20	70	-0.492	0.242
42	100	18	82	11.508	132.427
43	120	20	100	29.508	870.704

Tiempo	Antes	Después	D_i	$D_i - \bar{D}$	$(D_i - \bar{D})^2$
44	85	19	66	-4.492	20.181
45	100	20	80	9.508	90.396
46	95	19	76	5.508	30.335
47	100	18	82	11.508	132.427
48	70	18	52	-18.492	341.965
49	60	20	40	-30.492	929.781
50	95	20	75	4.508	20.319
51	60	19	41	-29.492	869.796
52	95	18	77	6.508	42.350
53	95	19	76	5.508	30.335
54	120	18	102	31.508	992.735
55	85	20	65	-5.492	30.165
56	120	18	102	31.508	992.735
57	120	20	100	29.508	870.704
58	60	20	40	-30.492	929.781
59	60	18	42	-28.492	811.812
60	80	18	62	-8.492	72.119
61	120	19	101	30.508	930.719
62	85	20	65	-5.492	30.165
63	100	20	80	9.508	90.396
64	105	18	87	16.508	272.504
65	65	18	47	-23.492	551.889
66	85	19	66	-4.492	20.181
67	90	18	72	1.508	2.273
68	80	20	60	-10.492	110.089
69	60	19	41	-29.492	869.796
70	75	20	55	-15.492	240.012
71	60	20	40	-30.492	929.781
72	95	18	77	6.508	42.350
73	120	20	100	29.508	870.704
74	100	18	82	11.508	132.427
75	90	19	71	0.508	0.258
76	100	20	80	9.508	90.396
77	90	20	70	-0.492	0.242
78	120	18	102	31.508	992.735
79	60	20	40	-30.492	929.781
80	90	18	72	1.508	2.273
81	60	19	41	-29.492	869.796
82	65	20	45	-25.492	649.858
83	70	20	50	-20.492	419.935
84	80	19	61	-9.492	90.104
85	95	18	77	6.508	42.350
86	90	20	70	-0.492	0.242
87	120	20	100	29.508	870.704
88	80	18	62	-8.492	72.119
89	120	20	100	29.508	870.704
90	120	19	101	30.508	930.719
91	70	20	50	-20.492	419.935
92	80	19	61	-9.492	90.104

Tiempo	Antes	Después	D_i	$D_i - \bar{D}$	$(D_i - \bar{D})^2$
93	100	20	80	9.508	90.396
94	80	18	62	-8.492	72.119
95	120	20	100	29.508	870.704
96	70	18	52	-18.492	341.965
97	60	20	40	-30.492	929.781
98	100	19	81	10.508	110.412
99	85	20	65	-5.492	30.165
100	80	20	60	-10.492	110.089
101	75	18	57	-13.492	182.042
102	100	19	81	10.508	110.412
103	95	20	75	4.508	20.319
104	100	20	80	9.508	90.396
105	80	19	61	-9.492	90.104
106	90	19	71	0.508	0.258
107	95	20	75	4.508	20.319
108	85	19	66	-4.492	20.181
109	75	18	57	-13.492	182.042
110	95	20	75	4.508	20.319
111	120	20	100	29.508	870.704
112	90	19	71	0.508	0.258
113	90	20	70	-0.492	0.242
114	90	20	70	-0.492	0.242
115	95	18	77	6.508	42.350
116	85	19	66	-4.492	20.181
117	100	20	80	9.508	90.396
118	120	20	100	29.508	870.704
119	85	20	65	-5.492	30.165
120	100	20	80	9.508	90.396
121	85	18	67	-3.492	12.196
122	95	20	75	4.508	20.319
123	90	20	70	-0.492	0.242
124	95	18	77	6.508	42.350
125	120	20	100	29.508	870.704
126	80	20	60	-10.492	110.089
127	100	18	82	11.508	132.427
128	95	20	75	4.508	20.319
129	85	18	67	-3.492	12.196
130	90	20	70	-0.492	0.242
SUMA	**11650**	**2486**	**9164**		**38372.49**
PROMEDIO	**89.615**	**19.12**	**70.49**		**297.4612**

ANEXO 3. Manual del Sistema

UNIVERSIDAD NACIONAL DEL ALTIPLANO - PUNO
FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA
ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

MANUAL DEL SISTEMA

SISTEMA WEB INTEGRADO PARA LA OFICINA DE CIRCULACIÓN TERRESTRE DE LA DIRECCIÓN REGIONAL DE TRANSPORTES Y COMUNICACIONES PUNO

PRESENTACIÓN

En el presente documento se presenta el manual de usuario del SISTEMA WEB INTEGRADO PARA LA OFICINA DE CIRCULACIÓN TERRESTRE DE LA DIRECCIÓN REGIONAL DE TRANSPORTES Y COMUNICACIONES PUNO que tiene Como objetivo proporcionar una guía de práctica a los encargados de dicha área, para el manejo y domino del sistema también se detalla cada una de las opciones del menú principal, así como las instrucciones necesarias y las acciones a realizar en cada pantalla.

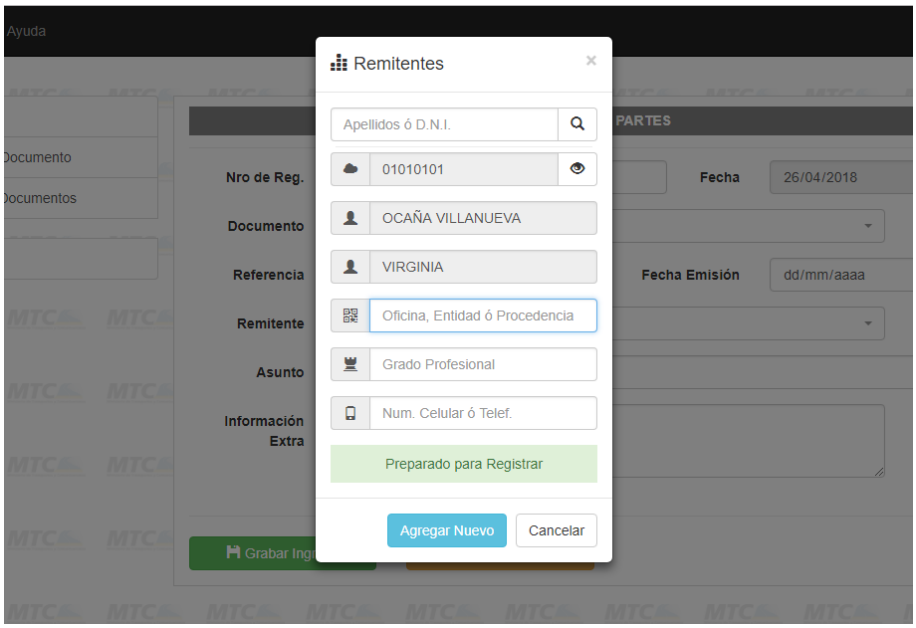
1. INTERFAZ DEL SOFTWARE DE TRAMITE DOCUMENTARIO

Aquí se muestra la primera pantalla para el ingreso del administrador al sistema.

Nos mostrará lo siguiente

2. INGRESO DE DOCUMENTOS O INICIO DE TRAMITE

Búsqueda de datos de usuarios mediante número de DNI



Después de haber llenado los campos de registro haga clic en guardar y se guardara el nuevo documento ingresado en la base de datos.

Listado de documentos ingresados y despliegue por oficinas registradas.

Finalizando ya el uso del sistema daremos click en el botón cerrar sesión.

ANEXO 4. Script SQL de la Base de Datos

```
SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";

--
-- Base de datos: `mtc_tramites`
--

--
-- Estructura de tabla para la tabla `dicdocums`
--

CREATE TABLE `dicdocums` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `IdOfi` int(11) NOT NULL,
  `Nombre` varchar(75) NOT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=10 ;

--
-- Volcado de datos para la tabla `dicdocums`
--

INSERT INTO `dicdocums` (`Id`, `IdOfi`, `Nombre`) VALUES
(1, 1, 'SOLICITUD'),
(2, 1, 'TRAMITE DE LICENCIA'),
(3, 1, 'MEMORANDUM'),
(9, 1, 'CARTA');

--
-- Estructura de tabla para la tabla `dicremites`
--

CREATE TABLE `dicremites` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `DNI` varchar(10) DEFAULT NULL,
  `Apellidos` varchar(75) DEFAULT NULL,
  `Nombres` varchar(75) DEFAULT NULL,
  `Grado` varchar(10) DEFAULT NULL,
  `Celular` varchar(22) DEFAULT NULL,
  `Entidad` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=7 ;

--
-- Volcado de datos para la tabla `dicremites`
--

INSERT INTO `dicremites` (`Id`, `DNI`, `Apellidos`, `Nombres`, `Grado`, `Celular`, `Entidad`) VALUES
(4, '41939070', 'EGUIZABAL ROJAS', 'LISBETH LILA', NULL, NULL, ''),
(6, '44917369', 'GOMEZ SACARI', 'ROSA MARIA', NULL, NULL, '');

-- --------------------------------------------------------
--
-- Estructura de tabla para la tabla `documentos`
--

CREATE TABLE `documentos` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `IdOfi` int(11) NOT NULL,
  `Estado` int(11) NOT NULL,
  `Anio` int(11) DEFAULT NULL,
  `Orden` int(11) DEFAULT NULL,
  `NroReg` varchar(12) DEFAULT NULL,
  `Folios` int(11) DEFAULT NULL,
  `IdDocumen` int(11) DEFAULT NULL,
  `Referens` varchar(70) DEFAULT NULL,
  `FechaEmi` date DEFAULT NULL,
  `IdRemiten` int(11) DEFAULT NULL,
  `Asunto` varchar(70) DEFAULT NULL,
  `FechaReg` datetime DEFAULT NULL,
  `Obs` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;

--
-- Volcado de datos para la tabla `documentos`
--

INSERT INTO `documentos` (`Id`, `IdOfi`, `Estado`, `Anio`, `Orden`, `NroReg`, `Folios`, `IdDocumen`, `Referens`, `FechaEmi`,
`IdRemiten`, `Asunto`, `FechaReg`, `Obs`) VALUES
(5, 2, 1, 2018, 1, '0001-2018', 1, 3, '001-2018', '0001-01-01', 4, 'TRAMITE', '2018-06-27 10:19:47', NULL);

--
-- Estructura de tabla para la tabla `historiales`
--

CREATE TABLE `historiales` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `Usuario` varchar(20) NOT NULL,
  `Accion` varchar(50) NOT NULL,
  `Fecha` datetime NOT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=79 ;

--
-- Volcado de datos para la tabla `historiales`
--


--
-- Estructura de tabla para la tabla `movimientos`
--

CREATE TABLE `movimientos` (
```

```
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `Iter` int(11) DEFAULT NULL,
  `IdOfi` int(11) DEFAULT NULL,
  `IdDoc` int(11) DEFAULT NULL,
  `Estado` int(11) DEFAULT NULL,
  `Accion` varchar(100) DEFAULT NULL,
  `FechaIn` datetime DEFAULT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=26 ;

--
-- Volcado de datos para la tabla `movimientos`
--

INSERT INTO `movimientos` (`Id`, `Iter`, `IdOfi`, `IdDoc`, `Estado`, `Accion`, `FechaIn`) VALUES
(23, 1, 1, 5, 1, 'Ingreso', '2018-06-27 10:19:47'),
(24, 1, 2, 5, 1, '...', '2018-06-27 10:33:58');

--
-- Estructura de tabla para la tabla `oficinas`
--

CREATE TABLE `oficinas` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `Nivel` int(11) NOT NULL,
  `NombOfi` varchar(25) NOT NULL,
  `Operador` varchar(75) NOT NULL,
  `Usuario` varchar(20) NOT NULL,
  `Clave` varchar(100) NOT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=4 ;

--
-- Volcado de datos para la tabla `oficinas`
--

INSERT INTO `oficinas` (`Id`, `Nivel`, `NombOfi`, `Operador`, `Usuario`, `Clave`) VALUES
(1, 0, 'Mesa de Partes', 'El Admin', 'mesa', '*23AE809DDACAF96AF0FD78ED04B6A265E05AA257'),
(2, 0, 'Asesoria Legal', 'Ase Legal', 'asesoria', '*23AE809DDACAF96AF0FD78ED04B6A265E05AA257'),
(3, 0, 'Tramite Documentario', 'Tram Doc', 'tramite', '*23AE809DDACAF96AF0FD78ED04B6A265E05AA257');

--
-- Estructura de tabla para la tabla `personas`
--

CREATE TABLE `personas` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `TipoDoc` int(11) NOT NULL,
  `NroDoc` varchar(12) NOT NULL,
  `Apellidos` varchar(150) NOT NULL,
  `Nombres` varchar(150) NOT NULL,
  `FechaNac` date NOT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;

--
-- Volcado de datos para la tabla `personas`
--

INSERT INTO `personas` (`Id`, `TipoDoc`, `NroDoc`, `Apellidos`, `Nombres`, `FechaNac`) VALUES
(1, 1, '01010101', 'OSCAÑA VILLANUEVA', 'GLORIA', '1940-01-01');

--
-- Estructura de tabla para la tabla `proveidos`
--

CREATE TABLE `proveidos` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `IdTramite` int(11) NOT NULL,
  `Destino` varchar(100) NOT NULL,
  `Obs` varchar(50) NOT NULL,
  `Fecha` datetime NOT NULL,
  `FechaReg` datetime NOT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;


--
-- Estructura para la vista `vwmovims`
--
DROP TABLE IF EXISTS `vwmovims`;

CREATE VIEW vwMovims AS
select `m`.`Id` AS `Id`,`d`.`Id` AS `IdOrg`,
       `m`.`Iter` AS `Iter`,`m`.`Estado` AS `Estado`,
       `m`.`IdOfi` AS `IdOfi`,`m`.`IdDoc` AS `IdDoc`,
       `d`.`NroReg` AS `NroReg`,`dd`.`Nombre` AS `Nombre`,
       `d`.`Folios` AS `Folios`,`d`.`Referens` AS `Referens`,
       `d`.`IdRemiten` AS `IdRemiten`,
       ((((`dr`.`Grado` + ' ') + `dr`.`Nombres`) + ' ') + `dr`.`Apellidos`) AS `Remitente`,
       `dr`.`Entidad` AS `Entidad`,
       `d`.`Asunto` AS `Asunto`,`m`.`Accion` AS `Accion`,
       `m`.`FechaIn` AS `FechaIn`,`d`.`Obs` AS `Obs`
from (((`movimientos` `m` join `documentos` `d`)
join `dicdocums` `dd`) join `dicremites` `dr`)
where ((`d`.`Id` = `m`.`IdDoc`)
  and (`dd`.`Id` = `d`.`IdDocumen`)
  and (`dr`.`Id` = `d`.`IdRemiten`))
order by `d`.`Id`,`m`.`Iter`;
```

ANEXO 5. Código Fuente

```
/*
**Paso 1. coneccion manual en DbContext
  using Microsoft.EntityFrameworkCore;
  using MySql.Data.EntityFrameworkCore;

**Paso 2.
  Install-Package MySql.Data.Entity -Version 6.9.5
  Install-Package MySql.Data -Version 6.9.5
*/

using System;
using System.Text;
using System.Collections.Generic;
using System.Diagnostics;
using System.Threading.Tasks;
using System.Linq;
using Microsoft.AspNetCore.Mvc;
using coreTramiMtc.Models;


namespace coreTramiMtc.Controllers
{
    public class InicioController : Controller
    {
        DbTramDoc db;
        //DbPosgrado pg;

        public InicioController( DbTramDoc _db )
        {
            db = _db;
        }

        public IActionResult Index()
        {
            if ( InSession() )
            {
                ViewBag.Sess = GetSession();
                return View("Index");
            }
            else
                return View("Login");
        }

        public IActionResult About( string id )
        {
            SessData sess =
HttpContext.Session.Get<SessData>("tmpSess");

            string str = ( id != null )? id : "123";

            ViewData["msg1"] = Crypt.doSHA1(str);
            ViewData["msg2"] = Crypt.doPass(str);

            if( sess != null )
                ViewData["msg3"] = sess.UserName + sess.UserAlias;
            return View();
        }

        //[Route("{id}")]
        //[Route("pepa/{id}")]
        public string Aqui( string id="" )
        {
            return "aqui: " + id;
        }

        public IActionResult Ayuda()
        {
            return View();
        }
        [HttpPost]
        public IActionResult Login(string user, string pass, int[]
valor)
        {
            pass = Crypt.doPass(pass);

            var usr = from e in db.Oficinas
                      where e.Usuario==user && e.Clave==pass
                      select e;

            if ( usr.Count() == 0 )
            {
                ViewBag.Error = "(" + user + ") : Usuario
incorrecto";
                return View("Login");
            }

            // quitar los mensajes de alerta
            ViewBag.Error = "";

            // datos de admin en session
            Oficina ofi = usr.First();

            HttpContext.Session.Set<SessData>( "tmpSess", new
SessData{
                    IdUser    = ofi.Id,
                    Level     = ofi.Nivel,
                    UserName  = ofi.NombOfi,
                    UserAlias = ofi.Usuario
                }
            );

            logAdd( ofi.Usuario, "Login" );

            return RedirectToAction("");
        }

        public ActionResult Logout()
        {
            SessData sess =
HttpContext.Session.Get<SessData>("tmpSess");
            if( sess!=null )
                logAdd( sess.UserAlias, "Logout" );

            HttpContext.Session.Clear();
            ViewBag.Error = null;

            return RedirectToAction("");
        }

        private SessData GetSession()
        {
            return  HttpContext.Session.Get<SessData>( "tmpSess"
);
        }

        private bool InSession()
        {
            SessData sess =
HttpContext.Session.Get<SessData>("tmpSess");
            return (sess!=null)? true : false;
        }

        private void logAdd( string user, string action )
        {
            // Id Primary Key Auto_numeric
            //
            db.Historiales.Add( new Historial{
                    Usuario = user,
                    Accion = action,
                    Fecha = DateTime.Now
                } );

            db.SaveChanges();
        }

    }
}
```

INNER: MODULE

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;

using Newtonsoft.Json;
using coreTramiMtc.Models;


namespace coreTramiMtc.Controllers
{
    public class InnerController : Controller
    {
        DbTramDoc db;

        public InnerController( DbTramDoc _db )
        {
            db = _db;
        }

        public IActionResult Index()
        {
            // Content("html content");
            return RedirectToAction("Index","Inicio");
        }

        private SessData GetSession()
        {
            return  HttpContext.Session.Get<SessData>( "tmpSess"
);
        }

        public IActionResult Ingreso()
        {
            var sess = GetSession();
            if( sess==null ) return View("Refresh");

            ViewBag.inNoReg =
string.Format("{0:0000}",db.GetLibOrder());
            ViewBag.inFecha = DateTime.Now.Date.ToString("yyyy-MM-
dd");
            ViewBag.dicRemi = db.DicRemites.ToList();
            ViewBag.dicDocu = db.DicDocums.ToList();

            return View();
        }

        public IActionResult Listado( string id )
        {
            var sess = GetSession();
            if( sess==null ) return View("Refresh");

            ViewBag.IdOfi    = sess.IdUser;
```

```
            ViewBag.dicOfis  = from e in db.Oficinas select e;

            ViewBag.vwMovims = from e in db.vwMovims
                               where e.IdOfi == sess.IdUser
                               orderby e.Id descending
                               select e;
            return View();
        }

        //[HttpPost]
        public string savDest( int iddoc, int iddest,
                               int idest, string coment )
        {
            var sess = GetSession();
            if( iddoc <= 0 ){
                return "Sin ID";
            }

            var docu = (from e in db.Documentos where e.Id==iddoc
select e).First();
            var movi = new Movimiento {
                    Iter    = 1,
                    Estado  = idest,
                    IdOfi   = iddest,
                    IdDoc   = iddoc,
                    Accion  = TryUpper(coment),
                    FechaIn = DateTime.Now
                };

            db.Movimientos.Add( movi );

            docu.IdOfi  = iddest;
            docu.Estado = idest;
            db.Documentos.Update( docu );

            db.SaveChanges();

            return "Enviado...";
        }

        [HttpPost]
        public string GrabIngreso( int noreg, int folios, DateTime
fecha,
                                   int iddocu, string refdoc,
                                  DateTime fechemi, int idremi,
                                  string asunto, string obs )
        {
            var sess = GetSession();
            if( sess==null ) return "Sesion Finalizada";

            int anio = db.GetAnio();
            int orde = db.GetLibOrder();
            string R = string.Format( "{0:0000}-{1}", orde, anio
);

            var docu = new Documento{
                    IdOfi  = sess.IdUser,
                    Estado = 1,
                    Anio   = anio,
                    Orden  = orde,
                    NroReg = R,
                    Folios = folios,
                    IdDocumen = iddocu,
                    Referens  = refdoc,
                    FechaEmi  = fechemi,
                    IdRemiten = idremi,
                    Asunto    = TryUpper(asunto),
                    FechaReg  = DateTime.Now, //.ToString("yyyy-
MM-dd"),
                    Obs = obs
                };

            db.Documentos.Add( docu );
            db.SaveChanges();

            db.Movimientos.Add(
                new Movimiento{
                    Iter    = 1,
                    IdOfi   = docu.IdOfi,
                    IdDoc   = docu.Id,
                    Estado  = 1,
                    Accion  = "Ingreso",
                    FechaIn = DateTime.Now //.ToString("yyyy-MM-
dd")
                }
            );
            db.SaveChanges();

            return string.Format("<hr> Grabado con Registro Nro:
{0}", R, docu.Id );
        }

       [HttpPost]
        public string evaDocs( string id )
        {
            var tbl = from e in db.DicDocums
                      where e.Nombre.Contains(id)
                      select e;  //  (SQL).First();

            if( tbl.Count() >= 1 ) {
                return "{\"exists\":true}";
            } else {
                return "{\"exists\":false}";
            }
        }

        [HttpPost]
        public string savDocs( string id )
        {
            var sess = GetSession();
            var res1 = "{\"error\":true}";


            if( id == null || sess == null )
                return res1;

            if( evaDocs(id) == "{\"exists\":false}" )
            {
                DicDocum p1 = new DicDocum{
                        IdOfi  = sess.IdUser,
                        Nombre = id.ToUpper()
                    };

                db.DicDocums.Add( p1 );
                //db.SaveChanges();
                db.SaveChangesAsync();

                string htm =
 "<input type=hidden name=iddocu value=" +p1.Id.ToString()+ ">" +
 "<input type=text value='"+p1.Nombre+"' class='form-control
input-md' readonly>";
                return "{\"error\":false, \"xhtml\":\""+htm+"\"
}";
            }
            return res1;
        }

        protected string TryUpper( string id )
        {
            return  (id==null)? "":id.ToUpper();
        }

        [HttpPost]
        public async Task<string> evaData( string id )
        {
            bool numb = Crypt.IsNumeric(id);

            var t1 = from e in db.DicRemites
                     where e.DNI == id
                     select e;

            var t2 = from e in db.DicRemites
                     where e.Apellidos.Contains(id) ||
e.Nombres.Contains(id)
                     select e;
            if( numb && t1.Count()==0 ){
                // modo API nuevo
                if( id.Length == 8 ){
                    return await LibHttp.apiSunacDatos( id );
                }
                // reg corto nuevo
                return "{\"modo\":2}";

            } else if( numb ){
                // reg previo, modo edición.
                string dta = JsonConvert.SerializeObject(
t1.First() );
                return "{\"modo\":3,\"data\":"+dta+"}";

            } else if( t2.Count()==0 ){
                // reg corto nuevo
                return "{\"modo\":2}";

            } else {
                // editar == 3
                string dta = JsonConvert.SerializeObject(
t2.First() );
                return "{\"modo\":3,\"data\":"+dta+"}";
            }
        }
        [HttpPost]
        public string savRemi( int modo, string numdni, string
apells,
                               string nombre, string entida,
                               string gradox, string celula )
        {
            var sess = GetSession();
            var res1 = "{\"error\":true}";

            if( sess == null || numdni==null )
                return res1;

            if( modo<1 && modo>3 )
                return res1;

            // buscamos registro previo general
            DicRemite p1;
            var  t1 = from e in db.DicRemites
                      where e.DNI == numdni
                      select e;


            // modo edicion de Remitente
            if( modo==3 ){

                p1 = t1.First();

                p1.DNI = numdni;
                p1.Apellidos = TryUpper(apells);
                p1.Nombres = TryUpper(nombre);
                p1.Entidad = TryUpper(entida);
                p1.Grado   = gradox;
                p1.Celular = celula;
```

```
            // modo insertado de datos
            } else {

                if( t1.Count() > 0 ) return res1;

                p1 = new DicRemite{
                        DNI       = numdni,
                        Apellidos = TryUpper(apells),
                        Nombres   = TryUpper(nombre),
                        Entidad   = TryUpper(entida),
                        Celular   = celula,
                        Grado     = gradox
                    };

                db.DicRemites.Add( p1 );
            }
            db.SaveChangesAsync();

            string htmx = "<input type=hidden name=idremi value="
+p1.Id.ToString()+ ">" +
                          "<input type=text value='"+p1.Nombres+"
"+p1.Apellidos+
                          "' class='form-control input-md'
readonly>";

            return "{\"error\":false, \"xhtml\":\""+htmx+"\" }";
        }

        public string doBaseUrl( String path="" )
        {
            return
$"{this.Request.Scheme}://{this.Request.Host}{this.Request.PathBas
e}{path}";
        }



    }
}
```

MODULO: SESSION

```
using System;
using System.Text;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Newtonsoft.Json;



namespace coreTramiMtc.Controllers
{
    // solo sesion local sin DB
    public class SessData
    {
        public int Level;
        public int IdUser;
        public string UserName;
        public string UserAlias;
    }

    public static class Crypt
    {
        // funciones de Extension para clase Session usar Objetos
        public static void Set<T>(this ISession session, string
key, T value)
        {
            session.SetString( key,
JsonConvert.SerializeObject(value) );
        }

        public static T Get<T>(this ISession session, string key)
        {
            var value = session.GetString( key );
            return  (value == null) ?
                    default(T) :
                    JsonConvert.DeserializeObject<T>( value );
        }
        //-----------------------------------------------------------
-------
        public static bool IsNumeric( string val )
        {
            decimal numx;
            return decimal.TryParse( val, out numx);
        }

        public static string doSHA1( string input )
        {
            if (input == null) return "";

            var sha1 = System.Security.Cryptography.SHA1.Create();
            byte[] b = Encoding.UTF8.GetBytes(input);
            b = sha1.ComputeHash(b);

            StringBuilder sb = new StringBuilder();
            foreach (byte x in b)
                sb.Append( x.ToString("x2") );

            return sb.ToString().ToUpper();
        }

        // * += 2 x SHA1
        public static string doPass( string input )
        {
            if (input == null) return "";

            var sha1 = System.Security.Cryptography.SHA1.Create();
            byte[] b = Encoding.UTF8.GetBytes(input);
            b = sha1.ComputeHash(b);
            b = sha1.ComputeHash(b);

            StringBuilder sb = new StringBuilder();
            foreach (byte x in b)
                sb.Append( x.ToString("x2") );

            return "*" + sb.ToString().ToUpper();
        }

        public static string doBaseUrl( String path="" )
        {
            return $"{this.Request.Scheme}://{this.Request.Host}
{this.Request.PathBase}{path}";
        }
    }
}
```

MODULO: MODELO DE DATOS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.EntityFrameworkCore;
using System.Data;

namespace coreTramiMtc.Models
{
    public class Oficina
    {
        // esta seria la tabla de usuario y accesos
        public int Id { get; set; }
        public int Nivel { get; set; }
        public string NombOfi { get; set; }
        public string Operador { get; set; }
        public string Usuario { get; set; }
        public string Clave { get; set; }
    }

    public class Historial
    {
        public int Id { get; set; }
        public string Usuario { get; set; }
        public string Accion { get; set; }
        public DateTime Fecha { get; set; }
    }



    public class DicRemite
    {
        public int Id { get; set; }
        public string DNI { get; set; }
        public string Apellidos { get; set; }
        public string Nombres { get; set; }
        public string Grado { get; set; }
        public string Celular { get; set; }
        public string Entidad { get; set; }
    }

    public class DicDocum
    {
        public int Id { get; set; }
        public int IdOfi { get; set; }
        public string Nombre { get; set; }
    }

    public class Movimiento
    {
        public int Id    { get; set; }
        public int Iter  { get; set; }
        public int IdOfi { get; set; }
        public int IdDoc { get; set; }
        public int Estado { get; set; }
        public string Accion { get; set; }
        public DateTime FechaIn { get; set; }
    }

    public class Documento
    {
        public int Id { get; set; }
        public int IdOfi { get; set; }
        public int Estado { get; set; }
        public int Anio { get; set; }
        public int Orden { get; set; }
        public string NroReg { get; set; }
        public int Folios { get; set; }
        public int IdDocumen { get; set; }
        public string Referens { get; set; }
        public DateTime FechaEmi { get; set; }
        public int IdRemiten { get; set; }
        public string Asunto { get; set; }
        public DateTime FechaReg { get; set; }
        public string Obs { get; set; }
    }

    public class vwMovim
    {
        public int Id { get; set; }
        public int IdOrg { get; set; }
        public int Iter { get; set; }
        public int Estado { get; set; }
        public int IdOfi { get; set; }
        public int IdDoc { get; set; }
```

```
        public string NroReg { get; set; }
        public string Nombre { get; set; }
        public int Folios { get; set; }
        public string Referens { get; set; }
        public int IdRemiten { get; set; }
        public string Remitente { get; set; }
        public string Entidad { get; set; }
        public string Asunto { get; set; }
        public string Accion { get; set; }
        public DateTime FechaIn { get; set; }
        public string Obs { get; set; }
    }
    // Conexion a BD1 y sus N tablas
    public class DbTramDoc : DbContext
    {
        public DbTramDoc( DbContextOptions<DbTramDoc> options ) :
base( options ) { }

        protected override void OnConfiguring(
DbContextOptionsBuilder optbuild  )
        {
            //var strconn = "workstation
id=rpGeneral.mssql.somee.com;packet size=4096;
user id=rpedrolm_SQLLogin_1;pwd=qatudkz2gs;data
source=rpGeneral.mssql.somee.com;persist security
info=False;initial catalog=rpGeneral";

            var strconn =
"Server=127.0.0.1;database=mtc_tramites;user=root;password=finesi+
+";
            optbuild.UseMySQL(strconn);
        }
        public virtual DbSet<Oficina>     Oficinas     { get; set;
}
        public virtual DbSet<Historial>   Historiales  { get; set;
}
        public virtual DbSet<DicRemite>   DicRemites   { get; set;
}
        public virtual DbSet<DicDocum>    DicDocums    { get; set;
}
        public virtual DbSet<Documento>   Documentos   { get; set;
}
        public virtual DbSet<Movimiento>  Movimientos  { get; set;
}

        public virtual DbSet<vwMovim>  vwMovims  { get; set; }
            // metodos publicos de recoleccion de datos
        public int GetLibOrder()
        {
            var tlib = from e in Documentos
                       where e.Anio == this.GetAnio()
                       orderby e.Orden descending select e;

            return (tlib.Count()==0)? 1 : tlib.First().Orden+1;
        }

        public int GetAnio()
        {
            return int.Parse( DateTime.Now.Date.ToString("yyyy")
);
        }
    }
    public class DbPosgrado : DbContext
    {
        public DbPosgrado(DbContextOptions<DbPosgrado> options) :
            base( options )
        {
        }
    }
}
```

ANEXO 6. Encuesta al Usuario

UNIVERSIDAD NACIONAL DEL ALTIPLANO - PUNO
FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA
ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

ENCUESTA PARA LOS USUARIOS QUE REALIZAN PROCESOS Y TRAMITES EN LA OFICINA DE CIRCULACIÓN TERRESTRE DE LA DIRECCIÓN REGIONAL DE TRANSPORTES Y COMUNICACIONES PUNO.

A continuación, se le presenta 3 preguntas, marcar con un aspa (X) una de las alternativas que cree conveniente de acuerdo a su criterio y experiencia durante su permanencia en la institución.

1. ¿Cómo considera el tiempo de atención?

a) Rápido
b) Medianamente Rápido
c) Lento

2. ¿Cuál fue el tiempo que se demoraba en la atención, antes de la implementación del sistema web?

…….:……..

3. ¿Cuál es el tiempo que se demora en la atención, con la implementación del sistema web?

……

ANEXO 7. Encuesta al Administrador

UNIVERSIDAD NACIONAL DEL ALTIPLANO - PUNO
FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA
ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

ENCUESTA PARA LOS ADMINISTRADORES DE LA OFICINA DE CIRCULACIÓN TERRESTRE DE LA DIRECCIÓN REGIONAL DE TRANSPORTES Y COMUNICACIONES PUNO.

A continuación, se le presenta 4 preguntas, marcar con un aspa (X) una de las alternativas que cree conveniente de acuerdo a su criterio y experiencia durante su permanencia en la institución.

1. **¿Cómo considera Ud. el diseño de la interfaz del sistema web?**
   a) Muy bueno
   b) Bueno
   c) Regular
2. **¿Cómo considera Ud. los servicios que ofrece el sistema Web?**
   a) Eficiente.
   b) Medianamente eficiente.
   c) Deficiente
3. **¿Cómo considera Ud. el acceso al sistema web?**
   a) Bueno
   b) Regular
   c) Pésimo
4. **¿Qué ventajas le proporciona el sistema web?**
   a) Malas
   b) Regulares
   c) Óptimas