FACULTAD DE INGENIERIA

ESCUELA ACADÉMICO PROFESIONAL DE INGENIERÍA DE SISTEMAS

TÍTULO

ARQUITECTURA BASADA EN UNA CAPA DE CONTROL DE EXCEPCIONES PARA MEJORAR LA FIABILIDAD DE LA APLICACIÓN SOFTWARE DE PRÉSTAMOS BANCARIOS

TESIS PARA OBTENER EL TÍTULO PROFESIONAL DE INGENIERO DE SISTEMAS

AUTOR

DAGNER ARQUIMIDES PILLAMANGO MENDOZA

ASESOR

ING. LAÍN CÁRDENAS ESCALANTE

LÍNEA DE INVESTIGACIÓN

SISTEMA DE INFORMACIÓN TRANSACCIONALES

TRUJILLO – PERÚ

2016

PÁGINA DEL JURADO

El presidente y los miembros del Jurado Evaluador designado por la escuela de ingeniería de sistemas

APRUEBAN

La tesis denominada:

“ARQUITECTURA BASADA EN UNA CAPA DE CONTROL DE EXCEPCIONES PARA MEJORAR LA FIABILIDAD DE LA APLICACIÓN SOFTWARE DE PRÉSTAMOS BANCARIOS”.

Presentado por:

PILLAMANGO MENDOZA, DAGNER ARQUIMIDES

Jurado evaluador:

Ing. Pacheco Torres Juan Francisco
PRESIDENTE DEL JURADO.

Ing. Oscar Alcántara Moreno
SECRETARIO

Ing. Cárdenas Escalante Laín
VOCAL

DEDICATORIA

A MIS PADRES

Los que me brindaron todo su apoyo, confianza para poder cumplir mis metas y así poder terminar satisfactoriamente.

A MIS DOCENTES

Por compartir sus conocimientos, sus buenos valores, sus experiencias durante todo este tiempo en la universidad y fueron de mucha ayuda en lo profesional.

AGRADECIMIENTO

A ti DIOS, que guías mi camino, me cuidas y que cada día mejor persona.

A la Universidad César Vallejo, que es una institución que nos brindó las facilidades y comodidades en el proceso de mi profesión.

Al Ing. Edward Vega Gavidia, por su guía durante el proceso de proyecto de investigación.

Al Ing. Laín Cárdenas Escalante quien dedico su tiempo en asesorarme durante el desarrollo del proyecto y por darme las facilidades de cederme un software para mi investigación.

A mi familia, que siempre me apoyaron e hicieron posible terminar el presente proyecto.

DECLARATORIA DE AUTENTICIDAD

Yo **Dagner Arquimides Pillamango Mendoza** con DNI Nº **44107130**, a efecto de cumplir con las disposiciones vigentes consideradas en el Reglamento de Grados y Títulos de la Universidad César Vallejo, Facultad de Ingeniería, Escuela Académico Profesional de Ingeniería de Sistemas, declaro bajo juramento que toda la documentación que acompaño es veraz y auténtica.

Así mismo, declaro también bajo juramento que todos los datos e información que se presenta en la presente tesis son auténticos y veraces.

En tal sentido asumo la responsabilidad que corresponda ante cualquier falsedad, ocultamiento u omisión tanto de los documentos como de información aportada por lo cual me someto a lo dispuesto en las normas académicas de la Universidad César Vallejo.

Trujillo, Diciembre del 2016

Dagner Arquimides Pillamango Mendoza

PRESENTACIÓN

Para cumplir el Reglamento de Grados y Títulos de la Universidad César Vallejo presento ante ustedes la Tesis titulada: **"ARQUITECTURA BASADA EN UNA CAPA DE CONTROL DE EXCEPCIONES PARA MEJORAR LA FIABILIDAD DE LA APLICACIÓN SOFTWARE DE PRÉSTAMOS BANCARIOS"**. La misma que dejo a su consideración y espero que cumpla con los requisitos de aprobación para obtener el título Profesional de Ingeniero de Sistemas.

Pillamango Mendoza, Dagner Arquimides

ÍNDICE

RESUMEN

La presente investigación titulada: **“ARQUITECTURA BASADA EN UNA CAPA DE CONTROL DE EXCEPCIONES PARA MEJORAR LA FIABILIDAD DE LA APLICACIÓN SOFTWARE DE PRÉSTAMOS BANCARIOS”**, teniendo como objetivo mejorar la fiabilidad de la aplicación software de préstamos bancarios a través de la arquitectura basada en una capa de control de excepciones. Su finalidad es mejorar la tolerancia a fallos, tener un software maduro, registrar las incidencias de errores que ocurran en el proceso de ejecución y pueda ser reutilizado en los diversos proyectos que se desarrollen en java.

Palabras Clave: Arquitectura de Software, Fiabilidad de Software, Préstamos bancarios

ABSTRACT

The present research entitled: **"ARCHITECTURE BASED ON AN EXCEPTION CONTROL CAPACITY TO IMPROVE THE RELIABILITY OF THE BANK LOAN SOFTWARE APPLICATION"**, aiming to improve the reliability of the banking loan software application through the layer-based architecture Exception control. Its purpose is to improve fault tolerance, have mature software, record the incidences of errors that occur in the execution process and can be reused in the various projects that are developed in Java.

Key Words: Software Architecture, Software Reliability, Bank Loans

I. INTRODUCCIÓN

1.1. Realidad Problemática

Los requerimientos en la creación de un software cada vez son más complejos tanto funcionales como no funcionales. Estos conjuntos de requerimientos se han convertido en el término *"arquitectura de la información"* el cual tiene relación con usabilidad y accesibilidad (Hassan, Martín e Iazza, 2004). El desarrollo y técnicas de los dos son diferentes pero tienen el mismo objetivo, que sea satisfactoria la interacción entre usuario y sistema.

El utilizar una arquitectura a través del tiempo ha permitido agilizar el desarrollo, unificar criterios, encapsular comportamientos que tengan en común, permite identificar y reducir los riesgos técnicos.

Existe sistemas de aplicación de software que están implementados en patrones de arquitectura tanto para web como para escritorio como modelo, vista, controlador, N-Capas entre otros más, todo esto permite tener una programación que está cumpliendo con una estructura de calidad de software

Las diversas empresas que desarrollan software, tienen área de testing, el cual se encargan de la "calidad del software" que permite minimizar los defectos del software. El error que cometen las organizaciones comúnmente es solo realizar pruebas unitarias en el desarrollo y es insuficiente según Marcelo Malluzzo (2013), en su experiencia y datos recopilados de las industrias, solo se minimizan un 30% de fallos.

El software critico se ha duplicado en estos años y es más complicado, por ende influye en *"la fiabilidad y seguridad"* de los sistemas. Lo que se requiere es sistemas críticos que contengan alto contenido de software, deben funcionar correctamente para lo que ha sido diseñado, con respecto a su entorno. Se han implementado funciones críticas en el sistema que pueda detectar peligros y mitigar las consecuencias de posibles accidentes, debe de exigirse que funcione correctamente y además pueda ser segura y ser controlados a cambios no esperados, según la revista española REICIS (2009).

El manejo de excepciones tiene ciertos beneficios: si no se tienen información del problema que está ocurriendo para ser solucionado deriva a alguien superior para que se encargue de tomar una decisión adecuada, como crear una cadena de comandos, limpia el código en el manejo de errores, puesto que una excepción garantiza que el error sea capturado en tiempo de ejecución, permitiendo que la escritura, lectura y depuración de código sea más sencilla a diferencia con la antigua forma de manejar los errores.

Las empresas que adquieren software para sus negocios cada vez crecen en la prestación de sus servicios de diferentes maneras y el avance de la tecnología no se queda atrás, las empresas necesitan sistemas de software confiable, fiable, seguro para prestar sus servicios a la sociedad.

En el proceso de desarrollo de software siempre nos encontramos con errores o dificultades que vamos teniendo cuando nuestra programación se va haciendo cada vez más amplia y se vuelve más dificultoso ver las fallas y es difícil desarrollar un software más confiable, usable y portable. El utilizar casos de pruebas permite evaluar nuestros sistemas de aplicación y ayuda a mejorar los errores y poder sacar un resultado esperado, el cliente siempre espera que su aplicación sea de calidad.

Los diferentes libros de lenguajes de programación, prácticamente del tema tratan muy poco y los manuales limitan a proporcionar una serie de instrucciones de manejo de las excepciones, pero no profundizan en las técnicas de aplicación de que se debe utilizar, aun cuando estamos desarrollando en base a una arquitectura de diseño. Las únicas propuestas a las que se pueden acoger los desarrolladores son en base a artículos, manuales o información que existe en la web.

Los diversos sistemas de aplicación de software que son desarrollados por las empresas en diversas partes del continente, son desarrollados para otras empresas, que luego ellos mismos lo prueban el producto, los errores que frecuentemente han tenido como resultado son: la calidad es insuficiente, incorporar nuevos requerimiento en la mitad de desarrollo del proyecto les

dificulta, el control de errores es insuficiente, entregando proyectos con dificultades.

El prototipo de software de préstamos estudiado está basado en un diseño de arquitectura que permite visualizar cuales son los requerimientos y como están distribuidos por capas y desarrollado vía web, encontramos que sigue una estructura de calidad de software pero también encontramos problemas que tienen en la estructuración de gestionar los errores en las diferentes capas de los cuales mencionamos:

- La declaración de variables para capturar las excepciones en tiempo de ejecución está mezclado con el código implementado en el desarrollo, teniendo dificultades para entender y el manejo de excepciones que son declaradas en las capas no se puede reutilizar en otras aplicaciones.
- Permite ingresar datos numéricos en los campos del formulario cuando solo debe de estar permitido ingresar datos con letras en el software, permitiendo guardar datos incoherentes y mostrar información errónea Permite guardar datos sin ninguna restricción y muestra errores de ejecución saliéndose totalmente de la interfaz gráfica

  Se crearon y ejecutaron casos de pruebas funcionales y evaluadas con métricas de fiabilidad obteniendo los siguientes resultados

  - El 67% nos indica que el sistema es tolerante a fallos pero todavía es marginal el sistema.
  - El 33% nos indica que tiene una madurez de pruebas pero todavía es insatisfactoria las pruebas ejecutadas.
  - El 100% no indica que la cobertura de pruebas se realizó satisfactoriamente.

1.2. Trabajos previos

- *(Marco, 2012)* escribe en su artículo *"Tratamiento de excepciones en Java" el cual hace un enfoque, como el manejo de excepciones permite gestionar los errores y situaciones excepcionales. Debido a que tratar con excepciones son pilares fundamentales del lenguaje, explica que los programadores saben cómo lanzarlas y capturarlas, pero se debe de conocer con profundidad el propósito porque tenemos dicho mecanismo y hacer un correcto uso de esta funcionalidad. Permitiendo que con el manejo de excepciones se puede recuperar de errores y continuar en la ejecución, se puede registrar el error y relanzar el error con una distinta excepción. Contribuye con la forma de declarar las excepciones y poder capturarlos, como crear excepciones desde cero y poder adecuarlo a la arquitectura utilizada.*

- Jorge Asiain Sastre (2013) "*es gerente de alter Evo Ingenieros y profesor en la escuela politécnica de Madrid* "escribe en su artículo *sobre "ingeniería de fiabilidad",* que combina las experiencias prácticas con los conocimientos de la matemática, física e ingeniería. Que en la realización de un estudio, los tres mencionados permiten crear modelos para profundizar conocimientos, estudiar los problemas y causas y permite dar soluciones en sus rendimientos, mitigar los fallos que existan y costos de operaciones. llegando a la conclusión para lograr una fiabilidad con el software debe seguir ciertos pasos:
    - Los objetivos deben estar basados en las expectativas del cliente y deben ser comprendidos.
    - La fiabilidad debe formar parte del proceso integral.
    - Todo producto debe de ser estudiado en el desarrollo de trabajo y tomar en cuenta todos los fallos encontrados.
    - Verificar y comprobar la fiabilidad para ver si se está cumpliendo con los objetivos obtenidos al principio.

- Revista Cubana de Ciencias Informáticas: escribe en su artículo el tema *(Aplicando métricas de calidad a proyectos y procesos durante las pruebas exploratorias, 2013)* se enfoca en cuatro razones que son evaluar, caracterizar, predecir y mejorar, el departamento de pruebas de software incorpora pruebas exploratorias y utiliza *"métricas internas y externas para evaluar el software".* Durante *"el proceso de prueba se puede*" utilizar la mediciones: "Monitorizar el proceso de pruebas", reportar las pruebas, controlar las pruebas, el departamento de pruebas de Calisoft (MIC) realiza diferentes tipos de pruebas que se ejecutan en: Funcionalidad, Fiabilidad, Usabilidad y Revisión técnica de Documentación, utilizando como referencia *"atributos de McCall (Macías Gómez, 2010) y la norma ISO 9126 (ONN, 2005)",* se evaluaron 42 proyectos que tuvieron la información y datos que eran necesarios para medir las métricas, obteniendo como resultado la efectividad del proceso de pruebas en la evaluación del producto, definir y seleccionar métricas en base a la calidad según la ISO 9126-1 que permite obtener un porcentaje de cumplimiento de las características de calidad y son: *"cobertura de pruebas 99%, madurez de pruebas 45.4%, total de fallos 6.8%, adecuación funcional 62.2%",* eficiencia de documentación de usuario 83.7% *"y todos estos resultados ayudan a que los directivos de la empresa tomen decisiones y tengan una mejor calidad en las pruebas y del producto"*.

  Contribuyendo al proyecto de investigación con las métricas y forma de medición del software.

1.3. Teorías relacionadas al tema

1.3.1. Arquitectura de software

1.3.1.1. Definición.

Según (Pressman, 2010) lo define como la estructuración de los componentes del software, en donde están representados sus propiedades y todas las interacciones que lo conforman.

Según (Ariel Bensussán, 2016) lo define como un proceso en el cual definen y se construye *"capas, lineamientos, estructuras"* que deberían tener una aplicación, con el cual se logra una estabilidad en la aplicación y poder continuar con las diferentes tareas durante el *"desarrollo del software".*

1.3.1.2 Importancia de la arquitectura

Se puede identificar tres razones:

- Utilizar una arquitectura de software permite que todas las partes puedan comunicarse en el desarrollo del software.
- En una arquitectura se tomaran todas las decisiones resaltantes que se tendrán en el trabajo, en el desarrollo del software propuesto y es importante para el éxito del sistema.
- El utilizar una arquitectura constituye un modelo, en donde está la forma como los componentes trabajan juntos y ver cómo está estructurado el sistema a desarrollar.

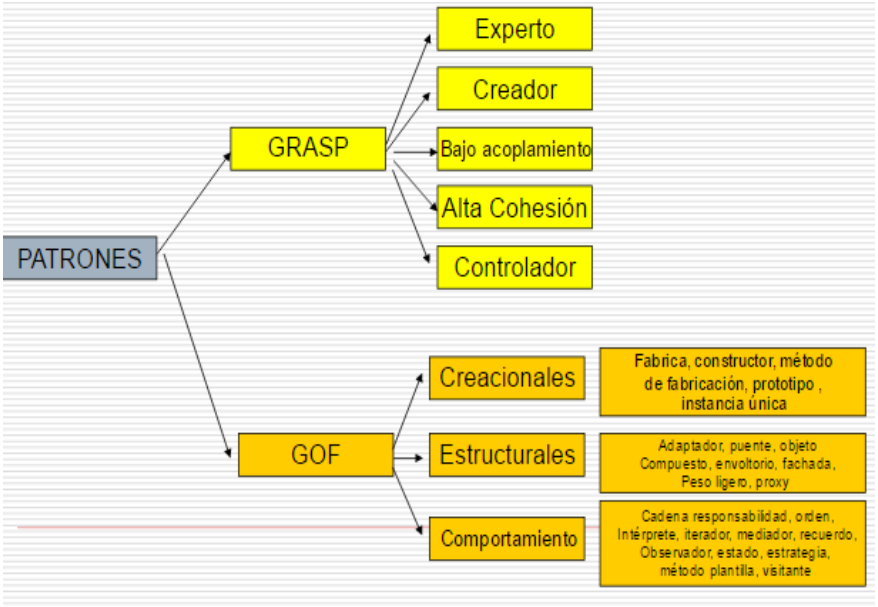
1.3.1.3. Patrones de arquitectura

Mientras se obtiene los requerimientos, el software se va a enfrentar a un número de problemas que puede tener una aplicación, los patrones están enfocados en un problema específico de una aplicación y tienen restricciones y limitaciones, al utilizar un patrón significa proponer una alternativa de solución arquitectónica, que va a servir como base en el diseño de una arquitectura.

- ✓ **Patrón de arquitectura MVC** (“*Modelo Vista Controlador*”)

Utiliza la realización de programación multicapa, está separado en tres y son datos de la aplicación, interfaz del usuario y lógica del negocio. Mayormente se utiliza en aplicaciones de servicio web.

- ✓ **Patrón de diseño GRASP** (“*Patrones Generales de Software para Asignar Responsabilidades”*) **y GOF**.
  Representan principios básicos de asignación de responsabilidades en los objetos y son:



Cuadro de principios

1.3.1.4. Decisiones de una arquitectura

Según (Sommervalle, 2011) el utilizar un diseño arquitectónico permite diseñar a la organización que cubren *“requerimientos funcionales y no funcionales”*. El enfoque que se puede usar permite implementar *“diferentes tipos de arquitectura”*. En el proceso de modelo se tomaran decisiones como se controlara *“la ejecución de componentes”, dependiendo de los requerimientos no funcionales el estilo y la estructura es:*

- **Con respecto a rendimiento.** Si es crítico se debería diseñar para localizar situaciones críticas dentro del *"número de componentes" permitiendo que los componentes sean desplegables en la* computadora y significa utilizar ciertos *"componentes relativamente grandes"* y reduce las comunicaciones entre los componentes.
- **Con respecto a seguridad.** *Si es crítico debería ser necesario estructúralo en capas con los procesos más críticos y protegerlos en capas más internas y tener un alto nivel en validación de seguridad que se aplicara a dichas capas.*
- ***Con respecto a*** **protección.** Si también es crítico, debería diseñarse de modo que todas las operaciones que tengan relación con la protección se debe de ubicar en *"algún componente individual".* Esto reduciría los costos y los problemas de la protección y se haría posible que se ofrezca, *"sistemas de protección"* y si en caso pudiera fallar se desactive con seguridad.
- **Con respecto a disponibilidad.** El caso que sea crítico, se tendría que diseñar para contener componentes redundantes para poder actualizar y sustituir componentes sin afectar al sistema.
- **Con respecto a mantenibilidad.** Si se considera crítico, se diseñara utilizando componentes de grano fino y se pueda cambiar con facilidad. Se lograra evitar que no se pueda compartir datos.

1.3.1.5. Ventajas de una arquitectura

- Permite agilizar el desarrollo.
- Permite unificar criterios.
- Encapsular comportamientos que tengan en común.
- Permite identificar y reducir los riesgos técnicos.

1.3.2. Control de excepciones

1.3.2.1. Definición

Según (Marco, 2012) lo define como una técnica que permite controlar errores durante la ejecución en un software, y *"es un mecanismo del lenguaje que permite gestionar errores y situaciones excepcionales".*

El trabajar con errores o excepciones *es "uno de los pilares fundamentales del lenguaje",* para poder mejorar el manejo de errores es necesario profundizar y poder "*hacer un uso correcto de esta funcionalidad".*

1.3.2.2. Definición de Excepciones

Según (Sommerville, 2005) lo define como *"evento que ocurren durante la ejecución de un programa",* permitiendo que salga de su flujo normal en las instrucciones definidas.

Las excepciones permiten ser lanzadas cuando se produce un error, y puede ser capturado el mensaje y tratar el error que ha ocurrido.

1.3.2.3. Tipos de excepciones.

Existen diferentes tipos de excepciones que descienden de la clase Throwable del cual son los siguientes.

Error. Son errores que ocurren dentro de una máquina virtual por ejemplo problemas con enlazar con librerías.

Exception. Trata los errores y permite continuar con la ejecución. Los programas utilizan las excepciones para dar tratamiento a los errores de código que ocurren durante una ejecución.

1.3.3.4. Ventajas.

Se puede separar los errores del código en la programación y se podrá tratar como código especial.

Se puede propagar los errores hacia la pila de llamadas, es una habilidad de las excepciones, que se busca el error hasta encontrarlo.

Permite agrupar y diferenciar los diferentes tipos de errores lo que es natural de la jerarquía de clases.

1.3.3.5. Tratamiento de Excepciones.

Podemos generar el código de la excepción y encerrarse dentro de un bloque try

```
Try
{
// Ingresar procedimiento a evaluar.
}
```

Después la excepción se captura con un bloque catch

```
Catch (Exception e)
{
//ejecución para tratar los errores
}
```

Exception (). Es un constructor sin parámetros.

Exception (String Message). Es un segundo constructor, que utiliza parámetros de tipo String, en donde podemos crear excepciones y mostrar mensajes.

1.3.3.6. Lanzamiento de excepciones

Después de tratar las excepciones también se puede hacer que se propague a un nivel superior. Indicando de la siguiente manera:

```
Public void leer_datos ()
   throws IOException, FileNotFoundException
{
   // Especificar la función
}
```

Se podría indicar diversos tipos que queramos en la función.

1.3.3. Fiabilidad del software

1.3.3.1. Definición de fiabilidad

Según (Alan Burns, 2003) define como una medida del éxito con el que el sistema se ajusta a alguna especificación definitiva de su comportamiento.

Según (Sommerville, 2005) lo define como *"la probalidad de que un sistema funcione correctamente tal y como se ha especificado",* en un tiempo y entorno determinado.

1.3.3.2. Validación de la fiabilidad

El proceso para verificación de la fiabilidad sigue cuatros pasos que son:

- Identificar todos los perfiles operacionales.
- Preparar un conjunto de datos de prueba.
- Aplicar las pruebas.
- Calcular la fiabilidad para obtener resultados específicos.

Así como se puede verificar también presenta dificultades que son:

- Incertidumbre con respecto al perfil operacional.
- Los datos de prueba que son generados tienen elevados costos.
- Si la fiabilidad especificada es alta es bien difícil provocar nuevos fallos.

1.3.3.3. Predicción de la fiabilidad

Según (Sommerville, 2005) el utilizar la predicción de la fiabilidad tiene dos ventajas principales:

- **Planificación de pruebas.** Si se tiene un calendario de todas las pruebas propuestas, se puede predecir cuándo se terminaran las pruebas. Si no se termina las pruebas conforme al calendario planificado, entonces se utilizaran recursos adicionales para realizar las pruebas y depurar y así acelerar el crecimiento de la fiabilidad.
- **Negociaciones con el cliente.** En algunas ocasiones el modelo diseñado de fiabilidad puede ser muy lento, y requiere un esfuerzo mayor de pruebas, *"para obtener un beneficio relativamente pequeño",* se puede renegociar los requerimientos con el cliente.

1.3.3.4. Especificaciones de fiabilidad

Según (Sommervalle, 2011) para lograr una fiabilidad requerida el diseño de sistema se debe especificar cuáles son *"los requerimientos funcionales"* y cuáles son las fallas y acciones que se tomaran para garantizar que no provoque fallas en el sistema. Las especificaciones se pueden basar en especificaciones por riesgo.

- **Identificaciones del riesgo.** Se examinan los diferentes tipos de fallas en un sistema que provocarían pérdidas económicas. Teniendo como ejemplo perdida de información durante en realización de *"ventas por internet".*
- **Análisis del riesgo.** Tiene relación con las consecuencias y costos de diferentes fallas del software y poder seleccionar un análisis de las fallas graves.
- **Descomposiciones del riesgo.** Se realizaran análisis de causa raíz graves y de fallas probables del sistema.
- **Reducciones del riesgo.** Se generara especificación cuantitativa con respecto a la fiabilidad que pueda establecer *"probabilidades aceptables"* de las diferentes fallas. Luego se tomaran en cuenta los costos por las fallas. Se podrían utilizar diferentes probabilidades, se podría generar *"requerimientos de fiabilidad funcional".*

1.3.3.5. Métricas de fiabilidad

Según (Sommervalle, 2011) las métricas que se utilizan son:

- **Probabilidad de falla ha pedido.** Esta métrica se usa para definir la probabilidad en que la demanda de un servicio derive por una falla del sistema.
- **Tasa de ocurrencia a fallos.** Se utiliza esta métrica para establecer el probable número de fallas que se observan en el sistema en cierto tiempo o el número de cuantas veces se ha ejecutado el sistema.

- **Disponibilidad.** Se refleja cuando se entrega un servicio solicitado, debe de estar disponible un 99.9% en el *"tiempo de operación".*

1.3.3.6. Fallos o Errores

Según (Alberto Tols, 2010) lo define como resultados de problemas en el interior de un sistema y son manifestados con el tiempo en su comportamiento al ser utilizados. Todos los problemas se consideran errores y las causas algorítmicas o mecánicas son denominados defectos, Si un componente del sistema es defectuoso producirá un error bajo ciertas circunstancias durante el tiempo de vida del sistema (cap.5).

Pueden ocurrir tres tipos de fallos o errores.

- **Fallos o errores transitorios.** Pueden comenzar en cualquier instante de tiempo y mantenerse en el sistema por algún periodo y después puede desaparecer.
- **Fallos o errores permanentes.** Aparecen en un determinado tiempo y permanecen en el sistema hasta que el error es reparado.
- **Fallos o errores intermitentes.** Son errores transitorios que pueden ocurrir cada cierto tiempo.

1.3.3.7. Seguridad y fiabilidad en el software

Según (Alan Burns, 2003) la seguridad en el software se puede considerar en término de percance, un percance viene a ser eventos no planificados o secuencias de eventos que producen lesión, muerte, perdida de equipos o perjudicial *"en el medio ambiente".*

Tanto fiabilidad como seguridad viene a ser como sinónimos pero existe un enfoque diferente entre ellos. La fiabilidad es considerada una *"medida de éxito"* en donde un sistema tiene que ajustarse a especificaciones se su comportamiento y esto puede expresarse en término de probabilidad. La seguridad viene a ser la improbabilidad

de las condiciones conduzcan a un percance, independientemente si se puede realizar una función prevista.

1.3.3.8. Confiabilidad y fiabilidad

Según (Magallanes, 2011) lo define como la habilidad de un sistema que debe operar bajo ciertas condiciones requeridas durante un tiempo definido.

Muchas investigaciones se han realizado sobre fiabilidad o tolerancia a fallos. Consecuentemente los términos han sido sobrecargados y los investigadores han tenido que buscar *"nuevas palabras"* para poder expresar lo que querían enfatizar. La confiabilidad tiene ciertas nociones de seguridad y fiabilidad. Según Laprie (1995), ilustra los aspectos de confiabilidad donde la seguridad está enfocada en la *"integridad y confidencialidad"* y la fiabilidad es medir continuamente la entrega de un servicio adecuado. *La confiabilidad puede describirse bajo tres componentes.*

- **Deficiencias**. Son *"circunstancias causales"* o productos que no son confiables.
- **Medios.** Las herramientas, métodos y soluciones que se requieren *"para entregar un servicio"* confiable con la confiabilidad requerida.
- **Atributos**. Las medidas o el modo mediante el cual pueden estimarse la calidad de los servicios confiables.

1.3.3.9. Disponibilidad

Según (Magallanes, 2011) lo define como *"la probalidad de que un ítem",* que cuando sea utilizado bajo ciertas condiciones en un ambiente acondicionado, estuviera operativo.

1.3.4. Software de Préstamos

Es una herramienta que permite gestionar cualquier tipo de préstamo, esta implementado para utilizarse en una proforma vía web, desde cualquier lugar debe de ser utilizado.

1.3.5. Metodología

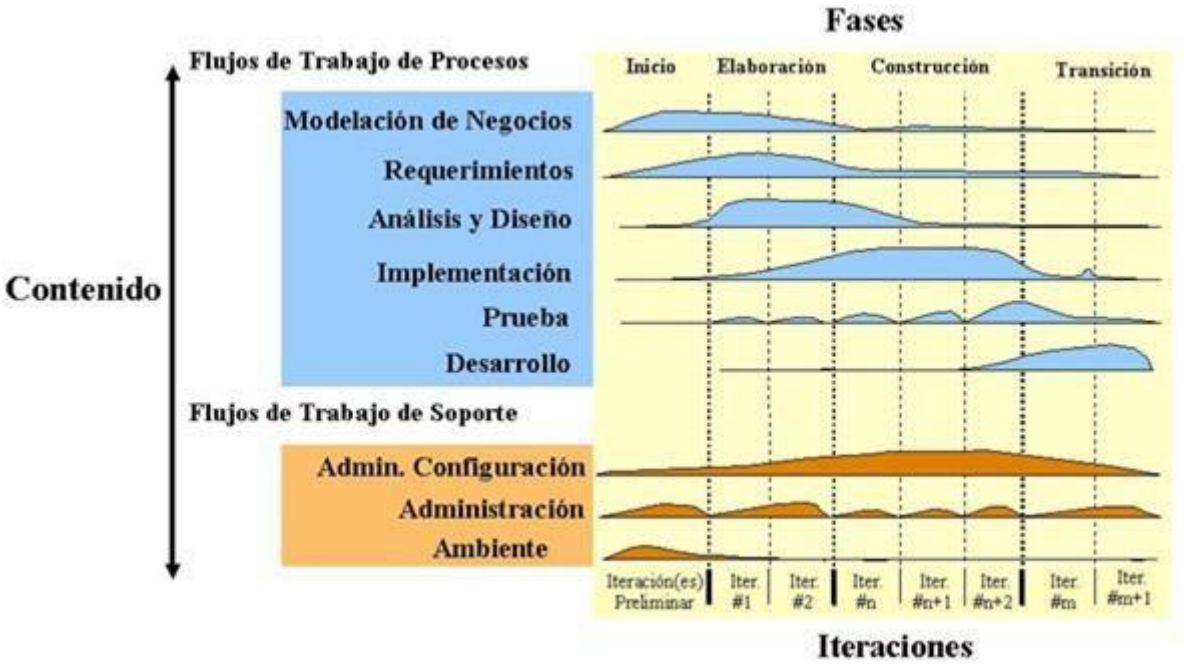
1.3.5.1. RUP (RATIONAL UNIFIED PROCESS)

Tiene un enfoque que asigna tareas y responsabilidades en una organización. Asegura que el software sea producido de *"alta calidad"* utiliza patrones y ofrece el soporte de UML.

Fuente: http://ima.udg.edu/~sellares/EINF-ES2/Present1011/MetodoPesadesRUP.pdf

Tiene algunas características:

- Ser Iterativo e incremental
- Tener un modelado del software
- Debe de administrase los requisitos
- Tener un modelo visual del producto
- Ser verificable la calidad



Proceso de desarrollo

1.3.5.2. Fases que utiliza RUP

1.3.5.2.1. Inicio.

- Define el entorno del negocio, el alcance, identifica los actores, casos de uso y se diseñan algunos casos de uso más significativos.
- Modelo de dominio. Identificar los objetos y cosas de la realidad que vas a intervenir en el sistema.
- Modelo casos de uso. Diseña el comportamiento de los usuarios dentro del sistema.
- Prototipo interfaz de usuario. se creara un modelo operativo de trabajo del sistema y deben estar de acuerdo tanto analistas como clientes.

1.3.5.2.2. Elaboración

Se crea una arquitectura del sistema en base a casos de uso y son desarrollados en este tramo, toma sus especificaciones y realiza un análisis del problema.

1.3.5.2.3. Construcción

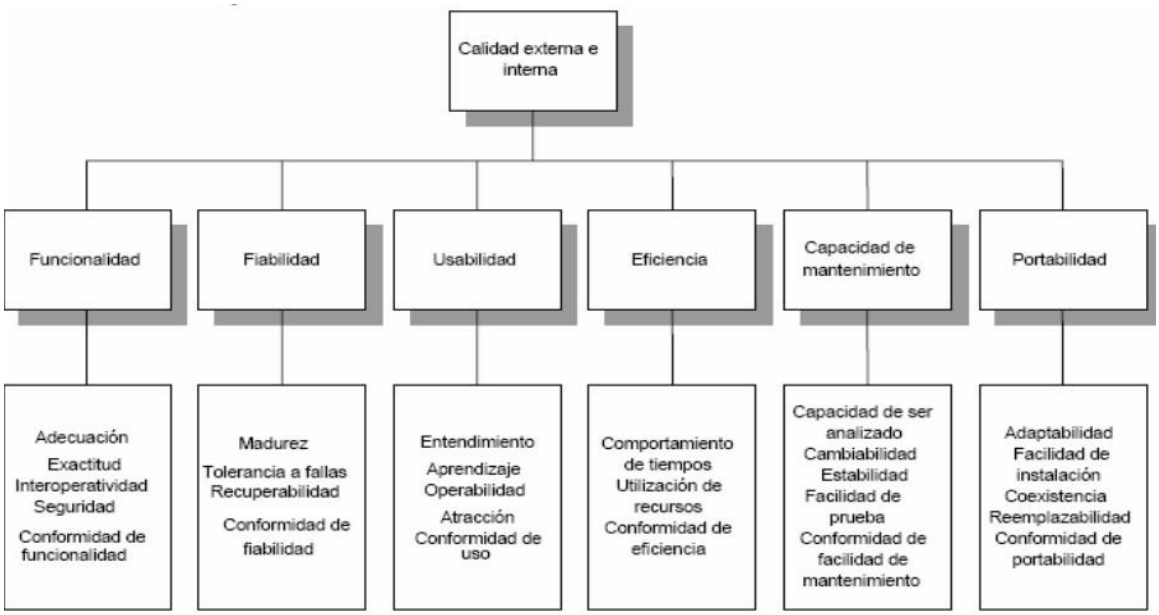
Debe lograrse un producto operacional a través de las iteraciones de ciertos casos de uso seleccionados, los componentes, requisitos y características deben implementarse, probarse y obtener un resultado de su primera versión del software.

1.3.5.2.4. Transición

Poner a prueba el producto desarrollado en el uso del usuario para corregir los errores y crear versiones, se debe de completar algunos documentos faltantes, capacitar, configurar, instalar y que cumpla con los requerimientos especificados. Permite la facilidad como el software sea modificado. Testeo de unidades, datos, casos y resultados. Test en conjunto con los usuarios para saber su aceptación.

1.3.6. NORMA ISO/IEC 9126.

Es una norma internacional publicado en 1992, el cual se utiliza para evaluar la calidad del software y tiene 6 características en general.



Fuente: ISO/IEC 9126-1

Fuente: https://jrvargas.files.wordpress.com/2009/03/guia_tecnica_para_evaluacion_de_software.pdf

http://www.austral.edu.ar/aplic/webSIA/webSIA2004.nsf/6905fd7e3ce10eca03256e0b0056c5b9/9decac133ffd010283257656004f60a5/$FILE/Norma%20ISO%209126.pdf

1.3.7 Framework log4j.

Es una herramienta que se utiliza para la captura de logging y se utiliza con el lenguaje de java y es configurable en cualquier proyecto. Se ha implementado en C, C++, C#, Perl, Python, Ruby y Eiffel. Entre sus características el usar y entender son fácil, contiene niveles de clasificar un error, tiene 3 componentes que son loggers, appenders y layouts el cual los tres juntos permiten capturar los log dependiendo el tipo.

Fuente: http://migranitodejava.blogspot.pe/2011/07/log4j.html

1.4. Formulación del problema

¿De qué manera la arquitectura basada en una capa de control de excepciones mejora en la fiabilidad de la aplicación software de préstamos bancarios?

1.5. Justificación del estudio

Desde el punto de vista teórico y práctico el presente proyecto de investigación busca información sobre formas de implementación de manejo de excepciones, conceptos de arquitectura, fiabilidad y de software, que luego unir los diversos conceptos y explicar cómo mejorar un software con el manejo de excepciones en el código, cuando se está desarrollando el software y por consiguiente se debe de obtener un software más fiable y de calidad. El cual se implementara la capa de excepciones en el software y se podrá reutilizar en las diferentes capas de diseño de la arquitectura y acoplar con otros proyectos basados en la misma tecnología y será beneficioso en ahorro de tiempo y recursos económicos para los desarrolladores de software, y les permitirá entender el código desarrollado con mayor facilidad a nuevos programadores que se integren al grupo de trabajo.

Desde el punto de vista tecnológico se utiliza la plataforma de Netbeans IDE el cual permite crear tecnologías tanto para web, escritorio y dispositivos móviles, da soporte a lenguaje de programación Java, PHP entre otros más y puede ser instalado en diferentes sistemas operativos tiene como características utilizar el uso de herramientas y reutilización de módulos, conexión con diferentes GBD permite tener una arquitectura escalable.

1.6. Hipótesis

La arquitectura basada en una capa de control de excepciones mejora la fiabilidad de la aplicación software de préstamos bancarios.

1.7. Objetivos.

1.7. 1.Objetivo General.

Mejorar la fiabilidad de la aplicación software de préstamos bancarios a través de la arquitectura basada en una capa de control de excepciones.

1.7. 2.Objetivos Específicos

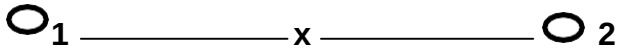
- Aumentar el índice de tolerancia a fallos mediante la capa de control de excepciones.
- Aumentar el grado de madurez de pruebas mediante la capa de control de excepciones.
- Aumentar el índice de cobertura de pruebas mediante la capa de control de excepciones.

II. METODO

2.1. Diseño

- El diseño a utilizar es experimental de tipo pre-experimental y para la realización de contrastación de hipótesis se va a utilizar el método de diseño Pre - prueba y post – prueba de lo cual se explica de la siguiente manera:

- Realizar un análisis previo a la variable dependiente y debe analizar variable efecto – después.
- Con respecto a la otra variable se evaluara la causa.
- Por último se evaluara con una nueva prueba la variable dependiente.



Estimulo

Dónde:

O1 = Fiabilidad de la aplicación software de préstamos bancarios antes de la arquitectura basada en una capa de control de excepciones.

X= Arquitectura basada en una capa de control de excepciones.

O2= Fiabilidad de la aplicación software de préstamos bancarios después de Implementación de la arquitectura basada en una capa de control de excepciones.

2.2. Variables y operacionalización de variables

Variable Independiente:

Arquitectura basada en una capa de control de excepciones.

Variable Dependiente:

Fiabilidad de la aplicación de software préstamos.

Operacionalización de variables

Variable	Definición conceptual	Definición operacional	indicadores	Escala de medición
Arquitectura basada en una capa de control de excepciones	Una técnica que permite controlar errores durante la ejecución en un software y *"es un mecanismo del lenguaje que permite gestionar errores y situaciones excepcionales"*. (Marco, 2012).	Describir el funcionamiento del manejo de excepciones en el software.	Capa de excepciones Diseño del manejo de excepciones .	Razón
Fiabilidad de la aplicación de software préstamos	Una medida del éxito con el que el sistema se ajusta a alguna especificación definitiva de su comportamiento. (Alan Burns, 2003).	Medir la tolerancia a fallos del software.	Porcentaje de Tolerancia de fallos.	Razón
		Medir la madurez del software.	Grado de Madurez de las pruebas.	

			Índice de cobertura de pruebas	

Indicadores

Indicador	Objetivo	Técnica	Calculo
Porcentaje de Tolerancia de fallos	Aumentar el índice de tolerancia a fallos mediante la capa de control de excepciones.	Casos de pruebas funcionales	Casos con errores / casos diseñados X=A/B
Grado de Madurez de las pruebas	Aumentar el grado de madurez de pruebas mediante la capa de control de excepciones.	Casos de pruebas funcionales	Número de casos satisfactorios / cosos diseñados X=A/B
Índice de cobertura de pruebas	Aumentar el índice de cobertura de pruebas mediante la capa de control de excepciones.	Casos de pruebas funcionales	Cosos ejecutados / cosos diseñados X=A/B

2.3. Población y muestra

2.3.1 población

La investigación tiene como unidad de estudio una aplicación software de préstamos bancarios y será evaluado en base a métricas según la normas ISO/IEC 9126.

Indicador 1: Porcentaje de Tolerancia de fallos.

Casos con errores / cosos diseñados

Indicador 2: Grado de Madurez de las pruebas.

Número de casos satisfactorios / casos diseñados

Indicador 3: Índice de cobertura de pruebas.

Cosos ejecutados / cosos diseñados

2.4 Técnicas e instrumentos de recolección de datos, validez y confiabilidad

La elaboración de los casos de prueba funcionales ha sido evaluada y validada por expertos, quienes aprobaron para luego aplicar dichos casos de prueba en la aplicación de software. **(ANEXO 5 FORMATO DE VALIDACIÓN DE CASOS DE PRUEBA FUNCIONALES)**

Confiabilidad de casos de prueba.

		EVALUADORES			Σri	Promedio	Proporcion de Rangos	Pe
DIMENSIÓN	Casos de pruebas	1	2	3		ri	de cada item	
I	1, 2, 3, 4, 5,6,7,8 9, 10,11,12,13, 14,15	3	3	3	9	3.00	1.00	0.04
	TOTALES	3	3	3		3.00	1.00	0.04

CPR	1.00
CPRc	0.96
J= 3 (Numero de expertos)	
k: 3	

Tabla 1: Validez y confiabilidad

		N°	%
Casos	Válido	15	100,0
	Excluido	0	,0

Total	15	100,0

Tabla 3: Casos de pruebas en porcentajes.

En la ilustración muestra el porcentaje al 100% de pruebas.

Método: Coeficiente de Proporción de Rangos	N de elementos
0.96	15

Tabla 4: Estadística de fiabilidad

En el cuadro se representa las estadísticas de fiabilidad del instrumento aplicado al software, el método: Coeficiente de Proporción de Rangos es 0.96 y según la escala de valoración de Coeficiente de Proporción de Rangos para este instrumento la apreciación de confiabilidad es Alta.

INTERPRETACION		
Mayor que	**Menor igual que**	**Validez y concordancia**
0	0.4	Baja
0.4	0.6	Moderada
0.6	0.8	Alta
0.8	1	Muy Alta

tabla 5: Escala de Valoración método: Coeficiente de Proporción de Rangos

2.5 Métodos de análisis de datos

En esta investigación se está evaluando un software como unidad de estudio.

- Prueba de Hipótesis
  - ✓ No Paramétrica

Se utilizara una norma ISO/IEC 9126.

Hipótesis Nula

$H_0 : \mu_B - \mu_A = 0$

La arquitectura basada en una capa de control de excepciones no mejora la fiabilidad de la aplicación software de préstamos bancarios.

Hipótesis Alternativa

$H_1 : \mu_B - \mu_A > 0$

La arquitectura basada en una capa de control de excepciones mejora la fiabilidad de la aplicación software de préstamos bancarios.

III. RESULTADOS

3.1. Fase Inicio

3.1.1. Requerimientos no funcionales con respecto al manejo de excepciones.

- ✓ El sistema debe de registrar todos los errores que estén ocurriendo en tiempo de utilización.
- ✓ Las excepciones personalizadas debe de estar aparte del demás código.
- ✓ Las excepciones deben de ser utilizadas en las diferentes partes del desarrollo del sistema si así lo requiera.
- ✓ Los formularios del sistema deben de estar validados según los datos de ingreso.
- ✓ El sistema debe de ser tolerante a fallos.
- ✓ El sistema debe de tener una madurez.

3.1.2. Especificaciones suplementarias

Introducción

El presente documento propone un modelo para la correcta elaboración e implementación del Sistema de Préstamos. El modelo de negocio abarca en su mayoría desde el planeamiento de gestión de préstamos, control de excepciones, registro de clientes, hasta la correcta gestión de implementación del sistema tanto por el lado del desarrollador como del cliente.

El Primer Entregable, se realiza en tres pasos. El primero paso, es donde se estudia al software para saber cómo están desarrollados los casos de uso, y se elabora los diagramas del software. El segundo paso, se identifica los que interactúan con el sistema, El tercer paso, se realiza el diagrama de secuencia de excepciones y se concluye con la implementación de las excepciones y registro de errores en el siguiente entregable.

a) Confiabilidad

Código	**es_sup_1**
Nombre	Registro de fallas
Descripción	El sistema debe de registrar los errores durante el desarrollo.

Código	**es_sup_2**
Nombre	Tiempo disponible
Descripción	El sistema debe de estar disponible las 24 horas.

Código	**es_sup_3**
Nombre	Cobertura de pruebas
Descripción	El sistema debe de ser probado en el proceso del desarrollo y tener un archivo de cada prueba realizada.

b) Soportabilidad

Código	**es_sup_4**
Nombre	Compatibilidad del sistema del lado del Servidor.
Descripción	El sistema será compatible con Windows 7 ultimate

Código	**es_sup_5**
Nombre	Compatibilidad de sistema del lado del Cliente.
Descripción	El cliente del sistema debe ser soportado con Windows

	7 ultimate, el Sistema Gestor de Base de Datos Postgres y con el desarrollador Netbeans 8.0.1.

c) Restricciones de diseño

Código	**es_sup_6**
Nombre	Características de la PC del cliente
Descripción	El sistema debe operar en cualquier computador personal con procesador Intel Core DUO o superior, 4 GB de memoria RAM y disco duro de 500 GB

Código	**es_sup_7**
Nombre	Lenguaje de programación
Descripción	El lenguaje de programación a emplear para el desarrollo del nuevo diseño será Netbeans 8.0.1

Código	**es_sup_8**
Nombre	Motor de base de datos.
Descripción	El motor de base de datos PostgresSQL.

Código	**es_sup_9**
Nombre	Arquitectura Lógica
Descripción	Deberá considerarse en tres capas claramente definidas: Modelo, vista, Controlador.

d) Implementación

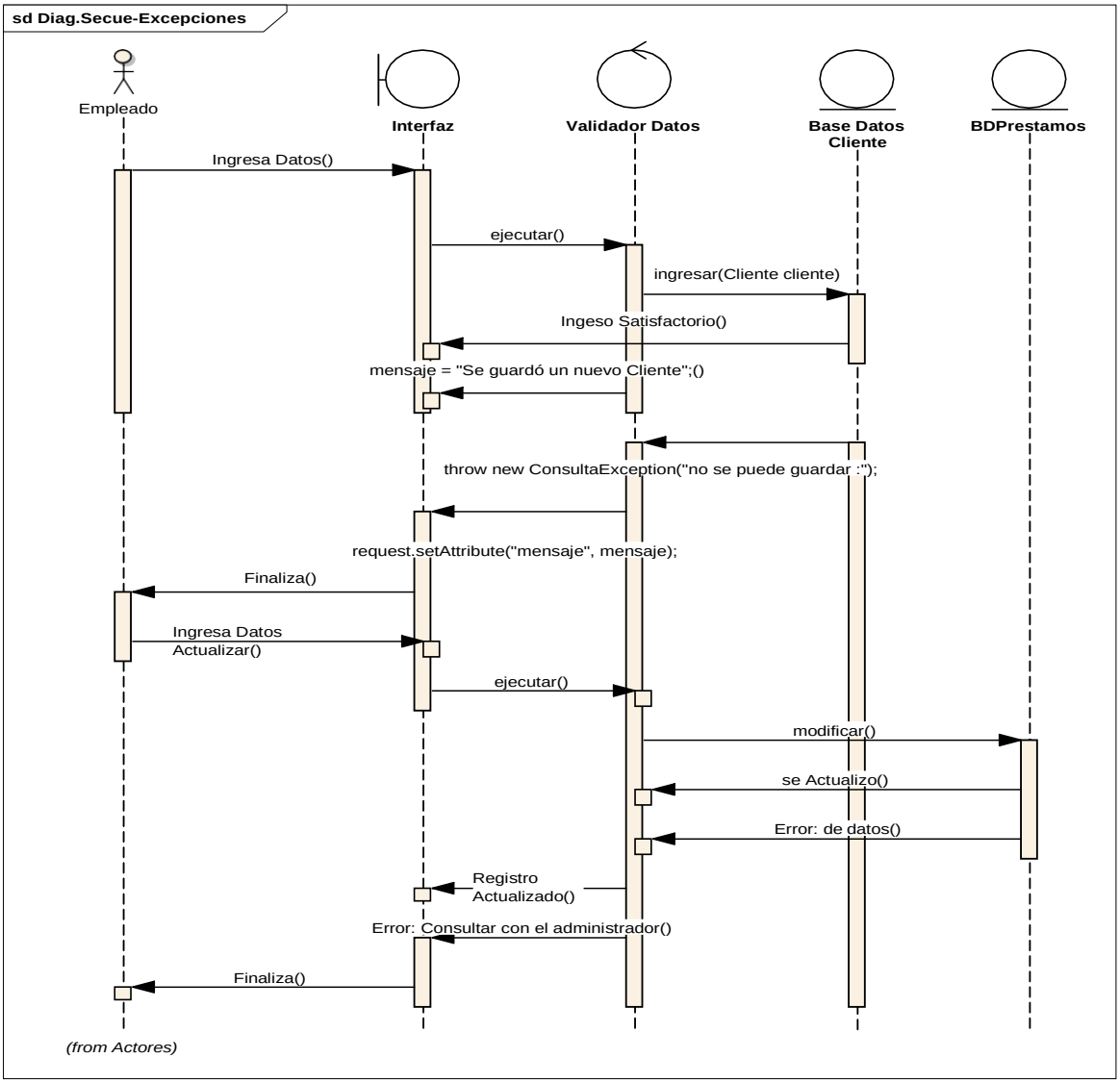
Código	**es_sup_10**
Nombre	Licencia del Sistema Operativo Ms Windows 7 ultímate
Descripción	Se necesitará adquirir una licencia del sistema operativo para servidor Ms Windows 7 ultímate

Código	**es_sup_11**
Nombre	Licencia del Motor de Base de Datos PostgresSQL.
Descripción	Se necesitará adquirir una licencia del software de motor de base de datos PostgresSQL.

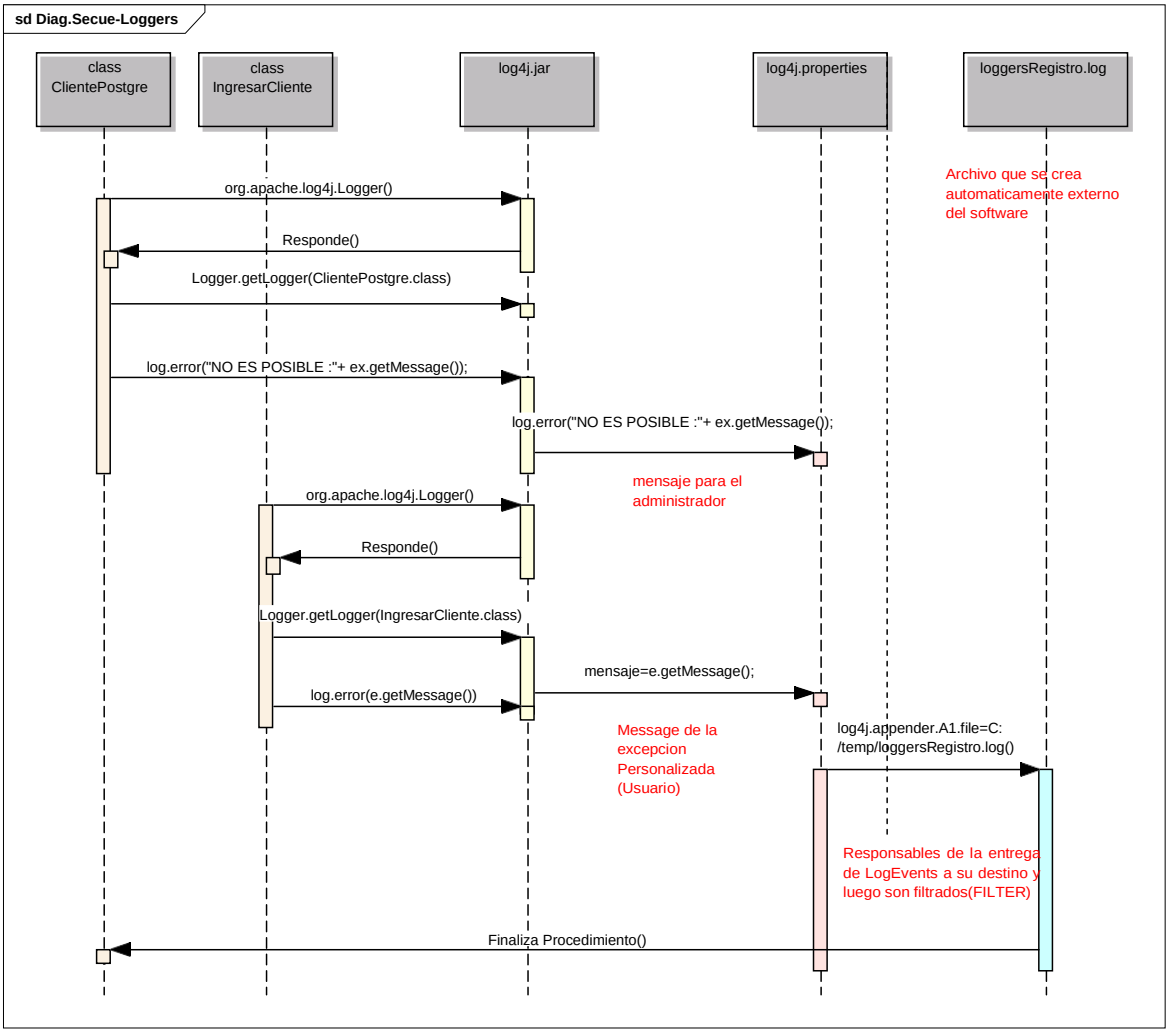
Código	**es_sup_12**
Nombre	Licencias para el equipo de desarrollo.
Descripción	Se deberán de comprar licencias de Netbeans para el equipo de desarrollo.

3.2. Fase de Elaboración

3.2.1. Diagrama de secuencia de Excepciones del caso de uso empleado



3.3.2. Diagrama de secuencia de Registro de loggers en el registro de un cliente.



3.4.3. Documento de Arquitectura de Software

Introducción

Este artefacto presenta vistas arquitectónicas de la aplicación Préstamos, las cuales representan los modelos más importantes del sistema guiados por los casos de uso arquitectónicamente más significativos. Esta arquitectura hace referencia a la estructura global del software y las formas en que esta estructura proporciona integridad conceptual al sistema materia de desarrollo. La descripción de la arquitectura es importante porque presenta o describe una estructura jerárquica de los módulos que presenta el sistema, la manera de interactuar de todos los componentes y la estructura de los datos presentes en cada módulo, para entender su total funcionamiento.

Vista de Casos de Uso

Diagrama de Casos de Uso

A continuación, la figura Nº 01 muestra el diagrama de casos de uso del sistema Préstamos, en donde se resaltan los dos casos de uso más importantes:

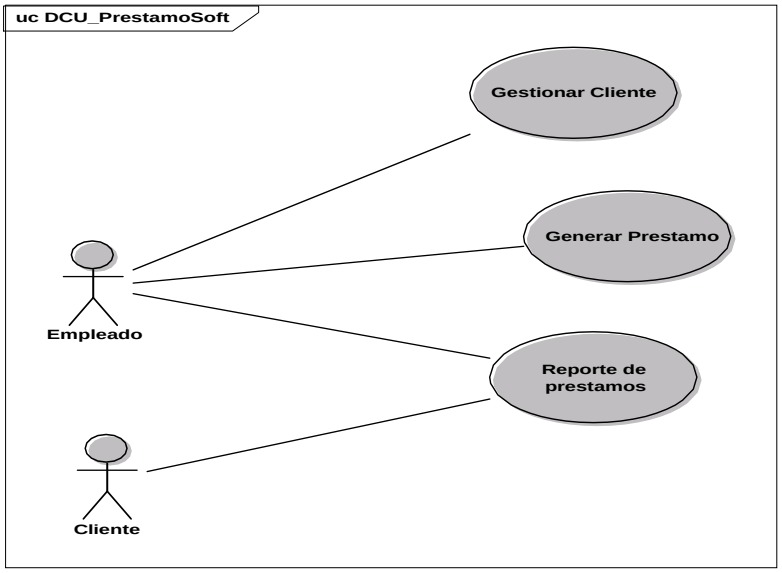


Figura 1: casos de uso del software

Casos de Uso relevantes a la Arquitectura

A continuación de muestra la tabla de priorización de los casos de uso. La tabla de priorización se ha elaborado en base a tres criterios:

CRITERIOS PARA VALORAR A LOS CASOS DE USO	PESO	RANGO
(RI) Riesgo tecnológico: complejo de implementar, nuevo, o puede ir variando debido a nuevas políticas o reglas de negocio.	3	0-3
(SA) Significativo para la arquitectura: debe cumplir características de calidad como fiabilidad, mantenibilidad, o rendimiento.	2	0-3
(NC) Naturaleza crítica: de valor para el negocio.	1	0-3

Tabla de priorización para el sistema **prestamos**:

CASO DE USO	RI	SA	NC	PUNTAJE
CU - Generar	3	3	3	18

Préstamo				
CU - Generar Reportes Clientes	2	1	2	15
CU - Gestionar Cliente	1	0	2	12

Tabla de priorización agrupado por niveles de prioridad:

CASO DE USO	PRIORIDAD	COMENTARIO
CU - Generar Préstamo	Alta	**Implementar en fase de Elaboración y construcción**
CU - Generar Reportes Diarios	media	**Implementar en fase de Construcción**
CU - Gestionar Cliente	baja	

Vista Lógica

Esquema general de la Arquitectura

A continuación se muestra la figura 2 que representa el diseño general de la Arquitectura de Software planteado para el Sistema Prestamos, esta arquitectura está basada en el **MVC**.

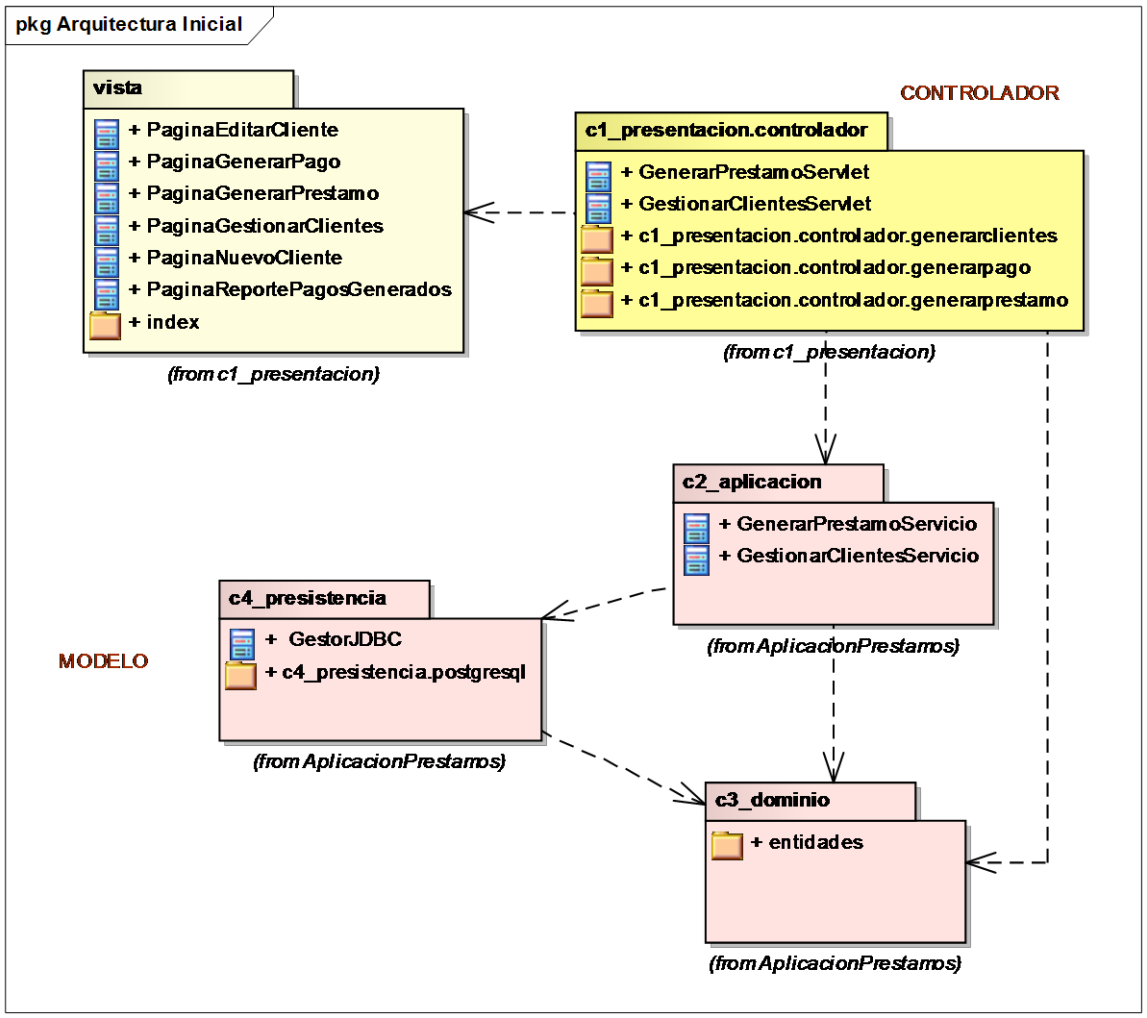


Figura 2: Esquema general de la Arquitectura de Software Inicial

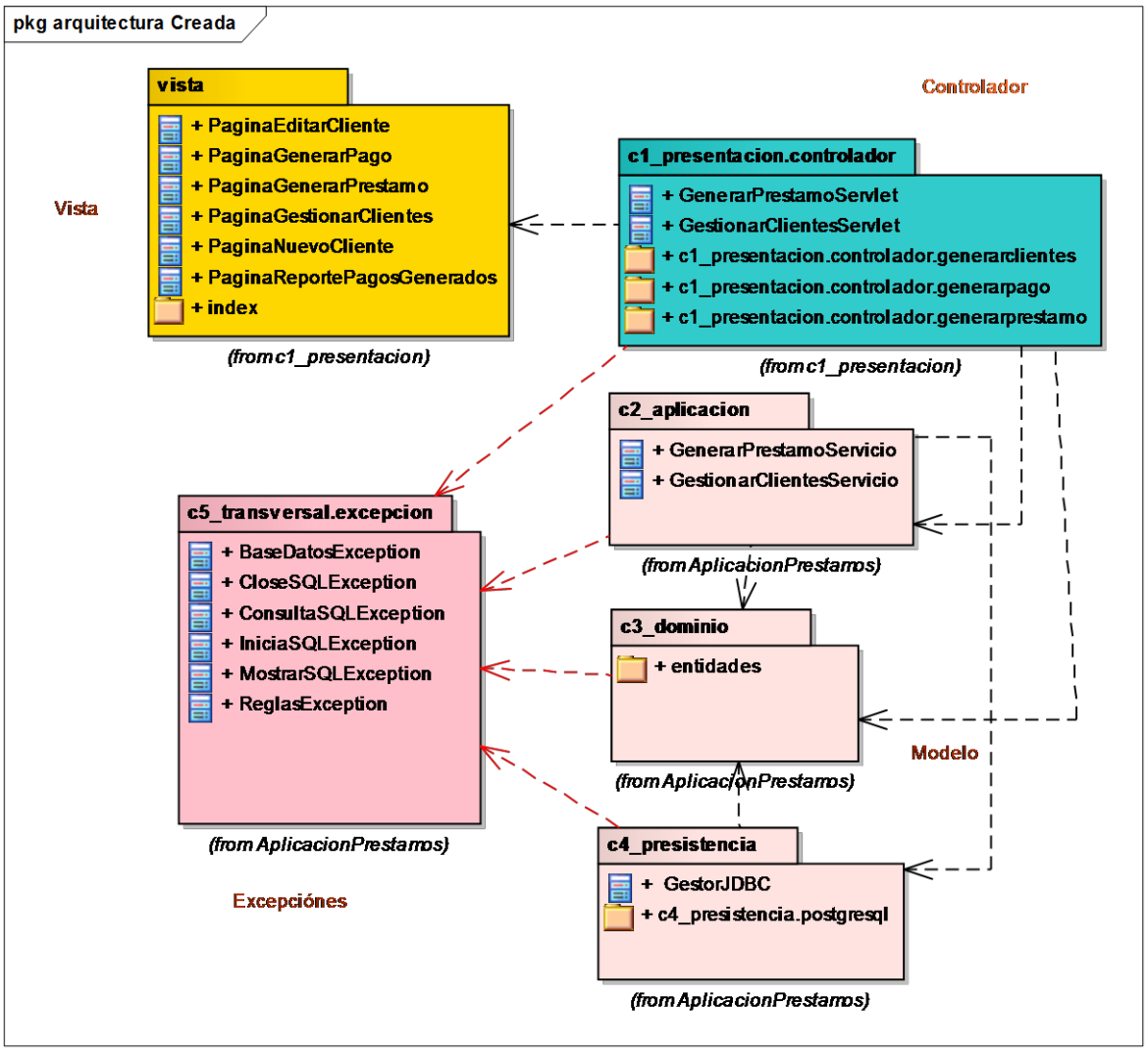


Figura 3: Diagrama general de la Arquitectura de Software Mejorado

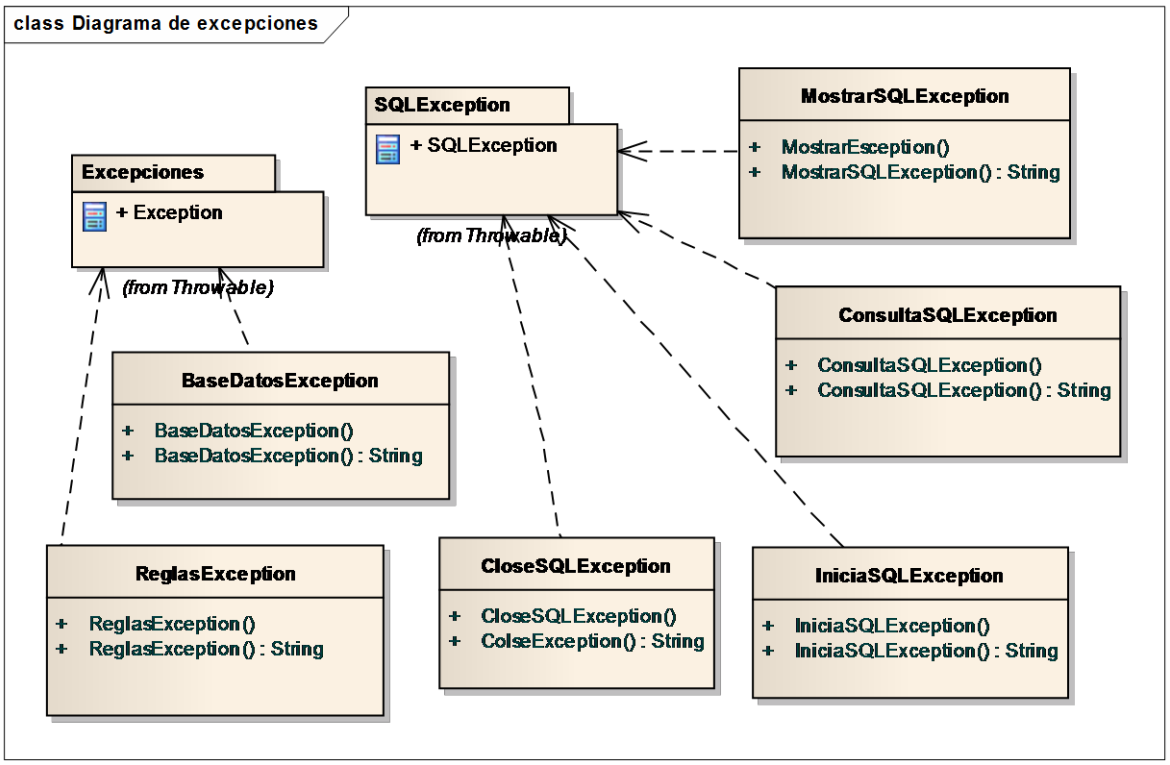


Figura 4: Esquema de la capa de excepciones implementada

Cómo funciona el framework log4j

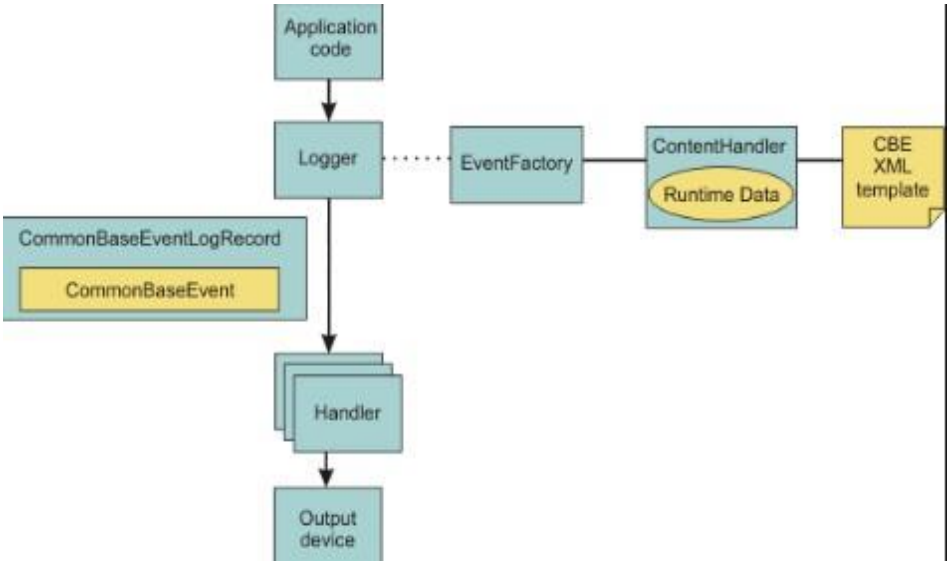


Figura 5: Esquema de framework log4j.

Permite capturar información de lo que está sucediendo cuando hacen una utilización del software tanto información como errores del sistema en tiempo de ejecución.

A. **Vista:** la figura 5 muestra las clases de la capa. Todas las clases de la capa de presentación o vista deben ser nombradas dependiendo las clases.

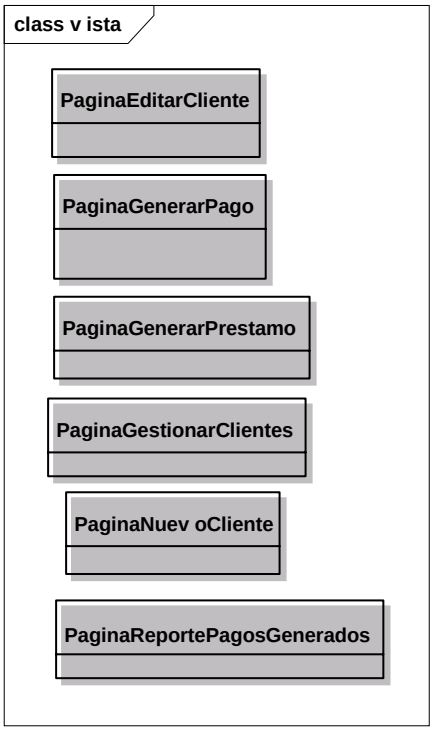


Figura 6: Diagrama de Clases parcial de la capa “vista”

B. **Controlador:** la figura 6 muestra las clases de la capa. Todas las clases de la capa de c1_Presentacion deben de interactuar con la capa c5_transversal.excepcion que contiene todas las excepciones que se van a implementar y con la librería log4j.jar y por ultimo log4j.properties (captura los errores) y lo registra en un archivo loggersRegistro.log.

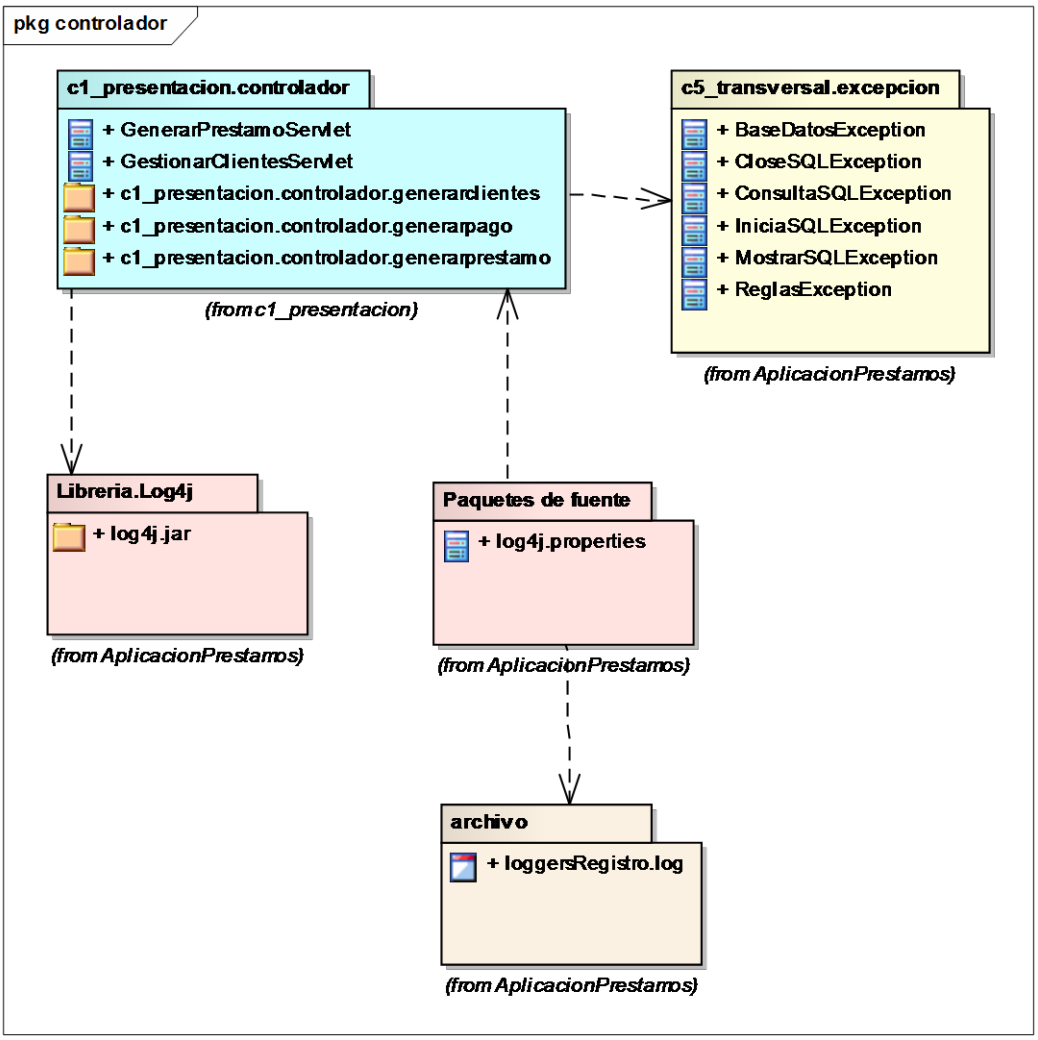


Figura 7: Diagrama de Clases parcial del “controlador”

c. Modelo: la figura 7: muestra las subcapas que contiene, la interacción con la capa de c5_tranversal.excepcion, con la librería log4.jar, con log4j.properties (captura los errores) y lo registra en un archivo loggersRegistro.log.

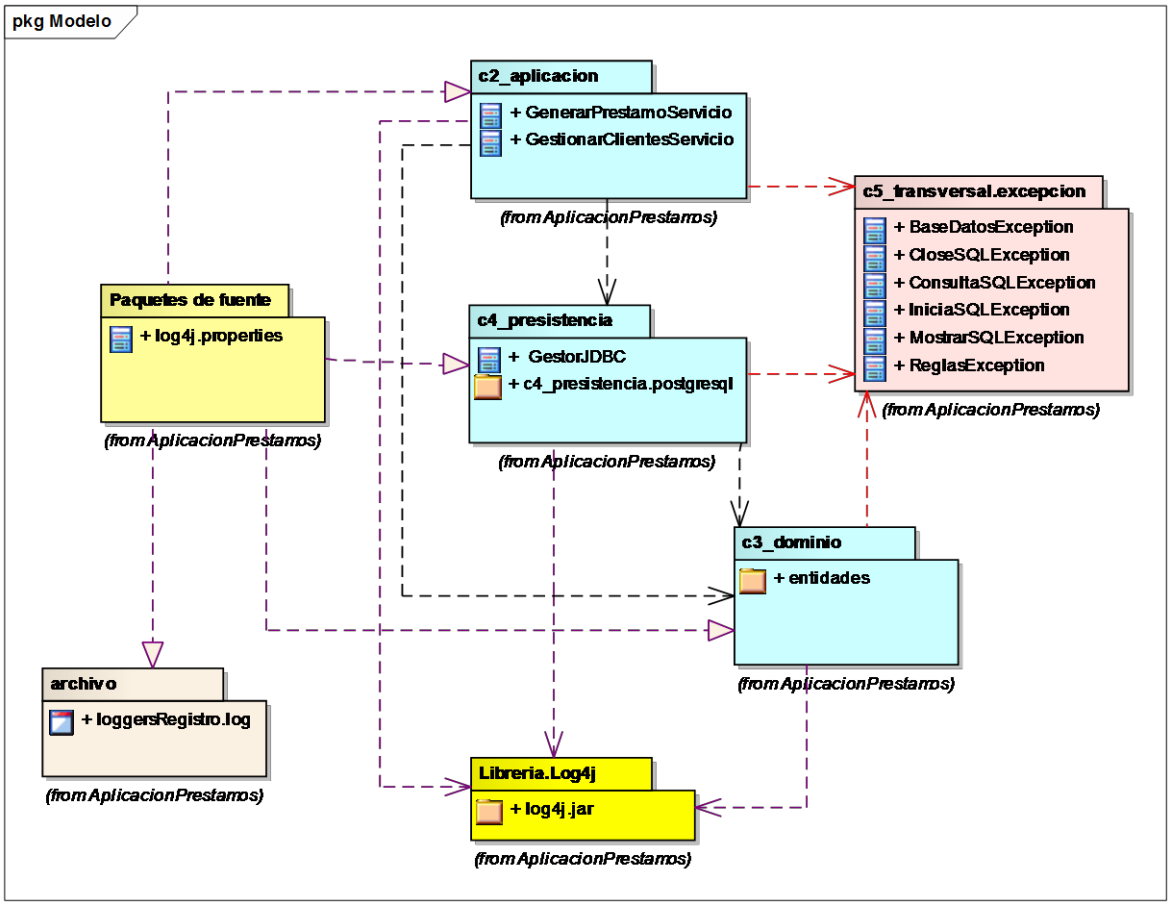


Figura 8: Diagrama de paquetes del “Modelo”

a. Subcapa Entidad: la figura 8 muestra las clases de la subcapa. Esta subcapa representa la parte más significativa de la arquitectura por estar centrado en resolver o hacer cumplir las reglas de negocio. Las clases de diseño que aparecen en esta subcapa, representan entidades del dominio de problema que se está analizando, surgen de los conceptos del negocio y que son representados inicialmente en el Modelo de Dominio en una fase de análisis anterior.

Se debe colocar en las clases del diagrama sólo los atributos y operaciones más significativas, en especial las operaciones que resuelven reglas de negocio. Estas clases se nombran como los conceptos mismos del negocio.

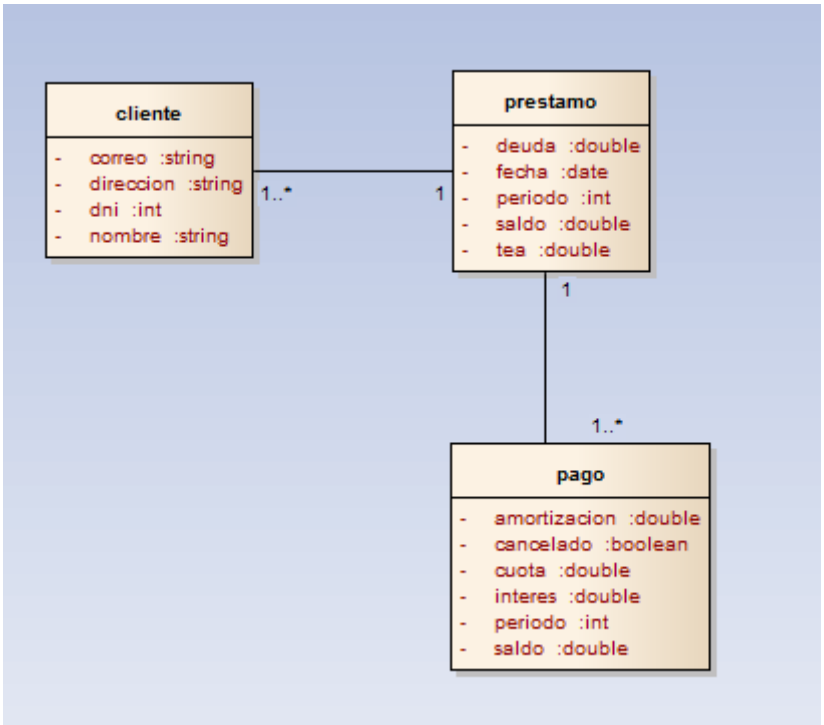


Figura 9: Diagrama de Clases parcial de la subcapa "entidades"

b) sub Capa Persistencia o conexión a base de datos: la figura 9 muestra la subcapa de la capa y a la vez la clase abstracta "GestorJDBC" encargada de gestionar el acceso a las bases de datos usando tecnología JDBC. Y la figura 9 muestra las clases de la subcapa "daoPostgreSql",

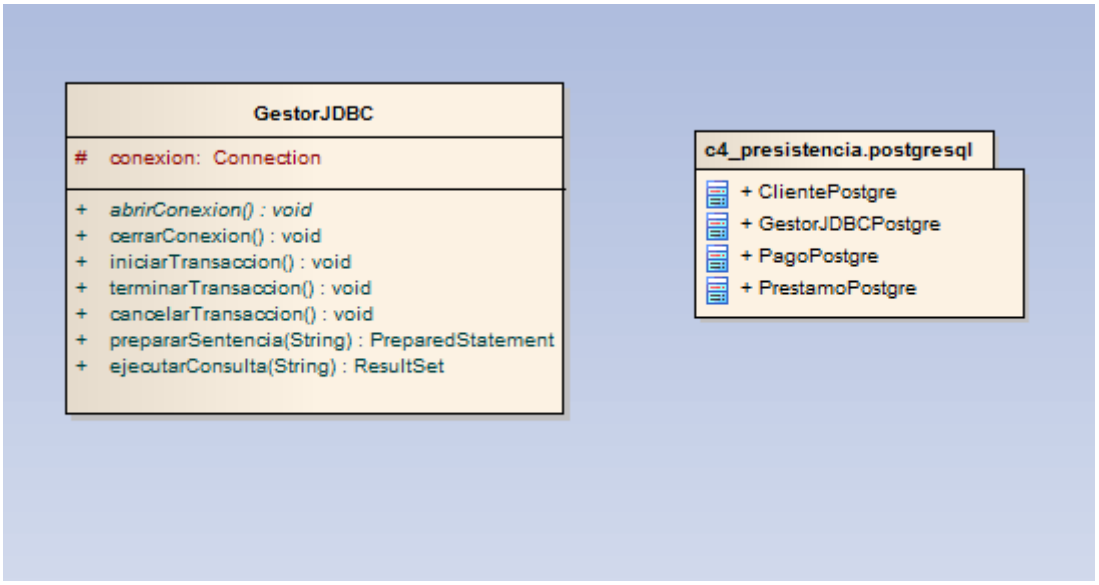


Figura 10: Diagrama de clases y paquete del "Modelo"

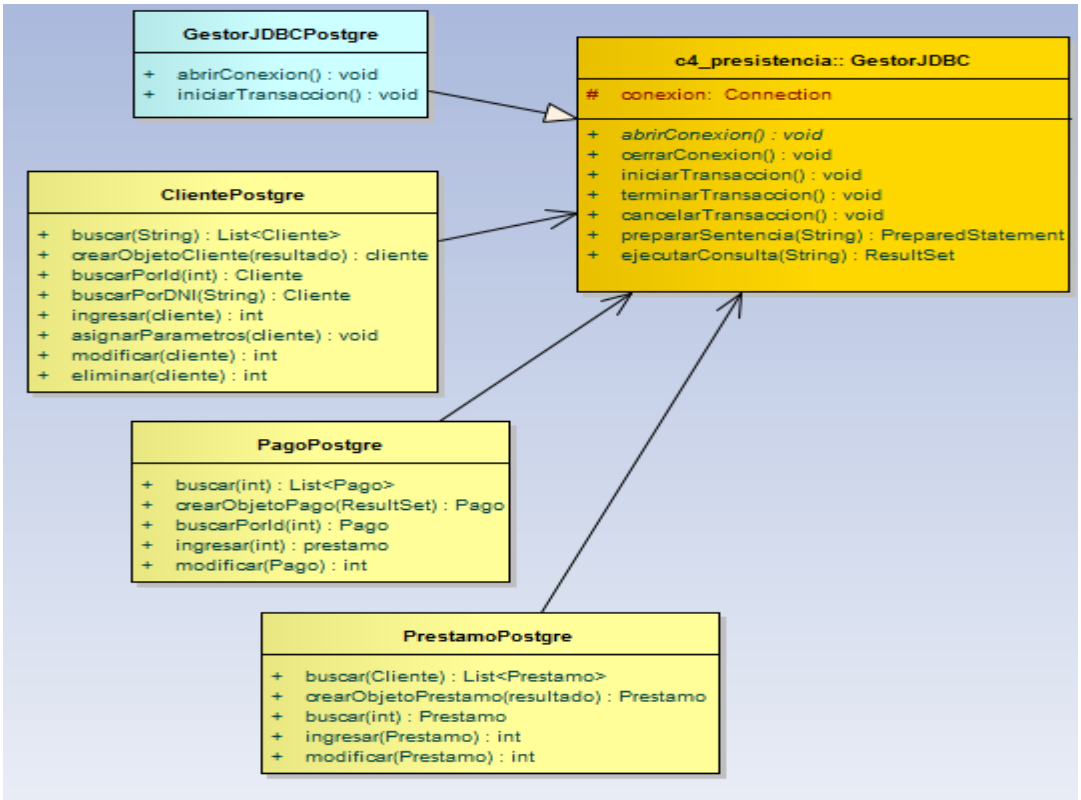


Figura 11: Diagrama de clases parcial de la subcapa “daoPostgreSql”

Vista de Implementación

Diagrama de Componentes

La figura 12 muestra el diagrama de componentes de la aplicación préstamos. Siguiendo el patrón de MVC, la aplicación será implementada en 2 componentes. Los componentes representan un archivo compilado con extensión .jar debido a que está desarrollado en lenguaje Java, además cada componente incluye todas las clases compiladas .class que serán generadas usando la herramienta NetBeans 8.0.1.

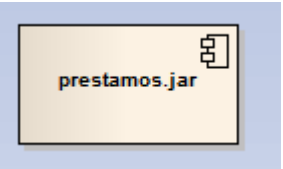


Figura 12: Diagrama de Componentes

Implementación

Capa de excepciones implementada en el software y el framework log4j.

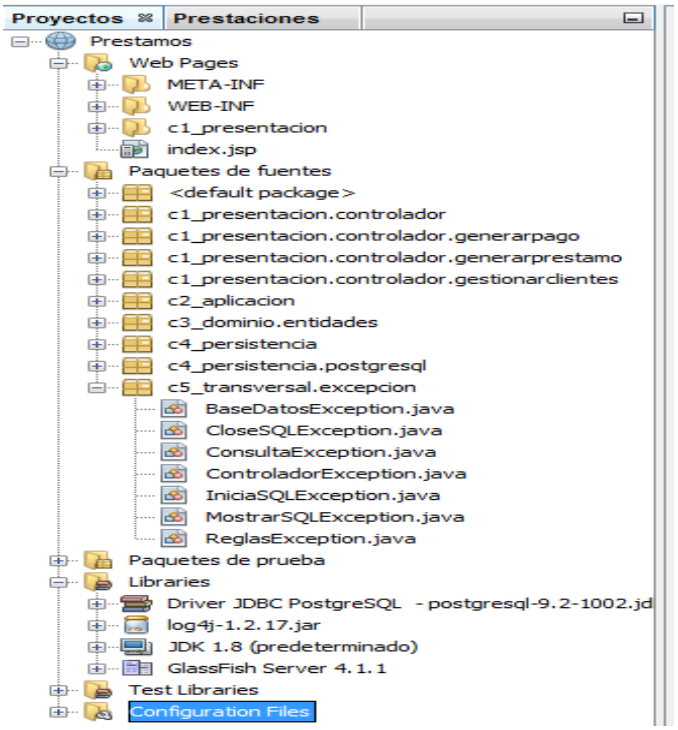


Figura 13: capa transversal con las excepciones

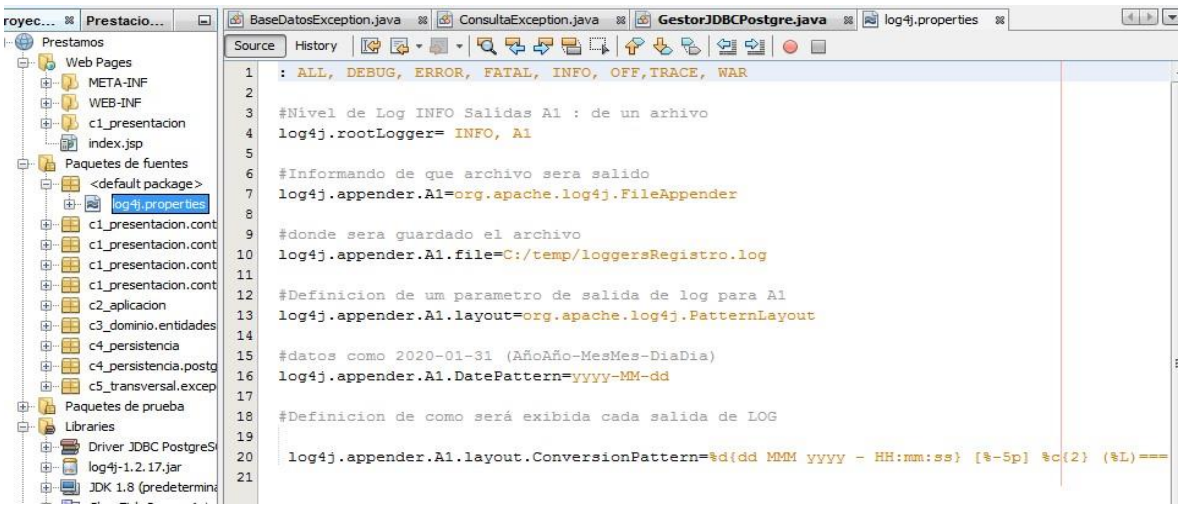
Excepción creada para controlar los errores de conexión con base de datos.

```java
package c5_transversal.excepcion;

public class BaseDatosException extends Exception {

    /**
     * Creates a new instance of <code>ExcepcionBaseDatos</code> without detail
     * message.
     */
    public BaseDatosException() {
    }
/**
     * Constructs an instance of <code>ExcepcionGeneral</code> with the
     * specified detail message.
     *
     * @param msg the detail message.

       */
    public BaseDatosException(String msg) {
        super(msg + " Consultar con el Administrador");

    }
}
```

Capturar la excepción creada BaseDatosExcepction y la utilización del framewrok log4j. Para mostrar lo errores y guardar los errores.

```java
package c4_persistencia.postgresql;
import c4_persistencia.GestorJDBC;
import c5_transversal.excepcion.BaseDatosException;
import java.sql.DriverManager;
import java.sql.SQLException;
import org.apache.log4j.Logger;

public class GestorJDBCPostgre extends GestorJDBC {
 private static Logger log = Logger.getLogger(GestorJDBCPostgre.class);
    @Override
    public void abrirConexion() throws BaseDatosException{
       log.info("Iniciando conexion");
        try {
             Class.forName("org.postgresql.Driver");
        String url = "jdbc:postgresql://localhost:5432/BaseFinancieraSoft";
        conexion = DriverManager.getConnection(url, "postgres", "1234566");
        } catch (ClassNotFoundException | SQLException e) {
           log.error("no se puede conectar" + e.getMessage());
            throw new BaseDatosException("Error del Sistema:");
        }


    }
```

Implementacion log4j.properties que me permite capturar todo que esta sucediendo en el software por cada proceso iniciado en las diferentes capas.

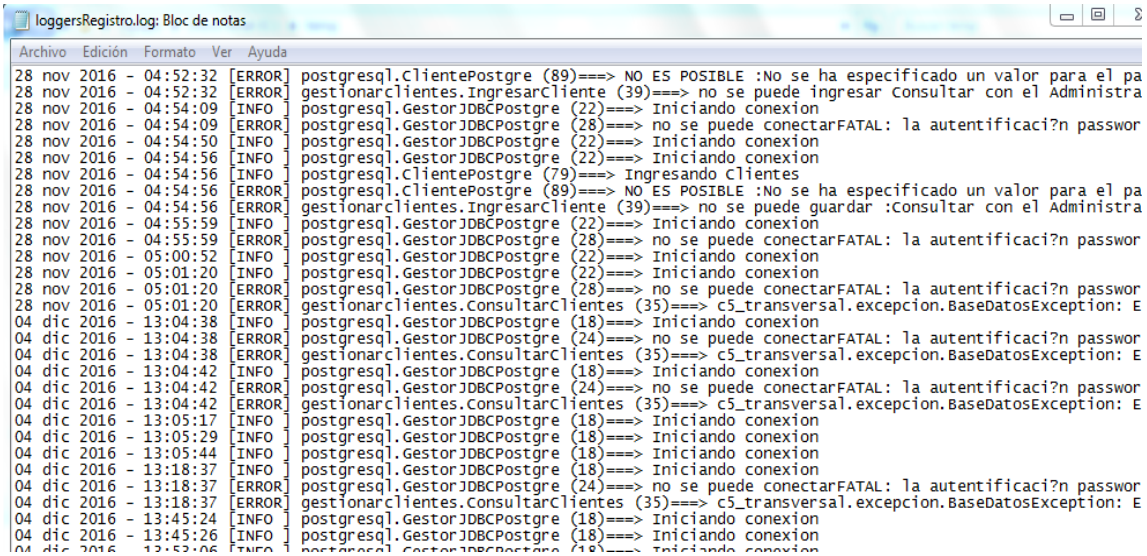


Implementación de la excepción ConsultaException para las diversas consultas y que heredan de SQLEception.

```
package c5_transversal.excepcion;

import java.sql.SQLException;

public class ConsultaException extends SQLException {



    public ConsultaException () {
    }

    public ConsultaException (String msg) {
        super(msg + "Consultar con el Administrador");


    }

}
```

Registro de log para el administrador y excepciones que muestra al usuario

El archivo que a continuación se muestra su contenido, se creó debido a que se utiliza un framework llamado log4j.jar, se utiliza en la codificación del código, en donde se quiere capturar los errores tanto de las excepciones personalizadas como los errores que muestre el sistema cuando se está utilizando. El cual registra la fecha, mes, año, hora, tipo de error, en que capa del desarrollo del software está ocurriendo el error, el número de línea, registra el mensaje como podemos ver tanto la excepción para el usuario como para el administrador del sistema.


loggersRegistro.log: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
28 nov 2016 - 04:52:32 [ERROR] postgresql.ClientePostgre (89)===> NO ES POSIBLE :No se ha especificado un valor para el pa
28 nov 2016 - 04:52:32 [ERROR] gestionarclientes.IngresarCliente (39)===> no se puede ingresar Consultar con el Administra
28 nov 2016 - 04:54:09 [INFO ] postgresql.GestorJDBCPostgre (22)===> Iniciando conexion
28 nov 2016 - 04:54:09 [ERROR] postgresql.GestorJDBCPostgre (28)===> no se puede conectarFATAL: la autentificaci?n passwor
28 nov 2016 - 04:54:50 [INFO ] postgresql.GestorJDBCPostgre (22)===> Iniciando conexion
28 nov 2016 - 04:54:56 [INFO ] postgresql.GestorJDBCPostgre (22)===> Iniciando conexion
28 nov 2016 - 04:54:56 [INFO ] postgresql.ClientePostgre (79)===> Ingresando Clientes
28 nov 2016 - 04:54:56 [ERROR] postgresql.ClientePostgre (89)===> NO ES POSIBLE :No se ha especificado un valor para el pa
28 nov 2016 - 04:54:56 [ERROR] gestionarclientes.IngresarCliente (39)===> no se puede guardar :Consultar con el Administra
28 nov 2016 - 04:55:59 [INFO ] postgresql.GestorJDBCPostgre (22)===> Iniciando conexion
28 nov 2016 - 04:55:59 [ERROR] postgresql.GestorJDBCPostgre (28)===> no se puede conectarFATAL: la autentificaci?n passwor
28 nov 2016 - 05:00:52 [INFO ] postgresql.GestorJDBCPostgre (22)===> Iniciando conexion
28 nov 2016 - 05:01:20 [INFO ] postgresql.GestorJDBCPostgre (22)===> Iniciando conexion
28 nov 2016 - 05:01:20 [ERROR] postgresql.GestorJDBCPostgre (28)===> no se puede conectarFATAL: la autentificaci?n passwor
28 nov 2016 - 05:01:20 [ERROR] gestionarclientes.ConsultarClientes (35)===> c5_transversal.excepcion.BaseDatosException: E
04 dic 2016 - 13:04:38 [INFO ] postgresql.GestorJDBCPostgre (18)===> Iniciando conexion
04 dic 2016 - 13:04:38 [ERROR] postgresql.GestorJDBCPostgre (24)===> no se puede conectarFATAL: la autentificaci?n passwor
04 dic 2016 - 13:04:38 [ERROR] gestionarclientes.ConsultarClientes (35)===> c5_transversal.excepcion.BaseDatosException: E
04 dic 2016 - 13:04:42 [INFO ] postgresql.GestorJDBCPostgre (18)===> Iniciando conexion
04 dic 2016 - 13:04:42 [ERROR] postgresql.GestorJDBCPostgre (24)===> no se puede conectarFATAL: la autentificaci?n passwor
04 dic 2016 - 13:04:42 [ERROR] gestionarclientes.ConsultarClientes (35)===> c5_transversal.excepcion.BaseDatosException: E
04 dic 2016 - 13:05:17 [INFO ] postgresql.GestorJDBCPostgre (18)===> Iniciando conexion
04 dic 2016 - 13:05:29 [INFO ] postgresql.GestorJDBCPostgre (18)===> Iniciando conexion
04 dic 2016 - 13:05:44 [INFO ] postgresql.GestorJDBCPostgre (18)===> Iniciando conexion
04 dic 2016 - 13:18:37 [INFO ] postgresql.GestorJDBCPostgre (18)===> Iniciando conexion
04 dic 2016 - 13:18:37 [ERROR] postgresql.GestorJDBCPostgre (24)===> no se puede conectarFATAL: la autentificaci?n passwor
04 dic 2016 - 13:18:37 [ERROR] gestionarclientes.ConsultarClientes (35)===> c5_transversal.excepcion.BaseDatosException: E
04 dic 2016 - 13:45:24 [INFO ] postgresql.GestorJDBCPostgre (18)===> Iniciando conexion
04 dic 2016 - 13:45:26 [INFO ] postgresql.GestorJDBCPostgre (18)===> Iniciando conexion
```

Vista de Despliegue

Diagrama de Despliegue

Esta vista permite mostrar y describir los nodos en donde serán distribuidos físicamente los distintos componentes del Sistema. La arquitectura física del Sistema se basará en una distribución de dos niveles, siguiendo la típica arquitectura Cliente / Servidor. En el primer nivel, representado por el nodo *Equipo Cliente* (cualquier equipo cliente), se debe configurar o instalar todos los componentes de la aplicación préstamos. En el segundo nivel, representado por el nodo *Servidor de base de datos,* se debe configurar o instalar todos los componentes del Motor de base de datos y la base de datos de préstamos. La aplicación préstamos será ejecutada en el nodo

Cliente y hará peticiones al nodo Servidor para conectarse a la base de datos y procesar sentencias SQL.

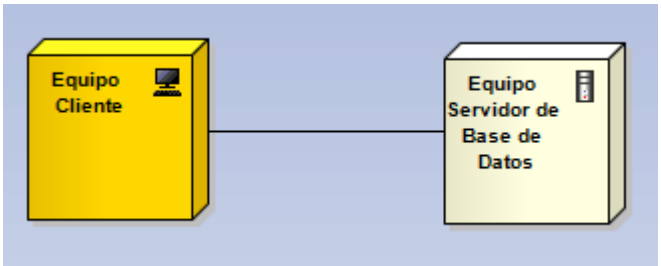


Figura 13: Diagrama de Despliegue

Nodos

Nodo 1: Equipo Cliente

Tipos de componentes	Componentes
Sistema operativo	Windows 7
Frameworks	JDK 1.7 o superior Log4j.jar
Componentes de la aplicación	prestamos.jar

Nodo 2: Equipo Servidor de Base de Datos:

Tipos de componentes	Componentes
Sistema operativo	Windows 7
Motor de base de datos	PostgresSQL
Base de datos	PresamosSoft

3.4. Fase de Transición

3.4.1. Plan de Pruebas

El principal propósito es encontrar errores y defectos que puedan existir en el uso del sistema con el fin de corregirlos. Verificar que los ingresos de datos funcionen y no puedan ingresar datos que no estén permitidos. Se quiere comprobar que el sistema cumple con los requerimientos

establecidos al inicio de la investigación y tenga un rendimiento adecuado en el ambiente donde se encuentra instalado. Otro aspecto importante es registrar todas las incidencias de errores que puedan ocurrir.

Este plan de pruebas es para el **sistema de préstamos** trata de cumplir lo siguiente:

Se utiliza el tipo de prueba de caja negra y un framework log4j para la registración de todas las incidencias de las excepciones implementadas que pueda ocurrir en su ejecución del software:

Prueba Funcional:

Cuadro 1: Prueba Funcional

Objetivo:	Asegurar la funcionalidad requerida de entrada de datos, su procesamiento y registración de errores que por las excepciones implementadas.
Técnica de Caja Negra:	Partición de equivalencias y utilización de un framework log4j.jar
Consideraciones especiales:	Ejecutar cada caso de uso, utilizando datos válidos y no válidos para verificar lo siguiente: ▪ Se obtiene los resultados esperados cuando se utilizan datos válidos ▪ Cuando se utilizan datos no válidos se muestran los mensajes de error o advertencia adecuados. Utilización del framework para: ▪ Registro de errores cuando ocurre en el proceso de ejecución. ▪ Registro de que está sucediendo en cada caso de uso implementado (info).
Criterios de finalización	Se han ejecutado todas las pruebas planeadas (todos los casos de usos priorizados han sido probados). Se ha utilizado el framework para el registro de log de todas las incidencias que ha pasado con

	los casos de pruebas funcionales ejecutadas.

Resultados de las pruebas

Aquí se incluyen las salidas esperadas para cada prueba, las salidas reales y, para cualquier prueba que haya fallado. Se muestran los detalles de cada prueba y sus resultados esperados.

Casos de prueba funcional

Responsable de prueba: investigador

Cuadro 2: Nombre del caso de uso: Generar préstamo

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	**CLASE VALIDA**	**CLASE NO VALIDA**
El DNI es una cadena de 8 números	1. cualquier cadena de 8 números	2: cadena menos de 8 números
Monto entre 10 y 50000	3. 10<= Monto 50000	4: Monto<10 5:Monto>50000
Tasa efectiva esta entre 1% y 3%	6: 1%<=TE<=3%	7:TEA<1% 8:TE>3%
Total periodos es una cadena de 3 caracteres como máximo	7: Cualquier cadena de 3 caracteres como máximo.	8: cualquier cadena más de 3 caracteres

CASOS DE PRUEBA						
NRO	**CLASES DE EQUIVALENCIA**	**DNI de cliente**	**Deuda préstamo**	**Tasa Efectiva Anual**	**Total periodos**	**RESULTADO ESPERADO**
CP-01	1, 3,6,9	44520103	5000	3	12	El préstamo ha sido registrado
CP-02	1,4,6,9	44520103	5	4	8	El monto es inferior.
CP-03	1,3,5,6,9	44107120	20000	16	10	El préstamo ha sido registrado.
CP-04	1,3,7,9	44107120	5000	0	6	Tasa efectiva anual no está permitido
CP-05	1,3,6,9	44107122				El cliente no existe
CP-06	1,3,6,9,10	5000	1	4	60	El préstamo no se ha registrado.

Nombre del caso de uso: Gestionar Clientes

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	**CLASE VALIDA**	**CLASE NO VALIDA**
El nombre acepta solo	1: cualquier cadena como	2: cualquier cadena más de

cadena de 40 caracteres como máximo y mínimo 4	máximo 40 caracteres y mínimo 4	40 caracteres 3: cualquier cadena menos de 4 caracteres
El DNI es una cadena de 8 caracteres	4. cualquier cadena de ocho números	5: cadena menos de 8 números
Dirección acepta una cadena de 50 caracteres	6: cualquier cadena como máximo de 50 caracteres	7: cualquier cadena más de 50 caracteres
Correo acepta una cadena de 40 caracteres.	8. cualquier cadena como máximo de 40 caracteres	9. cualquier cadena más de 40 caracteres

CASOS DE PRUEBA						
NRO	**CLASES DE EQUIVALENCIA**	**Nombre**	**DNI**	**Dirección**	**Correo**	**RESULTADO ESPERADO**
CP-01	1,4,6,8	SANCHEZ HERNANDEZ, KARINA	44520103	SAN ISIDRO 530	SANCHEZ-20@HOTMAIL.COM	Se guardó un nuevo Cliente
CP-02	2,4,6,8	CAR	78945113	AV. SANCHEZ CARRION 7852	ALGA_581@GMAIL.COM	Ingresar Nombre
CP-03	3,4,6,9	MALDONADO TINCO, SANDRA, MONICA	64501234	III ETAPA MANUEL AREVALO 984	SANDRA_785@HOTMAIL.COM	Nombre es superior a lo permitido
CP-04	1,5,6,8	MALDONADO TINCO, SANDRA, MONICA	6450123	III ETAPA MANUEL AREVALO 984	SANDRA_785@HOTMAIL.COM	DNI es inferior a lo permitido verificar.
CP-05	1,4,7,8	BEDREGAL CANALES, LUZ	30604203	BUENOS AIRES 452 Alcaldía	MANUEL_54_20@HOTMAIL.COM, MANUEL_54_	El correo es superior a lo permitido verificar.

		MARINA		de Placido Rosas	20@HOTMAIL .COM,	
CP-06	3,5,8,10	BEDREGAL CANALES, LUZ MARINA	30604203	BUENOS AIRES 452	MANUEL_54_20@hotmailL.COM	Verificar la dirección o consultar con el administrador

Resultados de la evaluación de métricas según la fiabilidad del software.

1. RESULTADO DETALLADO DE LAS PRUEBAS

CU-01 Realizar Genarar Prestamo

CP	Error	0	1	2	3	4	Valor	ITERACIÓN 1 PROM	D.E.	Nivel
CP-01	si		x				1	1.67	1.51	Marginal
CP-02	si	x					0			
CP-03	Si				x		3			
CP-04	Si				x		3			
CP-05	si	x					0			
CP-06	Si				x		3			
	6	2	1	0	3	0				

CU-02 Gestionar clientes

CP	Error	0	1	2	3	4	Valoración	ITERACIÓN 1 PROM	D.E.	Nivel
CP-01	Si	x					0	1.33	1.03	Marginal
CP-02	Si	x					0			
CP-03	Si			X			2			
CP-04	si			x			2			
CP-05	Si			x			2			
CP-06	Si			x			2			
	6	2	0	4	0	0				

Tabla 2: Detalle de resultados primera Iteración

2. RESULTADO ACUMULADO DE LAS PRUEBAS

Cobertura total de la prueba

	Iteración 1
Casos de Prueba diseñados	12
Casos de prueba ejecutados con error y sin ejecutar	12

Cobertura de pruebas ejecutadas

	Iteración 1
Casos de prueba ejecutados	12
Casos de prueba ejecutados con error	8

Total de errores por nivel de severidad

Nivel de severidad	Iteración 1
Ninguno	0
Menor	1
Marginal	4
Crítico	3
Catastrófico (sin ejecutar)	0

Cobertura de la prueba por caso de uso

Caso de Uso	Casos de Prueba diseñados
CU-01	6
CU-02	6
CU-03	
TOTAL	**12**

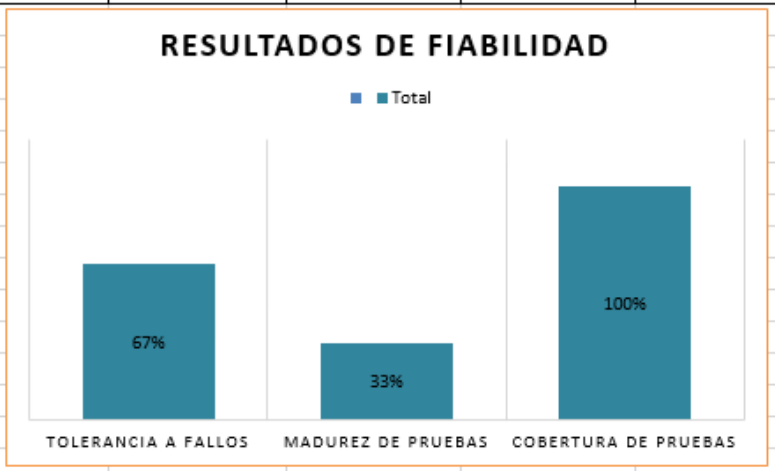
Promedio del nivel de severidad de error por Caso de Uso

	Iteración 1
Caso de Uso	Nivel promedio de severidad de error
CU-01	1.67
CU-02	1.33
CU-03	0.00

Resumen de pruebas

Tabla3: Resultados de la primera medición.

Métrica	ITERACIÓN 1 Medición Dato A	Medición Dato B	Fórmula: X=A/B	Nivel	
Tolerancia a fallos	8	12	0.67	Marginal	67%
Madurez de pruebas	4	12	0.33	Insatisfactorio	33%
cobertura de pruebas	12	12	1.00	Satisfactorio	100%



Resumen de pruebas en la segunda Interacción

1. RESULTADO DETALLADO DE LAS PRUEBAS

CU-01 Realizar Genarar Prestamo								ITERACIÓN 2		
CP	Error	0	1	2	3	4	Valor	PROM	D.E.	Nivel
CP-01	si		x				1	0.17	0.41	Menor
CP-02	si	x					0			
CP-03	Si	x					0			
CP-04	Si	x					0			
CP-05	si	x					0			
CP-06	Si	x					0			
	6	4	1	0	0	0				

CU-02 Gestionar clientes								ITERACIÓN 2		
CP	Error	0	1	2	3	4	Valoración	PROM	D.E.	Nivel
CP-01	Si	x					0	0.17	0.41	Menor
CP-02	Si	x					0			
CP-03	Si	x					0			
CP-04	si	X					0			
CP-05	Si	X					0			
CP-05	Si		x				1			
	6	5	0	0	0	0				

Cobertura total de la prueba

	Iteración 1
Casos de Prueba diseñados	12
Casos de prueba ejecutados con error y sin ejecutar	12

Cobertura de pruebas ejecutadas

	Iteración 1
Casos de prueba ejecutados	12
Casos de prueba ejecutados con error	2

Total de errores por nivel de severidad

Nivel de severidad	Iteración 1
Ninguno	10
Menor	2
Marginal	0
Crítico	0
Catastrófico (sin ejecutar)	0

Cobertura de la prueba por caso de uso

Caso de Uso	Casos de Prueba diseñados
CU-01	6
CU-02	6
CU-03	
TOTAL	**12**

Promedio del nivel de severidad de error por Caso de Uso

	Iteración 1
Caso de Uso	Nivel promedio de severidad de error
CU-01	0.41
CU-02	0.41

	ITERACIÓN 2				
Métrica	Medición Dato A	Medición Dato B	Fórmula: X=A/B	Nivel	
Tolerancia a fallos	11	12	0.92	Satisfactorio	92%
Madurez de pruebas	10	12	0.83	Marginal	83%
cobertura de pruebas	12	12	1.00	Satisfactorio	100%

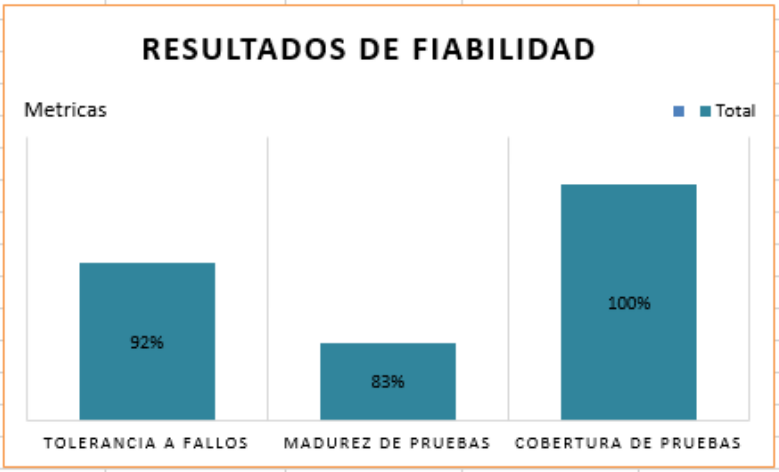


Tabla 4: Resultados de la segunda medición al software

IV. DISCUSIÓN

La siguiente investigación se analizó un software mediante introspección de código y casos de pruebas funcionales, permitiendo conocer cuál era su actualidad en el manejo de captura de errores y excepciones. Pudiendo encontrar que las excepciones y captura de errores no era el adecuado para tener un software seguro y fiable. Se analizó su situación del software y luego se implementó la metodología de RUP. El cual siguiendo su estructura se obtuvo los siguientes resultados.

En la primera fase 1 inicio, se analizó los requerimientos no funcionales permitiendo encontrar que debería ser un software confiable, escalable, usable y robusto, las capturas de excepciones deben de ser reutilizables en las diferentes capas del sistema implementado y debe de ser capturados en un registro para el control del administrador. Se diseñó el plan del Software para saber cómo se estaba estructurando el desarrollo de la investigación, se elaboró los requerimientos no funcionales para saber algunos requisitos a tener encuentra en la implementación de la excepciones como software tolerante a fallos, madurez,

que plataforma se estaba utilizando, que sistema entre otros más , en la fase 2 se diseñó se diseñó una nueva arquitectura con las excepciones que se ha implementado, el cual esta especificada según la arquitectura modelo, vista, controlador, en ella contiene la estructuración de cada una de las capas que interactúa tanto con las excepciones y con el framawok log4j para la captura de excepciones y registro de loggers

Siguiendo con el desarrollo de la metodología en la fase 3 llamada desarrollo se obtuvo según los resultados 3.4.3. Documento de arquitectura de software, el cual contenía todos los diseños a tener en cuenta en la implementación de las excepciones en el software.

Terminado con el desarrollo en la fase 4 llamado fase de transición se consideró las pruebas de caja negra el cual se determinó las clases válidas y no válidas para ingresar al sistema tal como indica en el cuadro N° 2, denominada partición de equivalencias, con el cual nos permite evaluar la funcionalidad del sistema y ver su fiabilidad en cuanto a la detención de errores.

Así como se comprobó la investigación publicada por la Revista Cubana de Ciencias Informáticas: describiendo en su artículo el tema *(Aplicando métricas de calidad a proyectos y procesos durante las pruebas exploratorias, 2013)* se enfoca en cuatro razones que son evaluar, caracterizar, predecir y mejorar y como utilizar *"métricas internas y externas para evaluar el software". Utilizando la norma ISO/IEC 2196 como referencia. El cual se tomó como referencia y poder evaluar la fiabilidad del software ya estando las excepciones implementadas el cual se midió el grado de la tolerancia a fallos obteniendo en la primera interacción un porcentaje de 67% y en la segunda interacción un 97% lo que representa un aumento de un 30% en la tolerancia a fallos.*

Con respecto la madurez de pruebas en la primera interacción se obtuvo un 33% y en la segunda interacción se obtuvo un 83% lo que representa un aumentado un 50% de madurez del software.

Con respecto a la cobertura de pruebas en la primera interacción se obtuvo un 100% y en la segunda interacción de un 100% lo que representa que en la

primera y segunda interacción las coberturas de pruebas se utilizaron todas las diseñadas.

Los resultados confirman la hipótesis planteada, existe una significativa diferencia entre la situación anterior a la implementación de la capa de excepciones en el sistema en comparación con la situación posterior a la implementación de la capa de excepciones en el sistema. La perspectiva de los resultados en tolerancia a fallos y madures de pruebas es otra y mientras tanto en cobertura de pruebas se mantiene los resultados.

V. CONCLUSIÓN

Según los resultados y discusiones que se pudieron observar, se obtuvo lo siguiente:

1. Se aumentó el porcentaje de Tolerancia de fallos de un 67% a un 97% con la implementación de la capa de excepciones en el sistema.
2. Se aumentó el grado de madurez de las pruebas de un 33% a un 83% con la implementación de la capa de excepciones en el sistema.
3. Se mantuvo el porcentaje en cuanto al Índice de cobertura de pruebas de un 100% con la implementación de la capa de excepciones en el sistema.

VI. RECOMENDACIONES

1. Seguir investigando sobre las diversas formas de utilización del framework log4j.
2. La utilización de la capa de excepciones para la visualización de excepciones personalizadas y del framework log4j para el registro de loggers en otros proyectos de desarrollo de software.
3. Seguir utilizando las buenas prácticas de desarrollo como estar estructurado en capas para incrementar la escalabilidad del sistema.

XII. REFERENCIAS

1) **Marco, David. 2012.** Tratamiento de excepciones en Java. [En línea] 06 de 06 de 2012. [Citado el: 09 de 05 de 2016.] http://www.davidmarco.es/articulo/tratamiento-de-excepciones-en-java.

2) **Pressman, Roger S. 2010.** *Ingeniería del software, Séptima Edición.* México : Edit: McGraw-Hill, 2010. ISBN: 978-607-15-0314-5.

3) **Sommervalle, Iam. 2011.** *Ingeniería de Software, 9na Edición.* Mexico : Pearson Education, 2011.

4) **Alan Burns, Andy Wellings. 2003.** *Sistemas de tiempo real y lenguajes de programación.* España : ADDISON-WESLEY, 2003. 8478290583, 9788478290581.

5) *Aplicando métricas de calidad a proyectos y procesos durante las pruebas exploratorias.* **Informáticas, Revista Cubana de Ciencias. 2013.** N°. 2, La Habana : s.n., 2013, Vol. 7. ISSN 2227-1899.

6) **artificial, El departamento de ciencia de la comunicación e inteligencia. 2012.** Curso de Especialista en Aplicaciones y Servicios Web con Java Interprise. [En línea] 2012. [Citado el: 03 de 05 de 2016.] http://www.jtech.ua.es/j2ee/publico/lja-2012-13/sesion03-apuntes.html.

7) *Asegurar que el software crítico se construye fiable y seguro.* **Rodríguez Dapena, Patricia. 2009.** núm. 2, España : Edit.Asociación de Técnicos de Informática, 2009, Vol. vol. 5. E-ISSN: 1885-4486.

8) **Asenjo, Jorge Sánchez. 2010.** Fundamentos de Programación Unidad 8 Control de Excepciones. [En línea] 2010. [Citado el: 07 de 05 de 2016.] http://www.jorgesanchez.net/programacion/apuntes2009/fpr0809.pdf.

9) **—. 2004.** NetBeans. [En línea] 2004. [Citado el: 2016 de 05 de 08.] http://www.jorgesanchez.net/programacion/manuales/NetBeans.pdf.

10) **CRISTANCHO, Mgr. JOSÉ ALBERTO.** EVALUACIÓN DE LA CALIDAD DEL SOFTWARE. [En línea] [Citado el: 25 de 06 de 2016.] http://ecotropicos.saber.ula.ve/db/ssaber/Edocs/pubelectronicas/evaluacioninvestigacion/vol2num1/articulo3.pdf.

11) **ELIZONDO, PERLA INÉS VELASCO. 2001.** *PRUEBA DE COMPONENTES DE SOFTWARE BASADAS EN EL MODELO DE JAVABEANS.* MEXICO : TLAXCALA, 2001.

12) **Funes, Luis Enrique. 2009.** Conociendo a NetBeans Platform: Introducción. [En línea] 06 de 11 de 2009. [Citado el: 2016 de 05 de 08.] http://wiki.netbeans.org/ConociendoNetbeansPlatformIntroduccion.

13) **Hernández, Fernándes, Baptista. 2006.** *Metodoloía de la Inveastigación.* Mexico : McGRAW-HILL Interamericana S.A, 2006.

14) **Jorge Sánchez Asenjo, Sun. 2004.** *Java2, incluye Swing, Threads,programación en red,JavaBeans, JDBC yJSP / Servlets.* 2004.

15) **Marco, David. 2012.** Tratamiento de excepciones en Java. [En línea] 06 de 06 de 2012. [Citado el: 09 de 05 de 2016.] http://www.davidmarco.es/articulo/tratamiento-de-excepciones-en-java.

16) **—. 2012.** Tratamiento de excepciones en Java. [En línea] 06 de 06 de 2012. [Citado el: 09 de 05 de 2016.] http://www.davidmarco.es/articulo/tratamiento-de-excepciones-en-java.

17) **Pressman, Roger S. 2010.** *Ingeniería del software, Séptima Edición.* México : Edit: McGraw-Hill, 2010. ISBN: 978-607-15-0314-5.

18) **Sastre, Jorge Asiain. 2013.** Ingeniera de Fiabilidad. [En línea] 11 de 03 de 2013. [Citado el: 02 de 05 de 2016.] http://es.slideshare.net/alterevo/articulo-ingeniera-de-fiabilidad.

19) **Sommerville, Ian. 2005.** *Ingeníeria de Software. 7° Edición.* Madrid : Edit;Pearson Educación S.A, 2005. ISBN: 84-7829-074-5.

ANEXOS

- ✓ Desarrollo de metodología

Anexo 1: Plan de Desarrollo de Software

Introducción

Propósito

El propósito de este Plan de Desarrollo de Software es definir el enfoque y las actividades de desarrollo en términos de las fases e iteraciones requeridas para la implementación de un *Sistema de Prestamos*".

Alcance

Este Plan de Desarrollo de Software describe el plan general para ser utilizado por el investigador de desarrollo del software. Los detalles de las iteraciones individuales se describen en los planes de iteración.

Definiciones, Acrónimos, y Abreviaturas

Referencias

- [1] Guía de Casos de Uso a mejorar en el software con la capa de excepciones.
- [2] Cronograma de Actividades del Proyecto.

Resumen

Este Plan de Desarrollo de Software contiene la siguiente información:

- ***Descripción del Proyecto*** - proporciona una descripción del propósito del proyecto, alcance y objetivos. También define los productos de trabajo que el proyecto se prevé obtener.
- ***Organización del Proyecto*** - describe la estructura organizacional para el investigador del proyecto.
- ***Gestión de Procesos*** - explica el costo estimado y el calendario, define las fases más importantes e hitos del proyecto y describe cómo el proyecto será objeto de seguimiento.
- ***Planes Técnicos de Proceso*** - ofrece una visión general del proceso de desarrollo de software, incluyendo métodos, herramientas y técnicas a seguir.
- ***Planes de Soporte de Proceso*** - esto incluye el plan de gestión de la configuración.

Descripción del proyecto

Propósito del Proyecto, Alcance y Objetivos

El propósito de este proyecto de investigación es poner en marcha una nueva arquitectura basada en una capa de control de excepciones que permita mejorar la fiabilidad del software préstamos. El alcance es cumplir con la implementación de todos los requisitos no funcionales relacionados a la fiabilidad del software.

Los objetivos del proyecto son:

- Gestionar y planificar el proyecto en base a un proceso iterativo e incremental.
- Especificar los requisitos no funcionales del sistema del software.

Disciplina	Artefacto	Inicio	Elaboración	Construcción	Transición
Modelado del Negocio	Modelo de Dominio	c	r		
Requisitos	Modelo de Casos de Uso (diagrama y especificación de CU)	c	r		
	Especificaciones Suplementarias	c	r		
Análisis y Diseño	Modelo de Diseño (diagramas de clases)		c	r	
	Documento de Arquitectura de Software		c	r	
	Modelo de Datos (diseño de base de datos)		c	r	
	Modelo de despliegue		c	r	
Implementación	Modelo de Implementación de la capa de excepciones(software)		c	r	r
Prueba	Caso de Prueba		c	r	r

	Plan de Prueba		c	r	
	Resultados de la prueba		c	r	r
	Resumen de evaluación de prueba		c	r	r
Gestión de Proyectos	Plan de desarrollo de software	c	r	r	r

- Tener definido el alcance y las estimaciones de esfuerzo y costo al finalizar la fase de elaboración.
- Implementar los requisitos en la segunda y tercera fase del proyecto, en las fases de elaboración y construcción.
- Aplicar las pruebas a las funcionalidades implementadas para garantizar la calidad del producto en la segunda y tercera fase del proyecto, en las fases de elaboración y construcción.
- Garantizar que la capa de excepciones implementada en el sistema esté con las pruebas de aceptación final, en la última fase del proyecto o fase de transición.

Supuestos y limitaciones

La fiabilidad es crucial para el software de préstamos lo cual permitirá controlar los diversos errores o excepciones que se presenten en la utilización del software.

Productos de Trabajo del Proyecto

Para el proyecto se ha seleccionado un conjunto de artefactos correspondientes a cada disciplina de la metodología RUP. A continuación se muestra el marco de desarrollo que permitirá guiar el uso de los artefactos en cada una de las fases durante el ciclo de vida del proyecto:

(c: se comienza a elaborar el artefacto; r: se refina el artefacto y se genera nueva versión)

Evolución del Plan de Desarrollo de Software

El Plan de Desarrollo de Software se revisará antes del inicio de cada fase subsiguiente o iteración.

Organización del proyecto

Estructura organizacional

No aplica

Roles y responsabilidades

Rol	Responsabilidad
Analista diseñador(investigador)	Investigar cual son los porcentajes de errores y manejos de excepciones más comunes durante los últimos años en el lenguaje de programación de java. Investigar las dificultades de los programadores en trabajar por capas y mantener un software limpio (ordenado o entendible). Es el responsable de como diseñar las excepciones propias, como funcionaria en el software que se está estudiando, capturarlos en tiempo de ejecución del software y guardar un registro de los errores que se ocasionen.
Desarrollador(investigador)	Es responsable de hacer el seguimiento de su propio progreso, Él es también quien implementa las ideas, y como tal, puede tener que discutir las posibilidades de la implementación con el diseñador. También,

	como responsabilidad importante, debe de documentar el código de excepciones a medida de lo que vaya avanzando, teniendo como objetivo explicar a otros desarrolladores aquellas cosas que no resulten evidentes o claras a partir de la lectura del propio código en sí.

.

Gestión de Procesos

Estimaciones del Proyecto

La estimación de tiempo general del proyecto se muestra en el Plan de Fase. Se estima que la fase de inicio debe durar cuatro semanas en una sola iteración, la fase de elaboración debe durar 6 semanas con dos iteraciones de tres semanas cada una, la fase de construcción debe durar 6 semanas con dos iteraciones de tres semanas cada una, y la fase de transición debe durar dos semanas en una sola iteración

Plan de Proyecto

Plan de Fase

Fase	**Nro. De iteraciones**	**Inicia**	**Finaliza**
Inicio	1	Semana 2	Semana 4
Elaboración	2	Semana 5	Semana 8
Construcción	2	Semana 9	Semana 12
Transición	1	Semana 13	Semana 14

Tabla - resumen de línea de tiempo

Fase	Descripción	Hito
Inicio	En esta fase se analizarán los procesos del software y los requisitos fundamentales que definirán el alcance inicial del software a mejorar.	El Plan de Desarrollo de Software con las estimaciones de tiempo y costo, marca el final de la fase.
Elaboración	En esta fase se obtiene la visión refinada del proyecto a realizar, la implementación iterativa del núcleo de la aplicación, la resolución de riesgos altos, nuevos requisitos y se ajustan las estimaciones.	Implementación de los requerimientos no funcionales en el software con respecto a las excepciones en los diferentes casos de usos implementados en el software.
Construcción	Esta abarca la evolución del software hasta convertirse en producto fiable incluyendo requisitos mínimos. Aquí se afinan los detalles menores como los diferentes tipos de casos o los riesgos menores.	Revisión de los modelos presentes en el plan, ayudados por el modelo de arquitectura del software con respecto a la capa de excepciones.
Transición	En esta fase final, el programa debe estar listo para ser probado, instalado y utilizado por el cliente sin ningún problema. Una vez finalizada esta fase, se debe comenzar a pensar en	Entrega de la primera versión del software mejorado, con la documentación de cada caso de uso para que el administrador del sistema entienda fácilmente la

	futuras novedades para la misma. Desde el punto de vista técnico: el proyecto está formado por los flujos de trabajo fundamentales: captura de requerimientos no funcionales, análisis, diseño, implementación y pruebas. Tanto el punto de vista gerencial como el técnico concuerdan en la iteración.	codificación y la estructura que se ha seguido al mejorar el software.

Tabla - Fases del proyecto y los principales hitos

Objetivos de iteración

Fase	Iteración	Descripción	Hitos asociados	Riesgos contemplados
Inicio	I1	Analizar los procesos del software y elaborar el Modelo de Casos de Uso y el Modelo del Dominio. Analizar los principales requisitos y elaborar la	El Plan de Desarrollo de Software.	Aclara el alcance del proyecto y define estimaciones preliminares que ayudan en la decisión de viabilidad del proyecto.

		Visión y el Modelo de Casos de Uso. Planificar el proyecto.		
Elaboración	E1	Elaborar el Modelo de Diseño, el Modelo de Datos, Modelo de Despliegue y Diseñar el Modelo de Arquitectura de Software.	Modelo de Diseño. Modelo de Datos. Documento de Arquitectura del Software. Modelo de Despliegue.	Determina la estructura de los datos que van a ser la base para poder hacer un diseño más consistente, siguiendo la secuencia de los procesos de la empresa.
Elaboración	E2	Analizar el Modelo de Dominio y agregar alguna especificación suplementaria adicional.	Plan de Gestión de la Configuración.	Revisar los requerimientos del usuario, o en su defecto agregar algún pedido que sea importante para los procesos del sistema.
Construcción	C1	Revisar el Modelo de Diseño y complementar	Documento de Arquitectura del Software. Casos de Uso	Diagramar los casos de uso para entender con amplitud

		lo establecido en los Modelos anteriores.	principal a mejorar en el control de errores.	los actores que intervienen en los procesos de la empresa.
Construcción	C2	Revisar y complementar el Plan de Gestión de la Configuración.	Casos de Uso secundarios. Modelo de Implementación Con la mejora de control de excepciones.	Implementar los diagramas de casos de uso secundarios para completar el proceso de diseño de la estructura del software.
Transición	T1	Planificar una última revisión a los artefactos elaborados y entregar la primera versión del software mejorado.	Entregables del proyecto.	Documentar el proyecto para facilitarle la compresión del mismo al administrador del sistema.

Versiones

Se contará con una primera versión del producto mejorado al finalizar la fase de Elaboración, y con la versión final al término de la fase de Construcción la cual será validada e implantada a lo largo de la fase de Transición.

Cronograma del Proyecto

Ver el Cronograma de Actividades del Proyecto [3].

Recursos del Proyecto

Denominación	Total(s/.)
Recursos humanos	520.00
Hardware	1,500.00
Software	439.98
Materiales e Insumos	171.00
Servicios	744.00
Servicios de internet	712.00
Consumo Energía eléctrica	133.28
Total	S/.4,220.26

Revisar proyecto

Plan de Dotación de Personal

No aplica.

Plan de Adquisición de Recursos

No aplica.

Plan de Formación

No aplica.

Presupuesto

La información siguiente sale del Cronograma de Actividades hecha en el proyecto

Planes de Iteración

Ver el Cronograma de Actividades del Proyecto [3].

Monitoreo y Control del Proyecto

Plan de Gestión de Requisitos

Los requisitos para este sistema son capturados en los Casos de Uso, ver la Guía de Casos de Uso [2].

Plan de Control de Programa

No aplica.

Plan de Control de Presupuesto

No aplica.

Plan de Control de Calidad

No aplica.

Plan de Gestión de Riesgos

Los riesgos son problemas potenciales que pueden afectar el proyecto. Estos riesgos deben ser identificados y listados para describir sus acciones de mitigación que permitan reducir la probabilidad de ocurrencia o sus acciones de contingencia para solucionarlos si llegan a ocurrir.

A continuación se listan los 4 principales riesgos, con mayor probabilidad de ocurrencia e impacto:

Riesgo	**Mitigación**	**Contingencia**
Estimaciones erradas en esfuerzo y tiempo de desarrollo para la fase de Construcción. Esto causa mala asignación de recursos y no cumplimiento con el cronograma planificado para terminar el desarrollo.	Al finalizar la fase de Elaboración se deben hacer estimaciones de esfuerzo aplicando la métrica de *Puntos de Casos de Uso* para tener una mayor certeza.	Coordinar con el cliente una ampliación del cronograma justificando técnicamente el retraso. Debe existir también la posibilidad de involucrar a un programador más para apoyar en el desarrollo.
Cambio de requerimientos en función al software. Esto causa una pérdida de tiempo, porque se volvería a realizar de nuevo los	Mantener reuniones periódicas	Realizar un control de los cambios.

requerimientos.		
La pérdida de la documentación/artefactos del trabajo del proyecto. Esto causa un enorme peligro en el avance de la construcción del sistema.	Al finalizar cada fase del proyecto se debe realizar una copia de seguridad para tener un respaldo de la documentación del proyecto.	Se debe coordinar con el grupo de trabajo de generar nuevamente la documentación del código desarrollado.
El abandono del proyecto, por diversos motivos. Esto causa que el proyecto planificado se retrase.	Realizar los documentos del proyecto correctamente especificado.	Crear toda la documentación bien hechos, para así si viene otra persona que se una al proyecto, analice de manera correcta el proyecto y pueda continuar con la construcción del software.

Planes de Soporte de Procesos

Plan de Gestión de Configuración

Referenciar al plan (nombre del archivo).

Plan de Evaluación

No aplica.

Plan de Aseguramiento de la Calidad

De evaluar fiabilidad del software en base a la norma ISO 9126

Anexo2: Resultados

RESULTADOS DE PRUEBAS FUNCIONALES

NÚMERO:	1	NOMBRE DEL SISTEMA TESTEADO:	SISTEMA DE PRESTAMOS

Estado	Descripción
Sin Error	caso de prueba ejecutado con éxito (ningún error)
Error Menor	caso de prueba ejecutado sin éxito (ej: no se muestran mensajes o alertas, falta ingresar o mostrar algún dato no muy importante)
Error Marginal	caso de prueba ejecutado sin éxito (ej: falla la validación de algunos datos de entrada, falta ingresar o mostrar algún dato importante)
Error Crítico	caso de prueba ejecutado sin éxito (ej: falla algún cálculo, no se cumple alguna regla de negocio, no guarda correctamente en la base de datos)
Error Catastrófico	caso de prueba sin poder ejecutar (ej: no está implementada la funcionalidad que se desea probar, al ejecutar la prueba se sale del sistema o se cuelga)

CASO DE USO:	CU1- Generar Préstamo
RESPONSABLE DE LA PRUEBA:	

CASOS DE PRUEBA

CASO DE PRUEBA: CP-01

ESTADO: [] Sin error [X] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: SE JECUTO EL PRESTAMO EN LOS 12 MESES ES CORRECTO

FIGURAS (evidencias del error): EXISTE ERROR DE DIGITOS DECIMALES NO ENTENDIBLE

Reporte de Pagos generados

DNI del Cliente: 44520103 SANCHEZ HERNANDEZ, KARINA

Periodo	Saldo	Amortización	Interes	Cuota
1	952.0636544323778	47.936345567622 18	95.87269113524432	143.8090367028665
2	899.5315224119986	52.532132020379166	91.27690468248734	143.8090367028665
3	841.9629935237538	57.568528888244856	86.24050781462165	143.8090367028665
4	778.8752148462959	63.087778677457 95	80.72125802540856	143.8090367028665
5	709.7390410492853	69.13617379701051	74.672862905856	143.8090367028665
6	633.9745962155614	75.764444833723 89	68.04459186914262	143.8090367028665
7	550.9464101632607	83.028186052300 75	60.78085065056575	143.8090367028665
8	459.9580884740481	90.988321689212 58	52.820715013653924	143.8090367028665
9	360.2464715226114	99.711616951436 71	44.097419751429 8	143.8090367028665
10	250.9752335165918	109.27123800601 76	34.537798696848 91	143.8090367028665
11	131.227867859259 5	119.74736565733429	24.061671045532215	143.8090367028665
12	4.263256414560601E-13	131.22786785925908	12.581168843607442	143.8090367028665

CASO DE PRUEBA: CP-02

ESTADO: [X] Sin error [ ] Error menor[ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: Se ejecutó la prueba con satisfacción.

FIGURAS (evidencias del error):

Generar Préstamo

DNI de Cliente: [ ] Buscar

Nombre del Cliente: [ ]

Datos del Prestamo

Deuda: [ ]

Tasa efectiva anual: [ ]

Total de periodos: [ ]

Guardar Cancelar

Ingresar monto mínimo de 20 O:Consultar con el Administrador

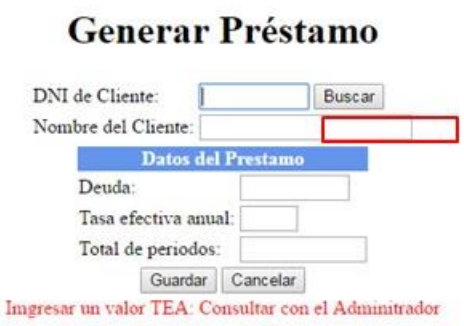
OBSERVACIÓN (opcional):

CASO DE PRUEBA: CP-03

ESTADO: [x] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: Datos mostrados no se guardan y muestra mensajes distintos.

FIGURAS (evidencias del error): el error es controlador por una excepción personalizada en la tasa de interés.



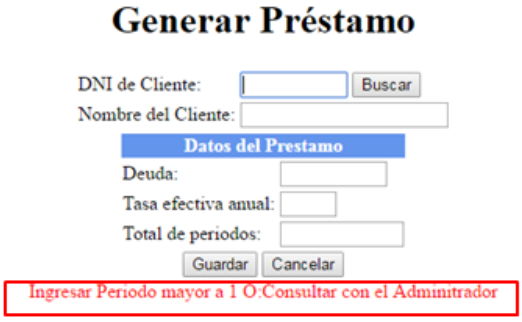
OBSERVACIÓN (opcional):

CASO DE PRUEBA: CP-04

ESTADO: [X] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: la ventana mostrada a continuación no coinciden los datos ni los campos con su CU

FIGURAS (evidencias del error): la excepción permite controlar la cantidad mínima de periodos y máximo de periodos.



OBSERVACIÓN (opcional):

CASO DE PRUEBA: CP-05

ESTADO: [X] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: SE INGRESO LOS DATOS Y MOSTRO EL MENSAGE

CASO DE PRUEBA: CP-05

ESTADO: [X] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: SE INGRESO LOS DATOS Y MOSTRO EL MENSAGE

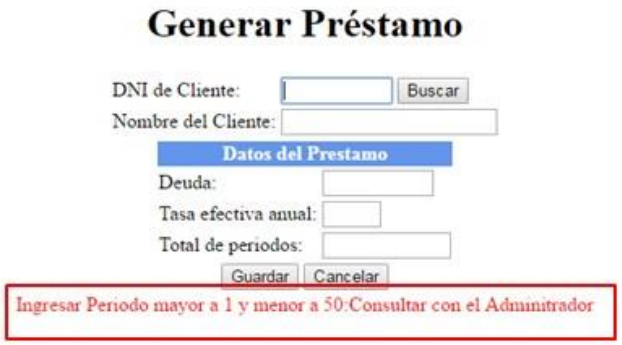
FIGURAS (evidencias del error):

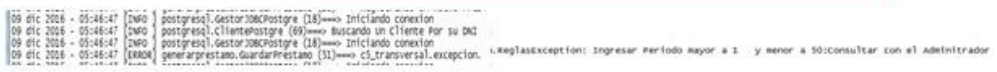
CASO DE PRUEBA: CP-06

ESTADO: [x] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: el control del máximo periodos que puede ser ingresado ha sido controlado.

FIGURAS (evidencias del error):





OBSERVACIÓN (opcional):

CASO DE USO:	CU2- Gestionar Cliente
RESPONSABLE DE LA PRUEBA:	

CASOS DE PRUEBA

CASO DE PRUEBA: CP-01

ESTADO: [x] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: se jecuto correctamente

FIGURAS (evidencias del error):
OBSERVACIÓN (opcional):

CASO DE PRUEBA: CP-02

ESTADO: [x] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: se ejecutó correctamente

FIGURAS (evidencias del error):

OBSERVACIÓN (opcional): no permite ingresar satos menores de cuatro dígitos.

CASO DE PRUEBA: CP-03

ESTADO: [X] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO:): no permite guardar caracteres más de 40 en el caso del nombre.

FIGURAS (evidencias del error):

Registro del error en tiempo de ejecución

```
07 dic 2016 - 10:13:43 [INFO ] postgresql.ClientePostgre (82)===> Ingresando Clientes
07 dic 2016 - 10:13:43 [ERROR] postgresql.ClientePostgre (92)===> NO ES POSIBLE :ERROR: el valor es demasiado largo para el tipo character varying(40)
07 dic 2016 - 10:13:43 [ERROR] gestionarclientes.IngresarCliente (39)===> no se puede guardar :consultar con el Administrador
```

OBSERVACIÓN (opcional):

CASO DE PRUEBA: CP-04

ESTADO: [X] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: No Permite ingresar datos DNI menores de 8 dígitos información incoherente

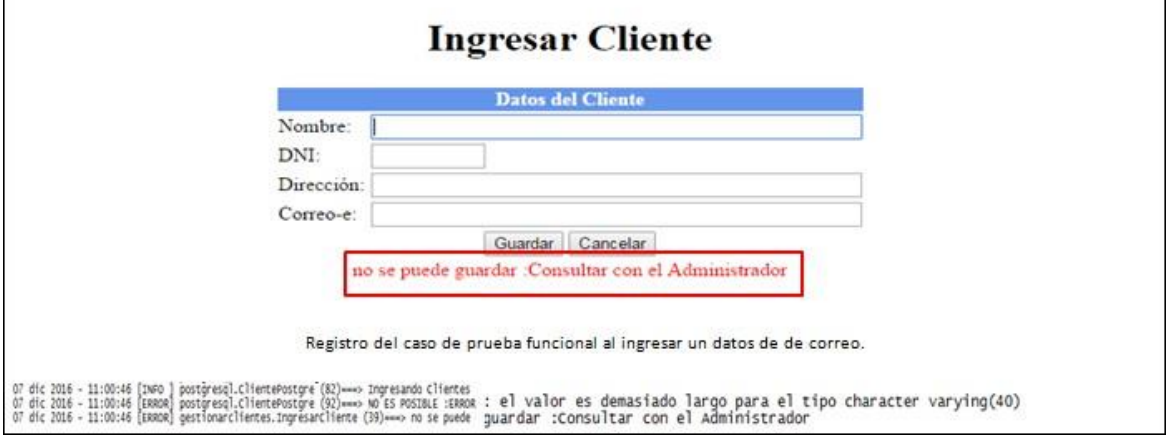
FIGURAS (evidencias del error):

CASO DE PRUEBA: CP-05

ESTADO: [x] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO:

FIGURAS (evidencias del error):



Registro del caso de prueba funcional al ingresar un datos de de correo.

```
07 dic 2016 - 11:00:46 [INFO ] postgresql.ClientePostgre (82)===> Ingresando Clientes
07 dic 2016 - 11:00:46 [ERROR] postgresql.ClientePostgre (92)===> NO ES POSIBLE :ERROR : el valor es demasiado largo para el tipo character varying(40)
07 dic 2016 - 11:00:46 [ERROR] gestionarclientes.IngresarCliente (39)===> no se puede guardar :Consultar con el Administrador
```

CASO DE PRUEBA: CP-06

ESTADO: [x] Sin error [ ] Error menor [ ] Error marginal [ ] Error crítico [ ] Error catastrófico

RESULTADO OBTENIDO: se ejecutó sin errores

FIGURAS (evidencias del error):

Registro de log de la excepción personalizada.

```
07 dic 2016 - 11:10:42 [INFO ] postgresql.ClientePostgre (82)===> Ingresando Clientes
07 dic 2016 - 11:10:42 [ERROR] postgresql.ClientePostgre (92)===> NO ES POSIBLE :ERROR : el valor es demasiado largo para el tipo character varying(50)
07 dic 2016 - 11:10:42 [ERROR] gestionarclientes.IngresarCliente (39)===> no se puede guardar :Consultar con el Administrador
```

Anexo 3. Recursos

Recurso Humanos

Función	Tiempo (Meses)	Pago x mes	Total (S/.)
Tesista	8	0	0
Asesor	8	65	520
Total			S/.520.00
2. 3. 2 7. 2	*"Servicios de consultorías, asesorías y similares desarrollados por personas naturales"*		

Hardware

Equipo	Descripción	Cantidad	Costo(s/.)	Total(s/.)
computadora de escritorio	HD 500 GB Sata PROC INT CORE i3 3.70 GHZ Teclado, Mause Monitor 18.5	1	1500	1500.00
Total				S/.1500.00
2. 6. 3 2. 3	*"Adquisición de equipos informáticos y de comunicaciones"*			

Anexo 5: proforma de compra de computadora

Software

Descripción	Cantidad	Costo x unidad(s/.)	Total(s/.)
Windows 7	1	219.99	219.99
PosgreSQL	1	0.00	0.00
Microsoft Office Professional Plus 2013	1	219.99	219.99
Netbeans 8.1	1	0.00	0.00
Total			s/.439.98

Anexo 6: Costo de Microsoft Office

Anexo 7: Costo de Windows 7

Anexo 8: Licencia de PosgreSQL

Anexo 9: Netbeans

Materiales e Insumos

Descripción	Cantidad	Costo x unidad(s/.)	Total(s/.)
Lapiceros	4	0.50	2.00
Folder	10	1.00	10.00
Fotocopias	100	0.10	10.00
Faster	10	0.30	3.00
Impresiones	1200	0.10	120.00
Anillados	3	4.00	12.00
CD rotulados	2	7	14.00
Total			S/.171.00
2. 3. 2 2. 4 4	*"Servicio de impresiones, encuadernación y empastado"*		

Anexo 10: Boleta de compra

Servicios

Descripción	Nro. de Días	Costo por personal	Total(s/.)
Alimentación	24	25.00	600.00
Movilidad	24	6.00	144.00
Total			S/.744.00

Servicios de internet

Descripción	Tiempo (Meses)	Costo mensual(s/.)	Total(s/.)
Acceso Internet	8	89.00	712.00
Total			S/.712.00

Consumo Energía eléctrica

EQUIPOS	CANTIDAD	TOTAL KW AL MES KW/H	N° DE MESES	COSTOS (S/.) KW/H	IGV (19%)	TOTAL (S/.)
Computadora	1	35	8	0,4	0.19	**S/. 133.28**

Anexo 11: consumo de energia kw/h

Presupuesto

Denominación	Total(s/.)
Recursos humanos	520.00
Hardware	1,500.00
Software	439.98
Materiales e Insumos	171.00
Servicios	744.00
Servicios de internet	712.00
Consumo Energía eléctrica	133.28
Total	**S/.4,220.26**

Anexo 4: formato de validación de casos de prueba funcionales

INSTRUMENTO DE RECOLECCION DE DATOS: CASOS DE PRUEBAS FUNCIONALES

DATOS GENERALES	
Nombre del Sistema:	Sistema de prestamos
Responsable de la prueba:	El investigador
Nombre del Caso de Uso:	Generar Préstamo
Descripción del Caso de Uso:	Se registra los préstamos previamente registrado a los clientes.

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	CLASE VALIDA	CLASE NO VALIDA
El DNI es una cadena de 8 números	1: Cualquier cadena de ocho números.	2: Cadena menos de ocho números.
Monto está entre 10 y 50000	3. 10<= Monto >=50000	4: Monto <10 5:Monto >50000
Tasa efectiva Anual esta entre 1% y 6%	6: 1%<=TE<=6%	7: TE<1% 8:TE>6%
Total periodos cadena de 3 caracteres como máximo	9: Cualquier cadena de 3 caracteres como máximo	10: Cualquier cadena más de 3 caracteres

PARTICIÓN DE EQUIVALENCIAS						
NRO	CLASES DE EQUIVALENCIA	DNI de cliente	Monto	Tasa Efectiva Anual	Total periodos	RESULTADO ESPERADO
CP-01	1,3,6,9	50404123	5000	6	12	El préstamo ha sido registrado
CP-02	1,4,6,9	50404123	5	4	8	El monto es inferior a lo permitido.
CP-03	1,3,5,6,9	44107120	20000	6	10	El préstamo ha sido registrado
CP-04	1,3,7,9	44107120	5000	0	6	Tasa Efectiva Anual es inferior a lo permitido
CP-05	1,3,6,9	44107120	5000	3	12	El préstamo ha sido registrado
CP-06	1,3,6,9,10	44107120	5000	1	4	Total periodo es superior a lo permitido

DATOS GENERALES	
Nombre del Sistema:	Sistema de prestamos
Responsable de la prueba:	El investigador
Nombre del Caso de Uso:	Gestionar Clientes
Descripción del Caso de Uso:	El administrador previamente con usuario y contraseña

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	CLASE VALIDA	CLASE NO VALIDA
El nombre acepta una cadena de 40 caracteres como máximo y mínimo 4	1: cualquier cadena como máximo 40 caracteres y mínimo 4 caracteres	2: cualquier cadena más de 40 caracteres 3. cualquier cadena menos de 4 caracteres
El DNI es una cadena de 8 números	4: Cualquier cadena de ocho números.	5: Cadena menos de ocho números.
Dirección acepta una cadena de 80 caracteres.	6: cualquier cadena como máximo 80 caracteres	7: cualquier cadena más de 80 caracteres
Correo acepta una cadena de 100 caracteres.	8.cualquier cadena como máximo 100 caracteres	9.cualquier cadena más de 100 caracteres

CASOS DE PRUEBA						
NRO	CLASES DE EQUIVALENCIA	Nombre	DNI	Dirección	Correo	RESULTADO ESPERADO
CP-01	1,4,6,8	SANCHEZ HERNANDEZ, KARINA	44520103	SAN ISIDRO 530	SANCHEZ-20@HOTMAIL.COM	Se guardó un nuevo Cliente
CP-02	2,4,6,8	CAR	78945113	AV. SANCHEZ CARRION 7852	ALGA_581@GMAIL.COM	Ingresar Nombre
CP-03	3,4,6,8	MALDONADO TINCO, SANDRA MONICA, MALDONADO TINCO, SANDRA MONICA	64501234	III ETAPA MANUEL AREVALO 984	SANDRA_785@HOTMAIL.COM	Nombre es superior a lo permitido
CP-04	1,5,6,8	MALDONADO TINCO, SANDRA MONICA	6450123	III ETAPA MANUEL AREVALO 984	SANDRA_785@HOTMAIL.COM	DNI es superior a lo permitido verificar
CP-05	1,4,7,8	BEDREGAL CANALES, LUZ MARINA	30604203	BUENOS AIRES 452, Alcaldía de	MANUEL_54_20@HOTMAIL.COM	Dirección es superior a lo permitido verificar

				Plácido Rosas		
CP-06	1,4,6,9	BEDREGAL CANALES, LUZ MARINA	30604203	BUENOS AIRES 452	MANUEL_54_20 @hotmailL.COM, , MALDONADO TINCO, SANDRA MONICA, MONICA	Correo es superior a lo permitido verificar

DATOS GENERALES	
Nombre del Sistema:	Sistema de prestamos
Responsable de la prueba:	El investigador
Nombre del Caso de Uso:	Generar Pago
Descripción del Caso de Uso:	Se registra los préstamos previamente registrado a los clientes.

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	CLASE VALIDA	CLASE NO VALIDA
El DNI es una cadena de 8 números	1: Cualquier cadena de ocho números.	2: Cadena menos de ocho números.
Monto solo acepta como máximo 10 caracteres numéricos	3. cualquier carácter numérico como máximo 10.	4: cualquier carácter numérico más 10 caracteres.

NRO	CLASES DE EQUIVALENCIA	DNI de cliente	Monto pago	RESULTADO ESPERADO
CP-01	1, 3	50404123	400	El pago ha sido realizado
CP-02	1,3	504004	110.00	DNI es inferior a lo permitido.
CP-03	1,4	50404123	150000000000	Monto es superior a lo permitido

UCV
UNIVERSIDAD CÉSAR VALLEJO

PLANTILLAS PARA LA EVALUACIÓN DE INSTRUMENTOS DE RECOLECCIÓN DE DATOS

1. **IDENTIFICACION DEL EXPERTO**

NOMBRE DEL EXPERTO: Lain Cárdenas E.
DNI 18133704 **PROFESION:** Ing. Computación y Sistemas.
LUGAR DE TRABAJO: UCV
CARGO QUE DESEMPEÑA: Docente.
DIRECCION: _____
TELEFONO FIJO: _____ **MOVIL:** _____
DIRECCION ELECTRONICA: laincardenas@gmail.com
FECHA DE EVALUACIÓN: 01/07/2016
FIRMA DEL EXPERTO: _____

2. **PLANILLA DE VALIDACION DEL INSTRUMENTO**

CRITERIOS	APRECIACION CUALITATIVA			
	EXCELENTE (4)	BUENO (3)	REGULAR (2)	DEFICIENTE (1)
Presentación del instrumento	✓			
Claridad en la redacción de los ítems	✓			
Pertinencia de las variables con los indicadores		✓		
Relevancia del contenido		✓		
Factibilidad de la aplicación		✓		

APRECIACION CUALITATIVA: _____

OBSERVACIONES: _____

3. JUICIO DE EXPERTOS:

- En líneas generales, considera Ud. que los indicadores de las variables están inmersos en su contexto teórico de forma:

SUFICIENTE ✓	MEDIANAMENTE SUFICIENTE	INSUFICIENTE

OBSERVACION:

_____
_____
_____
_____

- Considera que los reactivos del cuestionario miden los indicadores seleccionados para la variable de manera:

SUFICIENTE	MEDIANAMENTE SUFICIENTE ✓	INSUFICIENTE

OBSERVACION:

_____
_____
_____
_____

- El instrumento diseñado mide la variable de manera:

SUFICIENTE	MEDIANAMENTE SUFICIENTE ✓	INSUFICIENTE

OBSERVACION:

_____
_____
_____
_____

- El instrumento diseñado es:

_____
_____
_____
_____

4. VALIDACIÓN DEL INSTRUMENTO

ITEM / CASOS DE PRUEBA	ESCALA				OBSERVACIONES
	DEJAR	MODIFICAR	ELIMINAR	INCLUIR	
01	✓				
02	✓				
03	✓				
04	✓				
05	✓				
06	✓				
07	✓				
08	✓				
09	✓				
10	✓				
11	✓				
12	✓				
13	✓				
14	✓				
15	✓				
16					

DESEARIA INCLUIR	COMO LO MODIFICARIA

UCV
UNIVERSIDAD CÉSAR VALLEJO

PLANTILLAS PARA LA EVALUACIÓN DE INSTRUMENTOS DE RECOLECCIÓN DE DATOS

1. **IDENTIFICACION DEL EXPERTO**

NOMBRE DEL EXPERTO: Edwin Mendoza Toro.
DNI 18176811 **PROFESION:** Ing. Informático
LUGAR DE TRABAJO: UCV
CARGO QUE DESEMPEÑA: DOCENTE
DIRECCION: UCV
TELEFONO FIJO: — **MOVIL:** _____
DIRECCION ELECTRONICA: emendozat@ucvvirtual.edu.pe
FECHA DE EVALUACIÓN: 01/07/2016
FIRMA DEL EXPERTO: _____

2. **PLANILLA DE VALIDACION DEL INSTRUMENTO**

CRITERIOS	APRECIACION CUALITATIVA			
	EXCELENTE (4)	BUENO (3)	REGULAR (2)	DEFICIENTE (1)
Presentación del instrumento	✓			
Claridad en la redacción de los ítems	✓			
Pertinencia de las variables con los indicadores		✓		
Relevancia del contenido		✓		
Factibilidad de la aplicación	✓			

APRECIACION CUALITATIVA: _____

OBSERVACIONES: _____

3. JUICIO DE EXPERTOS:

- En líneas generales, considera Ud. que los indicadores de las variables están inmersos en su contexto teórico de forma:

SUFICIENTE X	MEDIANAMENTE SUFICIENTE	INSUFICIENTE

OBSERVACION:

_____
_____
_____
_____

- Considera que los reactivos del cuestionario miden los indicadores seleccionados para la variable de manera:

SUFICIENTE X	MEDIANAMENTE SUFICIENTE	INSUFICIENTE

OBSERVACION:

_____
_____
_____
_____

- El instrumento diseñado mide la variable de manera:

SUFICIENTE X	MEDIANAMENTE SUFICIENTE	INSUFICIENTE

OBSERVACION:

_____
_____
_____
_____

- El instrumento diseñado es:

_____
_____
_____
_____

4. VALIDACIÓN DEL INSTRUMENTO

ITEM / CASOS DE PRUEBA	ESCALA				OBSERVACIONES
	DEJAR	MODIFICAR	ELIMINAR	INCLUIR	
01	✓				
02	✓				
03	✓				
04	✓				
05	✓				
06	✓				
07	✓				
08	✓				
09	✓				
10	✓				
11	✓				
12	✓				
13	✓				
14	✓				
15	✓				
16					

DESEARIA INCLUIR	COMO LO MODIFICARIA

INSTRUMENTO DE RECOLECCION DE DATOS: CASOS DE PRUEBAS FUNCIONALES

DATOS GENERALES	
Nombre del Sistema:	Sistema de prestamos
Responsable de la prueba:	El investigador
Nombre del Caso de Uso:	Generar Préstamo
Descripción del Caso de Uso:	Se registra los préstamos previamente registrado a los clientes.

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	CLASE VALIDA	CLASE NO VALIDA
El DNI es una cadena de 8 números	1: Cualquier cadena de ocho números.	2: Cadena menos de ocho números.
Monto está entre 10 y 50000	3. 10<= Monto >=50000	4: Monto <10 5:Monto >50000
Tasa efectiva Anual esta entre 1% y 6%	6: 1%<=TE<=6%	7: TE<1% 8:TE>6%
Total periodos cadena de 3 caracteres como máximo	9: Cualquier cadena de 3 caracteres como máximo	10: Cualquier cadena más de 3 caracteres

PARTICIÓN DE EQUIVALENCIAS						
NRO	CLASES DE EQUIVALENCIA	DNI de cliente	Monto	Tasa Efectiva Anual	Total periodos	RESULTADO ESPERADO
CP-01	1,3,6,9	50404123	5000	6	12	El préstamo ha sido registrado
CP-02	1,4,6,9	50404123	5	4	8	El monto es inferior a lo permitido.
CP-03	1,3,5,6,9	44107120	20000	6	10	El préstamo ha sido registrado
CP-04	1,3,7,9	44107120	5000	0	6	Tasa Efectiva Anual es inferior a lo permitido
CP-05	1,3,6,9	44107120	5000	3	12l	El préstamo ha sido registrado
CP-06	1,3,6,9,10	44107120	5000	1	4	Total periodo es superior a lo permitido

DATOS GENERALES	
Nombre del Sistema:	Sistema de prestamos
Responsable de la prueba:	El investigador
Nombre del Caso de Uso:	Gestionar Clientes
Descripción del Caso de Uso:	El administrador previamente con usuario y contraseña

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	CLASE VALIDA	CLASE NO VALIDA
El nombre acepta una cadena de 40 caracteres como máximo y mínimo 4	1: cualquier cadena como máximo 40 caracteres y mínimo 4 caracteres	2: cualquier cadena más de 40 caracteres 3. cualquier cadena menos de 4 caracteres
El DNI es una cadena de 8 números	4: Cualquier cadena de ocho números.	5: Cadena menos de ocho números.
Dirección acepta una cadena de 80 caracteres.	6: cualquier cadena como máximo 80 caracteres	7: cualquier cadena más de 80 caracteres
Correo acepta una cadena de 100 caracteres.	8.cualquier cadena como máximo 100 caracteres	9.cualquier cadena más de 100 caracteres

CASOS DE PRUEBA						
NRO	CLASES DE EQUIVALENCIA	Nombre	DNI	Dirección	Correo	RESULTADO ESPERADO
CP-01	1,4,6,8	SANCHEZ HERNANDEZ, KARINA	44520103	SAN ISIDRO 530	SANCHEZ-20@HOTMAIL.COM	Se guardó un nuevo Cliente
CP-02	2,4,6,8	CAR	78945113	AV. SANCHEZ CARRION 7852	ALGA_581@GMAIL.COM	Ingresar Nombre
CP-03	3,4,6,8	MALDONADO TINCO, SANDRA MONICA, MALDONADO TINCO, SANDRA MONICA	64501234	III ETAPA MANUEL AREVALO 984	SANDRA_785@HOTMAIL.COM	Nombre es superior a lo permitido
CP-04	1,5,6,8	MALDONADO TINCO, SANDRA MONICA	6450123	III ETAPA MANUEL AREVALO 984	SANDRA_785@HOTMAIL.COM	DNI es superior a lo permitido verificar
CP-05	1,4,7,8	BEDREGAL CANALES, LUZ MARINA	30604203	BUENOS AIRES 452, Alcaldía de	MANUEL_54_20@HOTMAIL.COM	Dirección es superior a lo permitido verificar

				Plácido Rosas		
CP-06	1,4,6,9	BEDREGAL CANALES, LUZ MARINA	30604203	BUENOS AIRES 452	MANUEL_54_20 @hotmailL.COM, , MALDONADO TINCO, SANDRA MONICA, MONICA	Correo es superior a lo permitido verificar

DATOS GENERALES	
Nombre del Sistema:	Sistema de prestamos
Responsable de la prueba:	El investigador
Nombre del Caso de Uso:	Generar Pago
Descripción del Caso de Uso:	Se registra los préstamos previamente registrado a los clientes.

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	CLASE VALIDA	CLASE NO VALIDA
El DNI es una cadena de 8 números	1: Cualquier cadena de ocho números.	2: Cadena menos de ocho números.
Monto solo acepta como máximo 10 caracteres numéricos	3. cualquier carácter numérico como máximo 10.	4: cualquier carácter numérico más 10 caracteres.

NRO	CLASES DE EQUIVALENCIA	DNI de cliente	Monto pago	RESULTADO ESPERADO
CP-01	1, 3	50404123	400	El pago ha sido realizado
CP-02	1,3	504004	110.00	DNI es inferior a lo permitido.
CP-03	1,4	50404123	150000000000	Monto es superior a lo permitido

UCV
UNIVERSIDAD CÉSAR VALLEJO

PLANTILLAS PARA LA EVALUACIÓN DE INSTRUMENTOS DE RECOLECCIÓN DE DATOS

1. **IDENTIFICACION DEL EXPERTO**

NOMBRE DEL EXPERTO: Marcelino Torres Villanueva
DNI 47865708 **PROFESION:** Ingeniero de Sistemas
LUGAR DE TRABAJO: UCV
CARGO QUE DESEMPEÑA: Docente
DIRECCION: P
TELEFONO FIJO: _____ **MOVIL:** _____
DIRECCION ELECTRONICA: torres.marcelino1@gmail.com
FECHA DE EVALUACIÓN: 01/07/2016
FIRMA DEL EXPERTO:

2. **PLANILLA DE VALIDACION DEL INSTRUMENTO**

CRITERIOS	APRECIACION CUALITATIVA			
	EXCELENTE (4)	BUENO (3)	REGULAR (2)	DEFICIENTE (1)
Presentación del instrumento		✓		
Claridad en la redacción de los ítems		✓		
Pertinencia de las variables con los indicadores	✓			
Relevancia del contenido		✓		
Factibilidad de la aplicación		✓		

APRECIACION CUALITATIVA: _____

OBSERVACIONES: _____

3. JUICIO DE EXPERTOS:

- En líneas generales, considera Ud. que los indicadores de las variables están inmersos en su contexto teórico de forma:

SUFICIENTE X	MEDIANAMENTE SUFICIENTE	INSUFICIENTE

OBSERVACION:

_____
_____
_____
_____

- Considera que los reactivos del cuestionario miden los indicadores seleccionados para la variable de manera:

SUFICIENTE X	MEDIANAMENTE SUFICIENTE	INSUFICIENTE

OBSERVACION:

_____
_____
_____
_____

- El instrumento diseñado mide la variable de manera:

SUFICIENTE X	MEDIANAMENTE SUFICIENTE	INSUFICIENTE

OBSERVACION:

_____
_____
_____
_____

- El instrumento diseñado es:

_____
_____
_____
_____

4. VALIDACIÓN DEL INSTRUMENTO

ITEM / CASOS DE PRUEBA	ESCALA				OBSERVACIONES
	DEJAR	MODIFICAR	ELIMINAR	INCLUIR	
01	✓				
02	✓				
03	✓				
04	✓				
05	✓				
06	✓				
07	✓				
08	✓				
09	✓				
10	✓				
11	✓				
12	✓				
13	✓				
14	✓				
15	✓				
16					

DESEARIA INCLUIR	COMO LO MODIFICARIA

INSTRUMENTO DE RECOLECCION DE DATOS: CASOS DE PRUEBAS FUNCIONALES

DATOS GENERALES	
Nombre del Sistema:	Sistema de prestamos
Responsable de la prueba:	El investigador
Nombre del Caso de Uso:	Generar Prestamo
Descripción del Caso de Uso:	Se registra los préstamos previamente registrado a los clientes.

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	CLASE VALIDA	CLASE NO VALIDA
El DNI es una cadena de 8 números	1: Cualquier cadena de ocho números.	2: Cadena menos de ocho números.
Monto está entre 10 y 50000	3. 10<= Monto >=50000	4: Monto <10 5:Monto >50000
Tasa efectiva Anual esta entre 1% y 6%	6: 1%<=TE<=6%	7: TE<1% 8:TE>6%
Total periodos cadena de 3 caracteres como máximo	9: Cualquier cadena de 3 caracteres como máximo	10: Cualquier cadena más de 3 caracteres

PARTICIÓN DE EQUIVALENCIAS						
NRO	CLASES DE EQUIVALENCIA	DNI de cliente	Monto	Tasa Efectiva Anual	Total periodos	RESULTADO ESPERADO
CP-01	1,3,6,9	50404123	5000	6	12	El préstamo ha sido registrado
CP-02	1,4,6,9	50404123	5	4	8	El monto es inferior a lo permitido.
CP-03	1,3,5,6,9	44107120	20000	6	10	El préstamo ha sido registrado
CP-04	1,3,7,9	44107120	5000	0	6	Tasa Efectiva Anual es inferior a lo permitido
CP-05	1,3,6,9	44107120	5000	3	121	El préstamo ha sido registrado
CP-06	1,3,6,9,10	44107120	5000	1	4	Total periodo es superior a lo permitido

DATOS GENERALES	
Nombre del Sistema:	Sistema de prestamos
Responsable de la prueba:	El investigador
Nombre del Caso de Uso:	Gestionar Clientes
Descripción del Caso de Uso:	El administrador previamente con usuario y contraseña

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	**CLASE VALIDA**	**CLASE NO VALIDA**
El nombre acepta una cadena de 40 caracteres como máximo y mínimo 4	1: cualquier cadena como máximo 40 caracteres y mínimo 4 caracteres	2: cualquier cadena más de 40 caracteres 3. cualquier cadena menos de 4 caracteres
El DNI es una cadena de 8 números	4: Cualquier cadena de ocho números.	5: Cadena menos de ocho números.
Dirección acepta una cadena de 80 caracteres.	6: cualquier cadena como máximo 80 caracteres	7: cualquier cadena más de 80 caracteres
Correo acepta una cadena de 100 caracteres.	8.cualquier cadena como máximo 100 caracteres	9.cualquier cadena más de 100 caracteres

CASOS DE PRUEBA						
NRO	**CLASES DE EQUIVALENCIA**	**Nombre**	**DNI**	**Dirección**	**Correo**	**RESULTADO ESPERADO**
CP-01	1,4,6,8	SANCHEZ HERNANDEZ, KARINA	44520103	SAN ISIDRO 530	SANCHEZ-20@HOTMAIL.COM	Se guardó un nuevo Cliente
CP-02	2,4,6,8	CAR	78945113	AV. SANCHEZ CARRION 7852	ALGA_581@GMAIL.COM	Ingresar Nombre
CP-03	3,4,6,8	MALDONADO TINCO, SANDRA MONICA, MALDONADO TINCO, SANDRA MONICA	64501234	III ETAPA MANUEL AREVALO 984	SANDRA_785@HOTMAIL.COM	Nombre es superior a lo permitido
CP-04	1,5,6,8	MALDONADO TINCO, SANDRA MONICA	6450123	III ETAPA MANUEL AREVALO 984	SANDRA_785@HOTMAIL.COM	DNI es superior a lo permitido verificar
CP-05	1,4,7,8	BEDREGAL CANALES, LUZ MARINA	30604203	BUENOS AIRES 452, Alcaldía de	MANUEL_54_20@HOTMAIL.COM	Dirección es superior a lo permitido verificar

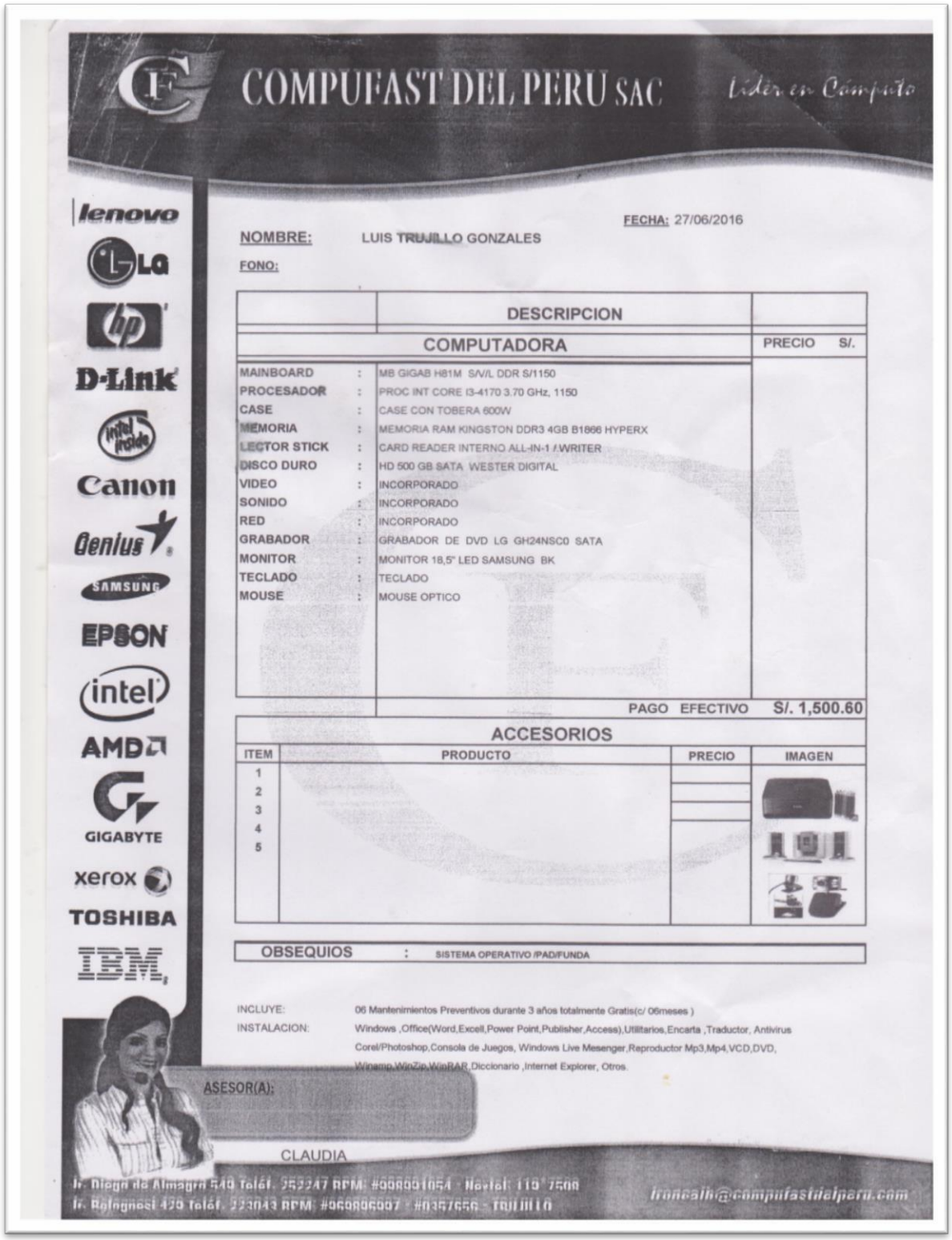
				Plácido Rosas		
CP-06	1,4,6,9	BEDREGAL CANALES, LUZ MARINA	30604203	BUENOS AIRES 452	MANUEL_54_20 @hotmailL.COM, , MALDONADO TINCO, SANDRA MONICA, MONICA	Correo es superior a lo permitido verificar

DATOS GENERALES	
Nombre del Sistema:	Sistema de prestamos
Responsable de la prueba:	El investigador
Nombre del Caso de Uso:	Generar Pago
Descripción del Caso de Uso:	Se registra los prestamos previamente registrado a los clientes.

PARTICIÓN DE EQUIVALENCIAS		
DATO DE ENTRADA	CLASE VALIDA	CLASE NO VALIDA
El DNI es una cadena de 8 números	1: Cualquier cadena de ocho números.	2: Cadena menos de ocho numeros.
Monto solo acepta como máximo 10 caracteres numéricos	3. cualquier carácter numérico como máximo 10.	4: cualquier carácter numérico más 10 caracteres.

NRO	CLASES DE EQUIVALENCIA	DNI de cliente	Monto pago	RESULTADO ESPERADO
CP-01	1, 3	50404123	400	El pago ha sido realizado
CP-02	1,3	504004	110.00	DNI es inferior a lo permitido
CP-03	1,4	50404123	150000000000	Monto es superior a lo permitido

Anexo 5: proforma de compra computadora



COMPUFAST DEL PERU SAC

Líder en Cómputo

FECHA: 27/06/2016

NOMBRE: LUIS TRUJILLO GONZALES

FONO:

	DESCRIPCION	
	COMPUTADORA	PRECIO S/.
MAINBOARD	: MB GIGAB H81M S/V/L DDR S/1150	
PROCESADOR	: PROC INT CORE I3-4170 3.70 GHz, 1150	
CASE	: CASE CON TOBERA 600W	
MEMORIA	: MEMORIA RAM KINGSTON DDR3 4GB B1866 HYPERX	
LECTOR STICK	: CARD READER INTERNO ALL-IN-1 / WRITER	
DISCO DURO	: HD 500 GB SATA WESTER DIGITAL	
VIDEO	: INCORPORADO	
SONIDO	: INCORPORADO	
RED	: INCORPORADO	
GRABADOR	: GRABADOR DE DVD LG GH24NSC0 SATA	
MONITOR	: MONITOR 18,5" LED SAMSUNG BK	
TECLADO	: TECLADO	
MOUSE	: MOUSE OPTICO	
	PAGO EFECTIVO	S/. 1,500.60

ACCESORIOS

ITEM	PRODUCTO	PRECIO	IMAGEN
1			
2			
3			
4			
5			

OBSEQUIOS : SISTEMA OPERATIVO /PAD/FUNDA

INCLUYE: 06 Mantenimientos Preventivos durante 3 años totalmente Gratis(c/ 06meses)

INSTALACION: Windows ,Office(Word,Excell,Power Point,Publisher,Access),Utilitarios,Encarta ,Traductor, Antivirus Corel/Photoshop,Consola de Juegos, Windows Live Mesenger,Reproductor Mp3,Mp4,VCD,DVD, Winamp,WinZip,WinRAR,Diccionario ,Internet Explorer, Otros.

ASESOR(A):

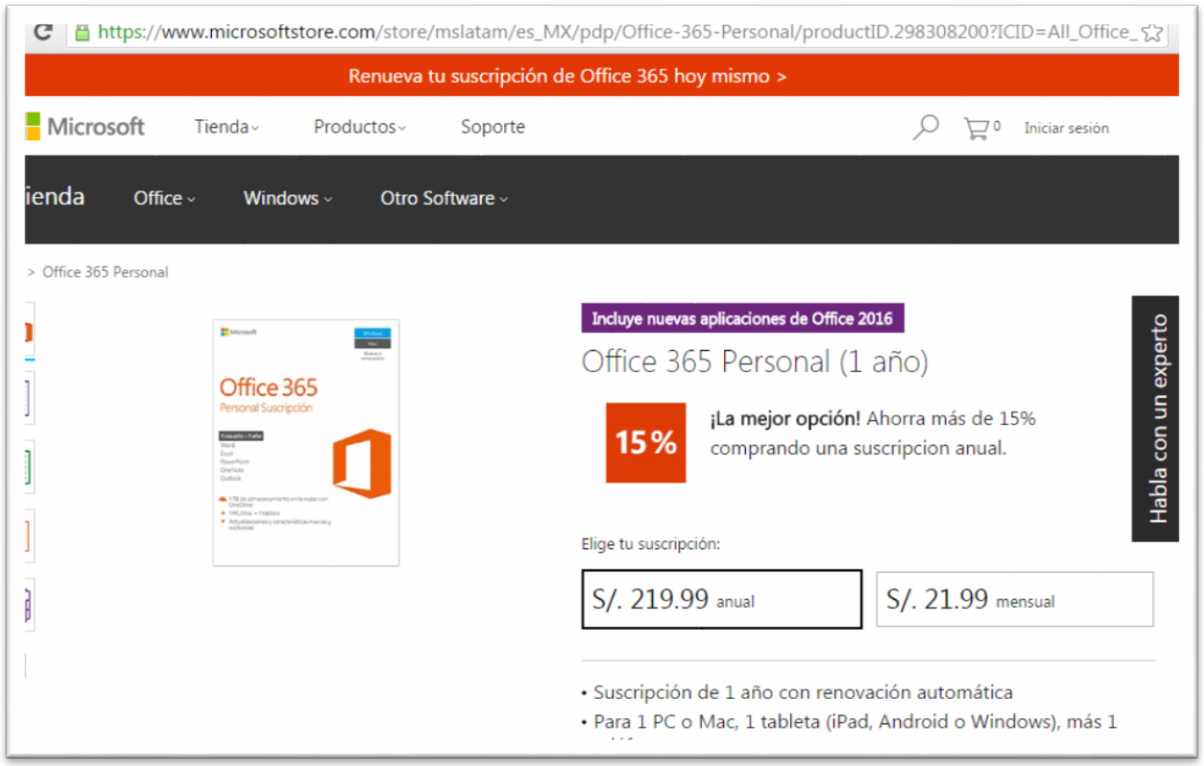
CLAUDIA

Jr. Diego de Almagro 549 Telef. 252247 RPM: #998991054 - Nextel: 119*7508

Jr. Bolognesi 429 Telef. 223043 RPM: #960886007 - #0357656 - TRUJILLO

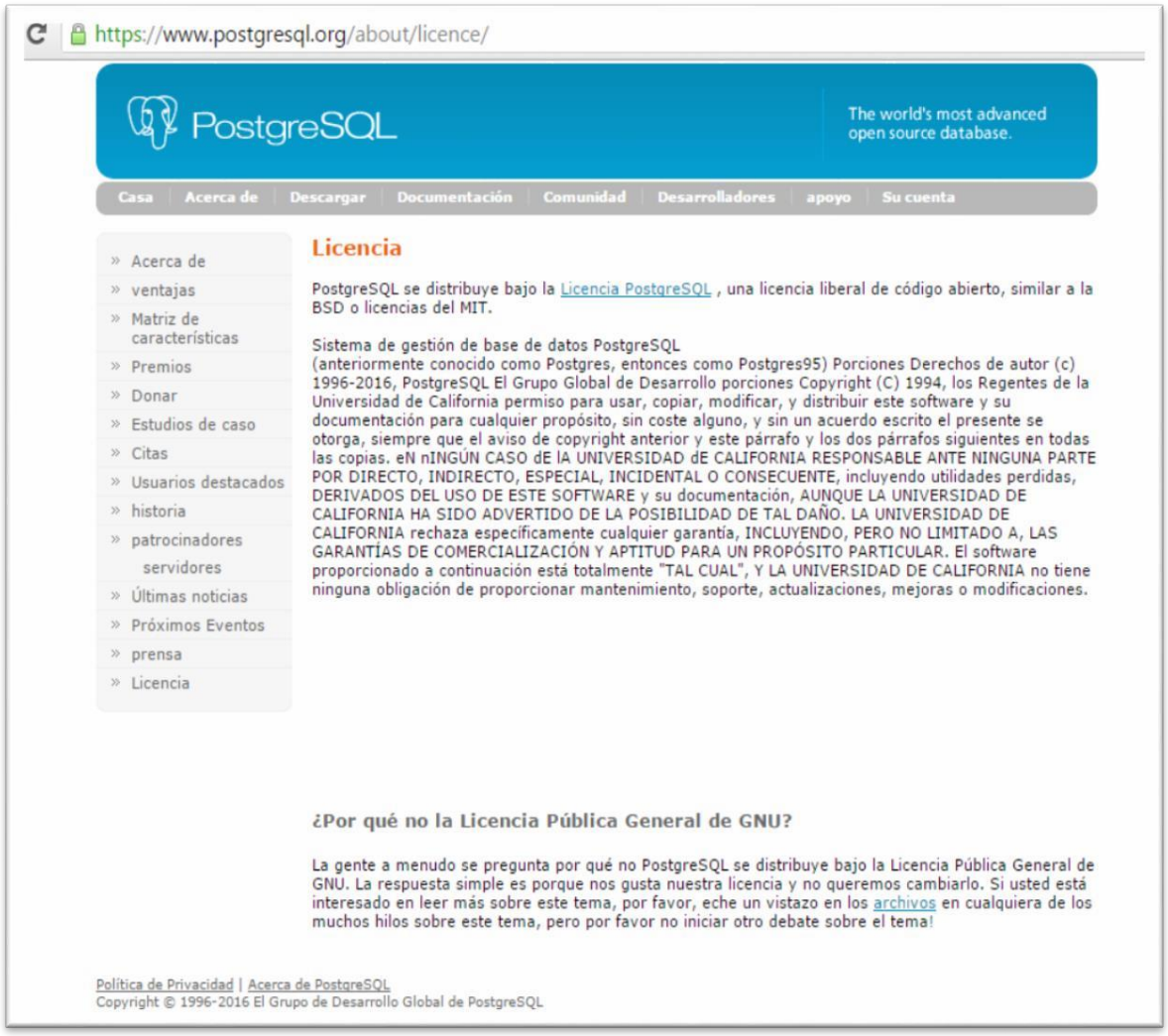
francalh@compufastdelperu.com

Anexo 6: costo de microsoft office



Anexo 7: costo windows 7

Anexo 8: licencia de postgresql



Anexo 9: netbeans 8.1

Anexo 10: boleta de compra

PC Doctor
De: Castro Vásquez Luis Javier

VENTA DE COMPUTADORAS Y SOPORTE TÉCNICO
REPARACIÓN DE LAPTOPS, SOFTWARE Y HARDWARE,
SUMINISTRO Y ACCESORIOS, CÁMARAS DE SEGURIDAD IP,
ADMINISTRACIÓN DE REDES
Mza. A12 Lote. 15 Urb. Manuel Arévalo 3ra Etapa - La Esperanza
Trujillo - La Libertad - Telf.: 044 - 652091
E-mail: informes_ma@pcdoctor.com.pe - www.pcdoctor.com.pe

R.U.C. N° 10180774063
BOLETA DE VENTA
0001 Nº 000179

Sr.(es) Jorge Bautista
Dirección
Fecha 08 06
Doc. Ident.

Cant.	DESCRIPCION	P. UNIT.	IMPORTE
20	Impresiones	1.20	1.20
2	Anillados	4.00	4.00
	Foto copias	3.00	3.00
	Lapicero	0.50	0.50

SON: Ocho con setenta Nuevos Soles

Conceptos Gráficos
Roland Ivan Villanueva Bobadilla
RUC: 10436254208
Psje. 2 Mzna. N°2
Telf.: 693624 - Trujillo
Serie: 0001 Del 000001 al 0001000
Aut. N° 1019220063 - F.I 03/08/2015

CANCELADO
Gracias por Elegirnos...!!!
TOTAL S/. 8.70
USUARIO

UB
BAZAR UNIVERSITARIO S.A.C.

UB BAZAR UNIVERSITARIO S.A.C.
TU MEJOR COMPAÑERO
Av. Los Paujiles Nº 106 Urb. Los Pinos
Trujillo - Trujillo - La Libertad

R.U.C. N° 20481114323
BOLETA DE VENTA
0002- Nº 010222

Señor(es): Dagner
Dirección:
FECHA DE EMISION 20-07-16
D.N.I.

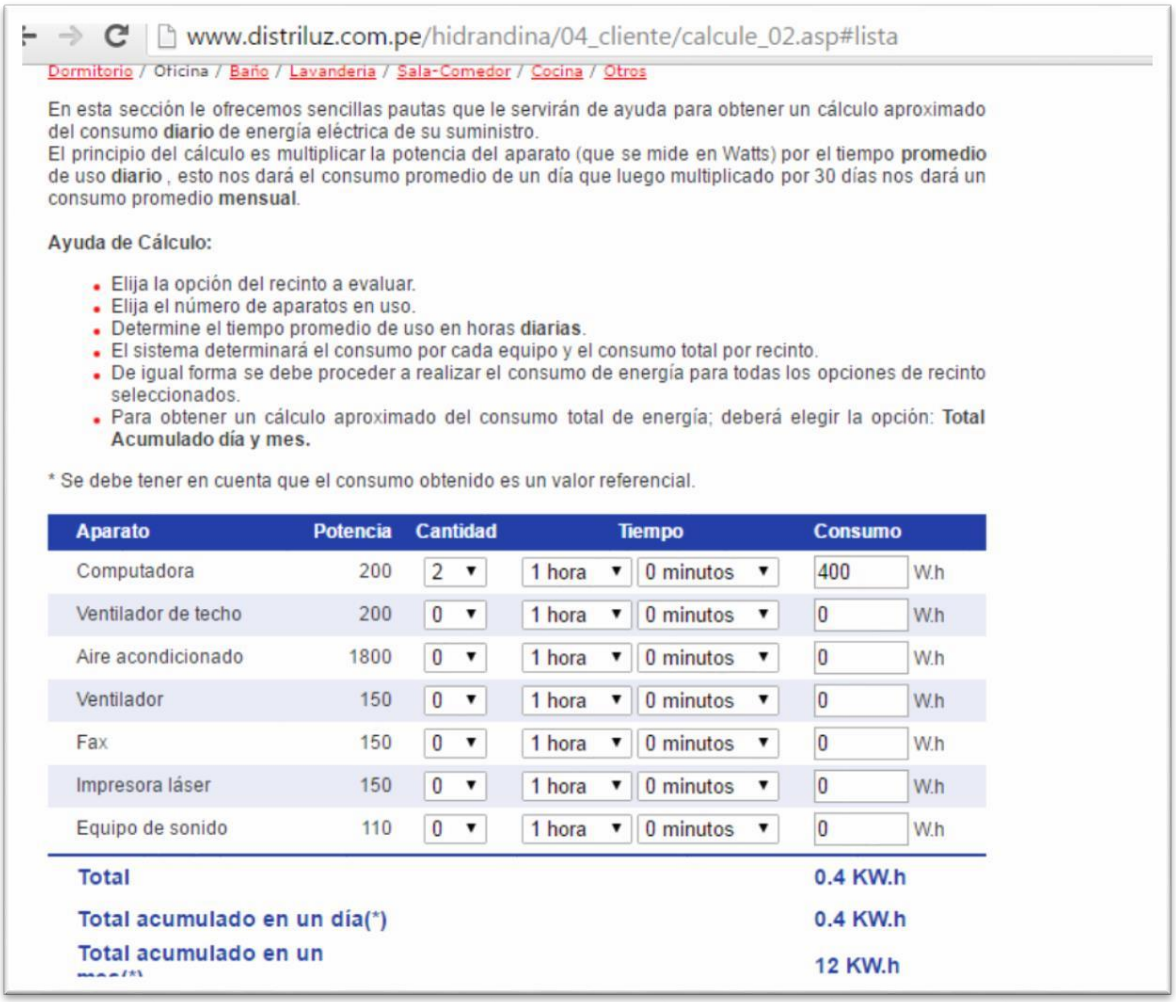
CANT.	DESCRIPCION	P. UNIT.	IMPORTE
	Rotulado		18.00

SON: Nuevos Soles

GRAPA CENTRO DE COPIADO E IMPRENTA S.A.C.
RUC 20440332243
SERIE 0002 DEL 10,001 AL 11,000
AUT. 1083329063 - F.I. 01.06.2016

CANCELADO
TOTAL S/. 18.00
USUARIO

Anexo 11: consumo de energia kw/h



www.distriluz.com.pe/hidrandina/04_cliente/calcule_02.asp#lista

Dormitorio / Oficina / Baño / Lavanderia / Sala-Comedor / Cocina / Otros

En esta sección le ofrecemos sencillas pautas que le servirán de ayuda para obtener un cálculo aproximado del consumo **diario** de energía eléctrica de su suministro.
El principio del cálculo es multiplicar la potencia del aparato (que se mide en Watts) por el tiempo **promedio** de uso **diario** , esto nos dará el consumo promedio de un día que luego multiplicado por 30 días nos dará un consumo promedio **mensual**.

Ayuda de Cálculo:

- Elija la opción del recinto a evaluar.
- Elija el número de aparatos en uso.
- Determine el tiempo promedio de uso en horas **diarias**.
- El sistema determinará el consumo por cada equipo y el consumo total por recinto.
- De igual forma se debe proceder a realizar el consumo de energía para todas los opciones de recinto seleccionados.
- Para obtener un cálculo aproximado del consumo total de energía; deberá elegir la opción: **Total Acumulado día y mes.**

* Se debe tener en cuenta que el consumo obtenido es un valor referencial.

Aparato	Potencia	Cantidad	Tiempo		Consumo
Computadora	200	2	1 hora	0 minutos	400 W.h
Ventilador de techo	200	0	1 hora	0 minutos	0 W.h
Aire acondicionado	1800	0	1 hora	0 minutos	0 W.h
Ventilador	150	0	1 hora	0 minutos	0 W.h
Fax	150	0	1 hora	0 minutos	0 W.h
Impresora láser	150	0	1 hora	0 minutos	0 W.h
Equipo de sonido	110	0	1 hora	0 minutos	0 W.h
Total					**0.4 KW.h**
Total acumulado en un día(*)					**0.4 KW.h**
Total acumulado en un mes(*)					**12 KW.h**

Anexo 12:Matriz de consistencia

MATRIZ DE CONSISTENCIA PARA ELABORACIÓN DE INFORME DE TESIS

NOMBRE DEL ESTUDIANTE:

Dagner Pillamango Mendoza

FACULTAD/ESCUELA:

Ingeniería de Sistema

TÍTULO DEL TRABAJO DE INVESTIGACIÓN	Arquitectura basada en una capa de control de excepciones para mejorar la fiabilidad de la aplicación software de préstamos bancarios.
PROBLEMA	¿De qué manera la arquitectura basada en una capa de control de excepciones influirá en la fiabilidad de la aplicación software de préstamos bancarios?
HIPÓTESIS	La arquitectura basada en una capa de control de excepciones mejora la fiabilidad de la aplicación software de préstamos bancarios.
OBJETIVO GENERAL	Mejorar la fiabilidad de la aplicación software de préstamos bancarios a través de la arquitectura basada en una capa de control de excepciones.
OBJETIVOS ESPECÍFICOS	Aumentar el índice de tolerancia a fallos mediante la capa de control de excepciones. Aumentar el grado de madurez de pruebas mediante la capa de control de excepciones. Aumentar el índice de cobertura de pruebas mediante la capa de control de excepciones

DISEÑO DEL ESTUDIO	Es experimental de tipo pre-experimental
POBLACIÓN Y MUESTRA	La investigación tiene como unidad de estudio una aplicación software de préstamos bancarios.
VARIABLES	Arquitectura basada en una capa de control de excepciones. Fiabilidad de la aplicación software de préstamos bancarios.

OPERACIONALIZACIÓN DE VARIABLES

Variable	Definición conceptual	Definición operacional	indicadores	Escala de medición
Arquitectura basada en una capa de control de excepciones	Una técnica que permite controlar errores durante la ejecución en un software y *"es un mecanismo del lenguaje que permite gestionar errores y situaciones excepcionales"*. (Marco, 2012).	Describir el funcionamiento del manejo de excepciones en el software.	Capa de excepciones Diseño del manejo de excepciones .	Razón
Fiabilidad de la aplicación software de préstamos bancarios.	Una medida del éxito con el que el sistema se ajusta a alguna especificación definitiva de su comportamiento. (Alan Burns, 2003).	Medir la tolerancia a fallos del software.	Porcentaje de Tolerancia de fallos.	Razón
		Medir la madurez del software.	Grado de Madurez de las pruebas.	

			Índice de cobertura de pruebas	

MÉTODOS DE ANÁLISIS DE DATOS	Se utiliza la norma ISO/IEC 2196 para la evaluacion de metricas
RESULTADOS	*Se midió el grado de la tolerancia a fallos obteniendo en la primera interacción un porcentaje de 67% y en la segunda interacción un 97% lo que representa un aumento de un 30% en la tolerancia a fallos.* *Con respecto la madurez de pruebas en la primera interacción se obtuvo un 33% y en la segunda interacción se obtuvo un 83% lo que representa un aumentado un 50% de madurez del software.* *Con respecto a la cobertura de pruebas en la primera interacción se obtuvo un 100% y en la segunda interacción de un 100% lo que representa que en la primera y segunda interacción las coberturas de pruebas se utilizaron todas las diseñadas.* *Los resultados confirman la hipótesis planteada, existe una significativa*

	diferencia entre la situación anterior a la implementación de la capa de excepciones en el sistema en comparación con la situación posterior a la implementación de la capa de excepciones en el sistema. La perspectiva de los resultados en tolerancia a fallos y madures de pruebas es otra y mientras tanto en cobertura de pruebas se mantiene los resultados.
CONCLUSIONES	Se aumentó el porcentaje de Tolerancia de fallos de 67% a un 97% con la implementación de la capa de excepciones en el sistema. Se aumentó el grado de madurez de las pruebas de 33% a un 83% con la implementación de la capa de excepciones en el sistema. Se mantuvo el porcentaje en cuanto al Índice de cobertura de pruebas de un 100% con la implementación de la capa de excepciones en el sistema.