FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

SISTEMA DE ENCRIPTACIÓN PARA OPTIMIZAR EL PROCESO DE DESARROLLO DE SOFTWARE DE UNA EMPRESA DE LIMA

TESIS PARA OBTENER EL TÍTULO PROFESIONAL DE INGENIERO DE SISTEMAS

AUTOR:

JOSUE WENCESLAO RÍOS TAIPE

ASESOR:

DR. ERNESTO FLORES CISNEROS

LÍNEA DE INVESTIGACIÓN:

SISTEMAS DE INFORMACIÓN TRANSACCIONALES

Lima – Perú

2017

Página de Jurado

……

Presidente

……

Secretario

……

Vocal

DEDICATORIA

A DIOS

A ti, mi Dios y salvador, por estar a mi lado y por apoyarme, en todo tiempo, gracias por el amor que siempre me has dado.

A MI FAMILIA

A mi esposa y mis hijas que con su comprensión y apoyo, me ayudaron y apoyaron en todas las decisiones que tomé.

AGRADECIMIENTO

A los docentes de la Universidad Cesar Vallejo por guiarme en todo momento de mi carrera, especialmente al área de investigación de sube por brindarme su conocimiento y su apoyo en todo momento para el desarrollo de la presente investigación. A mis queridas hijas a Luana y Abi que me alentaron para la culminación del presente trabajo.

DECLARACIÓN DE AUTENTICIDAD

Yo, Josue Wenceslao Ríos Taipe, con DNI Nº 41215647, 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 de Sistemas, declaro bajo juramento que toda la documentación que acompaño es veraz y auténtica.

Asimismo, declaro también bajo juramento que todos los datos e información que se muestran 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 la información aportada, por lo cual me someto a lo dispuesto en las normas académicas de la Universidad Cesar Vallejo.

Lima 07, setiembre del 2017

_____

Josue Wenceslao, Ríos Taipe

PRESENTACIÓN

Señores miembros del jurado:

En cumplimiento de las normas establecidas en el Reglamento de Grados y Títulos de la Universidad César Vallejo presento ante ustedes la tesis titulada “SISTEMA DE ENCRIPTACIÓN PARA OPTIMIZAR EL PROCESO DE DESARROLLO DE SOFTWARE DE UNA EMPRESA DE LIMA” la misma que someto a vuestra consideración y espero que cumpla con todos los requisitos de aprobación para obtener el título profesional de Ingeniero de Sistemas.

Esta investigación tiene como objetivo determinar los efectos de la aplicación de unos componentes para el cifrado y firmado de información en proceso de desarrollo de software, la cual consta de siete capítulos.

En el capítulo I se plantea una introducción describiendo la realidad problemática, trabajos previos, teorías relacionadas al tema, formulación del problema, justificación del estudio, hipótesis y los objetivos que lo guían. El capítulo II describe y explica el diseño de la investigación, las variables de estudio y su operacionalización. Adicionalmente, se explica la población, la muestra y se detalla las técnicas e instrumentos para la recolección y procesamiento de la información, la validación y confiabilidad de los instrumentos y los métodos de análisis de los datos y aspectos éticos de la investigación. El capítulo III se refiere a los resultados de la investigación, así como a la comprobación de las hipótesis; en el capítulo IV se presentan y se discuten los resultados de la investigación; en el capítulo V se presentan las conclusiones; en el capítulo VI se presentan las recomendaciones, en el capítulo VII se detallan las referencias bibliográficas utilizadas y finalmente se completa con los anexos.

Esperando señores miembros del jurado que la presente investigación se ajuste a los requerimientos establecidos y que este trabajo de origen a posteriores estudios.

El Autor

Índice General

Índice de Tablas

Índice de Figuras

Índice de Anexos

RESUMEN

La investigación “sistema de encriptación para optimizar el proceso de desarrollo de software de una empresa de Lima”, tuvo como objetivo determinar la influencia del sistema de encriptación para optimizar el proceso de desarrollo de software de una empresa de Lima; al respecto del sistema de encriptación el autor Sergio Talens, propone la necesidad de ejecución tomando en cuenta el nivel seguridad, así también para el proceso de desarrollo de software, Raúl Noriega expresa que se debe evaluar en función, a la productividad y la calidad del software.

La investigación realizada es de tipo aplicada, con un diseño de tipo pre experimental; la población y la muestra está formada por 10 productos de software. Se usó como técnica de recopilación de datos la observación y como instrumento se utilizó la ficha de observación. El instrumento de recolección de datos fue validado mediante el juicio de expertos con un resultado de opinión de aplicabilidad y confiabilidad, el método aplicado para el procesamiento de datos es la estadística descriptiva e inferencial.

Los resultados obtenidos en la investigación determinaron que la implementación del sistema de encriptación influye en productividad laboral del proceso de desarrollo de software de una empresa de lima, incrementado la productividad laboral en 1.3%, esto demuestra que la hipótesis que indica que la implementación del sistema de encriptación incrementa significativamente la productividad laboral del proceso de desarrollo de software de una empresa de lima.

Palabras claves: Sistema de encriptación – Proceso de desarrollo de software

ABSTRACT

The investigation "encryption system to optimize the software development process of a Lima company", aimed to determine the influence of the encryption system to optimize the software development process of a company in Lima; Regarding the encryption system, the author Sergio Talens, proposes the need for execution taking into account the security level, as well as for the software development process, Raúl Noriega says that it should be evaluated based on the productivity and quality of the software..

The investigation carried out is of an applied type, with a pre-experimental type design; The population and sample consists of 10 software products. Observation data was used as a data collection technique and the observation form was used as an instrument. The data collection instrument was validated through the expert judgment with a result of opinion of applicability and reliability, the applied method for data processing is descriptive and inferential statistics.

The results obtained in the investigation determined that the implementation of the encryption system influences labor productivity of the software development process of a lime company, increasing labor productivity by 1.3%, this shows that the hypothesis that indicates the implementation of the system of encryption significantly increases the labor productivity of the software development process of a Lima company.

Keywords: Encryption system - Software development process

I. INTRODUCCIÓN

1.1 Realidad Problemática

Con la aparición de nuevos servicios, crecieron las necesidades de software especializado, dando lugar a la creación de las fábricas de software en el año 1968, acuñado en el congreso IFIP (International Federation of Information Processing), donde se afirmaba que es imposible que los programadores hagan buen software simplemente bajo supervisión humana. Por otro lado en una fábrica, además de la supervisión humana, se mide y controla tanto la productividad como calidad, manteniendo los registros financieros sobre costo y planificación.[1]

Estas fábricas son complejas y dinámicas, con la finalidad de crear software de forma rápida y con niveles altos de calidad, para ello uno de los modelos que se implementan es el de ingeniería de software basado en componentes. El uso de este modelo volvió eficiente el proceso de desarrollo de software, elevando drásticamente los niveles de producción de software.

El desarrollo de software basado en componentes, construye y emplea técnicas de software para elaborar sistemas abiertos y distribuidos, mediante el ensamblaje de partes software reutilizables. Este es utilizado para reducir los costos, tiempo y esfuerzos para el desarrollo del software, y de esta manera incrementar el nivel de productividad de los grupos desarrolladores, minimizar los riesgos y a su vez ayudar a optimizar la fiabilidad, flexibilidad y la reutilización de la aplicación final.[2]

La empresa fue fundada en 1987, dedicada a proveer servicios de desarrollo de aplicaciones, de integración de sistemas, de gerencia de proyectos y de consultoría[3].

Las soluciones que provee la empresa generalmente se enmarcan en el desarrollo de aplicaciones bajo arquitectura Cliente/Servidor, de Sistemas

[1] GARZAS, Javier. Primeras fábricas de software, concepto e historia [En Línea]. [Fecha de consulta: 23 mayo 2017]. Disponible en: http://www.javiergarzas.com/2007/05/primeras-fbricas-software-concepto-e.html

[2] ECURED. Calidad en el Desarrollo Software Basado en Componentes [En Línea]. [Fecha de consulta: 23 febrero 2017]. Disponible en: https://www.ecured.cu/Calidad_en_el_Desarrollo_Software_Basado_en_Componentes

[3] Novatronic. Acerca de [En línea]. [Fecha de consulta: 23 enero 2017]. Disponible en: http://www.novatronic.com/archives/sobre_novatronic/quienes_somos/index.php

Distribuidos y de Internet (Web), utilizando tecnología de Bases de Datos, de Seguridad, de Comunicaciones y de ambiente Visual. El conocimiento y experiencia de su personal, así como la permanente labor de investigación y desarrollo que realiza, le ha permitido brindar soluciones de teleacceso de clientes, de integración de ambientes multiplataforma así como la implementación de redes institucionales, en los sectores de telecomunicaciones, comercial, financiero y gubernamental.[4]

La misión de la empresa es brindar soluciones y servicios transaccionales basados en software especializado que modernice el mundo en que vivimos y su visión es ser la empresa de TI especializada en soluciones y servicios transaccionales, líder en Latinoamérica.

La empresa tiene como clientes a entidades financieras (Bancos y Cajas municipales) y a telcos (Telefónica, Entel y Claro), estas cuentan con productos transaccionales desarrollados y licenciados por la empresa. Entre los principales productos están, los switch transaccionales, sistema de recargas, sistema de pagos, Banca Móvil, homebanking, dinero electrónico, etc.

Sin embargo, durante los últimos 10 años no ha contado con un componente de seguridad que satisfaga las necesidades de protección de la información del mercado peruano o internacional, tales como ley de protección de datos personales en el Perú, en ella se dictamina que la presente Ley tiene el objeto de "garantizar el derecho fundamental a la protección de los datos personales, previsto en el artículo 2 numeral 6 de la Constitución Política del Perú, a través de su adecuado tratamiento, en un marco de respeto de los demás derechos fundamentales que en ella se reconocen".[5]

EL problema por el cual los productos de la empresa no satisfacen las necesidades del mercado, es porque tiene un proceso de desarrollo de software inadecuado, debido a que no implementa nuevas metodologías,

[4] Novatronic. Acerca de [En línea]. [Fecha de consulta: 23 enero 2017]. Disponible en: http://www.novatronic.com/archives/sobre_novatronic/quienes_somos/index.php

[5] MINISTERIO DE JUSTICIA, Ley de protección de datos personales [en línea]. [Fecha de consulta: 23 enero 2017]. Disponible en:https://www.minjus.gob.pe/wp-content/uploads/2013/04/DS-3-2013-JUS.REGLAMENTO.LPDP_.pdf

componentes u herramientas tecnológicas, que le permitan desarrollar software a bajos costos y con altos niveles de calidad. Entre las funcionalidades implementadas en los productos de software, con las cuales tiene una mayor cantidad de reportes de incidencias, son los mecanismos de protección de la información, tales como la encriptación, validación y firma digital. Las implementaciones de software con características de seguridad cuentan con problemas de funcionalidades redundantes, problema excepciones, problemas de sincronismo, en síntesis problemas de calidad código o calidad del software, estos problemas impactaran directamente la productividad laboral, debido a las correcciones o parches que se generan para solucionar los incidentes.

El problema podría ser resuelto con la implementación de un sistema de encriptación, donde los desarrolladores no tendrían la necesidad de codificar la funcionalidad de protección, solo reutilizarían la lógica implementada en la aplicación, por consiguiente, el proceso de desarrollo se beneficiaría, a nivel de productividad y calidad del software.

La encriptación es, hoy por hoy, la forma más segura para proteger la información. Hay diferentes niveles de encriptación, diferentes maneras de encriptarlas y en niveles diferentes con otros algoritmos. Nada en computación es 100% seguro, nada es totalmente infalible. Pero es la encriptación mejor manera, entre los medios más efectivos, de proteger la información. [6]

La investigación de mecanismos de encriptación permitirá tener los conocimientos, las herramientas para proteger la información de los usuarios y los clientes de una empresa de Lima, con ello corregirán los problemas de seguridad de los productos de la empresa, demostrando que la Implementación de un sistema de encriptación optimiza significativamente el proceso de desarrollo de software de una empresa de Lima.

[6] MICROSOFT. Criptografía y seguridad en computadores. [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://support.microsoft.com/es-pe/kb/257591

Para poder demostrar y solucionar el problema de la empresa, se obtendrá la información de los datos de productividad, costo y calidad del software, mediante la recolección de información del proceso de desarrollo de software. Con los datos obtenidos, más el conocimiento teórico, se podrá realizar el estudio y la solución de la problemática, aportando a la empresa de un conocimiento que actual no tiene.

1.2 Trabajos previos

MOYA Caza Johanna B. y ESCOBAR Erazo Franklin A. Desarrollo De Una Aplicación Para Encriptar Información En La Transmisión De Datos En Un Aplicativo De Mensajería Web. Tesis (Titulo de ingeniería de sistemas). Quito, Ecuador: Pontífice universidad católica de Ecuador. Facultad de ingenia de sistemas. (2015. p. 106).

Como objetivo la investigación determinó la influencia de la implementación de diferentes modelos de encriptación para asegurar la confidencialidad en el intercambio de información, el estudio es de tipo aplicada con un diseño pre experimental, con una población y muestra censal del total de personas que utilizan la aplicación, teniendo como técnica la observación y como instrumento la ficha de observación, aunque esto último se deduce de lo expuesto en la tesis.

Entre las conclusiones más importantes se menciona que existe una variedad grande de funciones de encriptamiento, que se debe analizar y realizar las pruebas respectivas para la utilización individual o combinada según los requerimientos. También se menciona que la base conceptual es importante para comprender el funcionamiento de los algoritmos; sin embargo, es necesario realizar pruebas prácticas para validar y evaluar los resultados e ir realizando los ajustes indispensables.

La investigación contribuye de forma significativa en relación al conocimiento sobre encriptación, y sobre el proceso de desarrollo de sistemas de encriptación, también aporta los aspectos relacionados al proceso de negocio y la calidad del software.

CHICHARRO Jesús y BLASCO Jorge. Desarrollo De Una Aplicación Móvil Con Cifrado De Datos Por Geo Posición. Tesis (Título de ingeniero informático). Madrid, España: Universidad Carlos III. Facultad de ingeniería informática. (2013. p. 96)

El estudio determinó que el uso de los métodos y/o herramientas que ofrece el dispositivo móvil para la encriptación asegura que información del usuario sea fidedigna y no sea manipulable, la investigación es de tipo aplicada con un diseño experimental, con una población y muestra censal del total de personas que utilizan la web que implemente la aplicación, teniendo como técnica la observación y como instrumento la ficha de observación.

La conclusión del estudio menciona que la seguridad en los dispositivos móviles, es óptimo mediante el uso de un sistema gestor de información privada que permite descifrar los datos con un par de valores usuario/contraseña,

El contribución de esta tesis es valioso en relación al conocimiento de calidad y encriptación, también es importante marco conceptual en relación a los estándares para el desarrollo de software.

HORNA Lilly. Implementación de la ISO/IEC 12207:2008 para mejorar los procesos asociados al ciclo de vida de software en una micro empresa peruana cuyo objeto social es el desarrollo de sistemas de información. Tesis (Título de Magister en informática). Lima, Perú: Universidad pontífice católica del Perú. Facultad de ingeniería informática. (2013. p. 96)

La tesis determinó que la implementación de la ISO/IEC 12207:2008 mejora el proceso de desarrollo de software en una microempresa, la investigación es de tipo aplicada con un diseño experimental, La población y la muestra son la misma, la totalidad del área de operaciones o desarrollo de la empresa, teniendo técnica la observación y como instrumento la ficha de observación.

La conclusión más importante menciona que las propuestas de mejora fueron resultado de la observación entre lo que se tiene y lo que falta por alcanzar según el nivel objetivo. También se menciona que la elaboración del Plan de

Pruebas permitió integrar varios contenidos asociados que estaban dispersos y servirían para formalizar las pruebas a realizarse en los proyectos de desarrollo de sistemas.

Este estudio es importante por la contribución relacionada con la técnica y el instrumento de recolección de información, para este caso la observación como técnica y la ficha de observación como instrumento.

MARTÍNEZ Yulkeidi. Impacto del desarrollo de software dirigido por modelos en la mejora de productividad, mantenibilidad y satisfacción de aplicaciones Web. Tesis (Título doctorado en informática). Alicante, España: Universidad de alicante. Facultad de ingeniería informática. (2012. p. 246)

La investigación logro el objetivo del aumentar del acervo empírico existente respecto al impacto de las prácticas MDE (Ingeniería dirigida por modelos) sobre la mantenibilidad, productividad, satisfacción y, en último término, intención de uso de los desarrolladores y mantenedores de aplicaciones, para solucionar el problema de del proceso de desarrollo de software. La investigación es de tipo aplicada con un diseño experimental, con una población de 599 y muestra de 100 desarrolladores, la técnica es la observación y como instrumento la ficha de observación, como dimensiones se tiene la productividad, el costo y la calidad del software, etc.

Entre las conclusiones más relevantes se menciona que existe un incremento significativo de la productividad, así como de la mantenibilidad (correctiva y perfectiva) de los desarrollos cuando llevan a cabo tareas de desarrollo y mantenimiento de aplicaciones Web 2.0 con enfoques MDE, en comparación con el desarrollo code-céntrico MBD, también se menciona que el uso de las prácticas de ingeniería MDE permite duplicar la productividad del equipo de desarrollo con respecto al uso de enfoques MBD (con UML) e incrementar hasta 5,75 veces aproximadamente respecto a las prácticas de codificación.

La tesis aporta significativamente a mi variable dependiente en relación a la mejora en la producción de software, así como el marco conceptual para el incremento del conocimiento relacionado al tema tratado.

RUIZ Iván. Un framework para el despliegue y evaluación de procesos software. Tesis (Título doctor en ingeniería). Cádiz, España: Universidad de Cádiz. Facultad de ingeniería y arquitectura. (2013. p. 264)

El objetivo del estudio determinó la mejora los procedimientos necesarios para la evaluación de la calidad en los procesos software, con el fin de solucionar el problema de falta de alineamiento entre los procesos y las herramientas y de la complejidad en la evaluación de procesos software. La investigación es de tipo aplicada con un diseño experimental, como dimensiones a la calidad del software.

Las conclusiones más importantes mencionan que la mejora en los procesos está ligada a la implementación de estándares y herramientas que permitan automatizar los procesos, también es importante medir contantemente los indicadores para buscar una mejora continua de ellas.

El aporte de la tesis está en relación a la dimensión de calidad de desarrollo de software, así como el aporte de conocimiento en relación al tema en mención.

Trinidad Pablo, Segura Sergio. Ruiz Antonio. Evaluación y seguimiento de trabajos en equipo de desarrollo de software a través de la calidad del código fuente. Sevilla, España: Universidad de Sevilla. Facultad de ingeniería y arquitectura. (2012. p. 8)

El artículo científico tiene como objetivo diseño de software mediante el estudio y seguimiento de métricas de calidad del software, con el fin de poder utilizar herramientas para medir la calidad del software. El sonar es una herramienta de software, que evalúa el código e identifica problemas en el código, para ello se ha baso en métricas que ayudan a mejorar la calidad del código.

El contribución de la tesis, son los conocimientos de calidad de desarrollo de software, así como los indicadores de calidad que proporciona la herramienta sonar.

1.3 Teorías relacionadas al tema

1.3.1 Sistema de encriptación

Es un sistema para cifrar y descifrar información compuesto por un conjunto de algoritmos criptográficos, claves y, posiblemente, varios textos en claro con sus correspondientes versiones en texto cifrado. Los sistemas criptográficos actuales se basan en tres tipos de algoritmos criptográficos: de clave secreta o simétrica, de clave pública o asimétrica y de resumen de mensajes (funciones de dispersión)[7]

Es un método matemático que se emplea para cifrar y descifrar un mensaje. Generalmente funciona empleando una o más claves (números o cadenas de caracteres) como parámetros del algoritmo, de modo que sean necesarias para recuperar el mensaje a partir de la versión cifrada.[8]

Encriptación (Cifrado) y Desencriptación (Descifrado)

La encriptación es el proceso para volver ilegible información que se considera importante; en el cual los datos a proteger son traducidos en algo que parece aleatorio y que no tiene ningún significado. Este proceso protege la información para que no pueda ser leída sin una clave. Ahora, los algoritmos que encriptan manejan lo que se denomina llave. La llave es una serie de caracteres de determinado largo, que se utiliza para encriptar y desencriptar la información que se quiere proteger. [9]

Encriptación es el proceso mediante el cual cierta información o texto sin formato es cifrado de forma que el resultado sea ilegible a menos que se conozcan los datos necesarios para su interpretación[10]

[7] TALENS Sergio. Criptografía [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.uv.es/sto/articulos/BEI-2003-04/criptologia.pdf.

[8] TALENS Sergio. Criptografía [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.uv.es/sto/articulos/BEI-2003-04/criptologia.pdf.

[9] JIMÉNEZ, Pablo y QUEZADA, Iván. Desarrollo De Una Aplicación Para La Encriptación Y Desencriptación De La Información De Un Directorio Mediante Autenticación Por Pki Utilizando Tecnología Active Directory [en línea]. Universidad del Azuay. Quito, Ecuador. [Fecha Consulta: julio 2017]. Disponible en: http://dspace.uazuay.edu.ec/bitstream/datos/2685/1/09221.pdf

[10] Gretel, Abdaleis. Encriptación de Datos [En línea]. [Fecha consulta: febrero 2017]. Disponible en: http://encripdedatos.blogspot.pe/

Es una tecnología que permite la transmisión segura de información, al codificar los datos transmitidos usando una fórmula matemática que "desmenuza" los datos. Asegurar que la Información viaje segura, manteniendo su autenticidad, integridad, confidencialidad y el no repudio de la misma entre otros aspectos[11]

El algoritmo es la receta general para el cifrado y tu clave hace únicos los datos cifrados, sólo personas con tu clave única y el mismo algoritmo pueden descifrarlo. Las claves generalmente son una larga secuencia de números protegidos por un mecanismo común de autenticación como contraseñas, tokens o biométricos, como tu huella digital.[12]

Implementación del sistema de encriptación

El sistema de encriptación a desarrollar es un componente que permitirá a distintas aplicaciones acoplar funcionalidades de seguridad, también se podrá desplegar como servicio, para que distintos sistemas consuman sus servicios, tales como encriptación, desencriptación, firmado y validación.

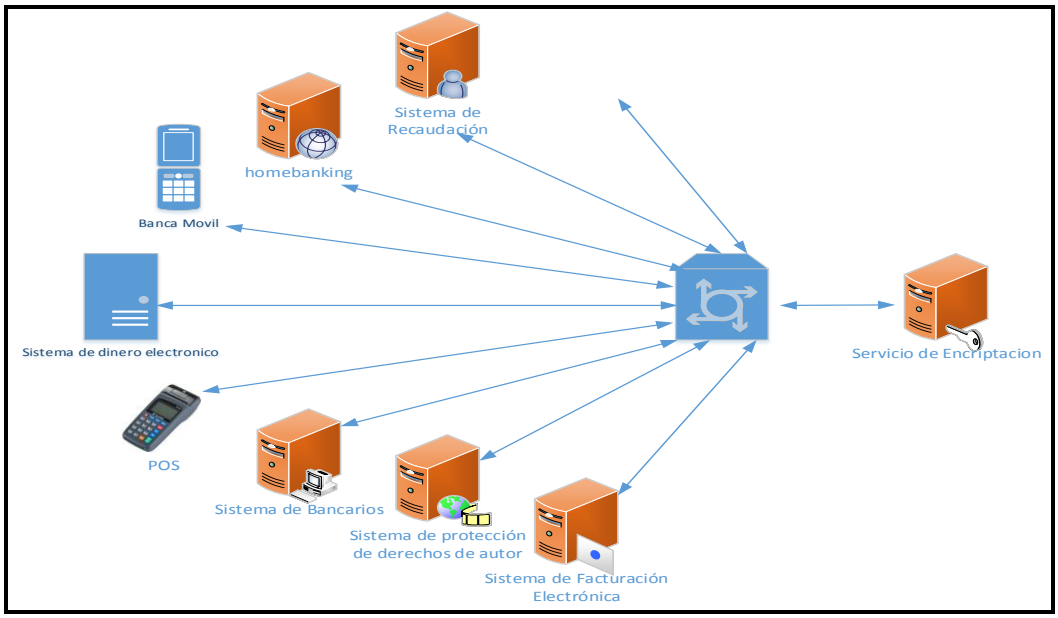
Arquitectura como servicio

Sistema se podrá utilizar dentro de una arquitectura transaccional, donde esta proveerá servicio de criptografía para sistemas que requieran el uso de estas.

[11] Ancelit, Analida. Encriptación de Datos [En línea]. [Fecha consulta: febrero 2017]. Disponible en: http://encriptaciondedatos.blogspot.pe/2007/09/encriptacion-de-datos.html

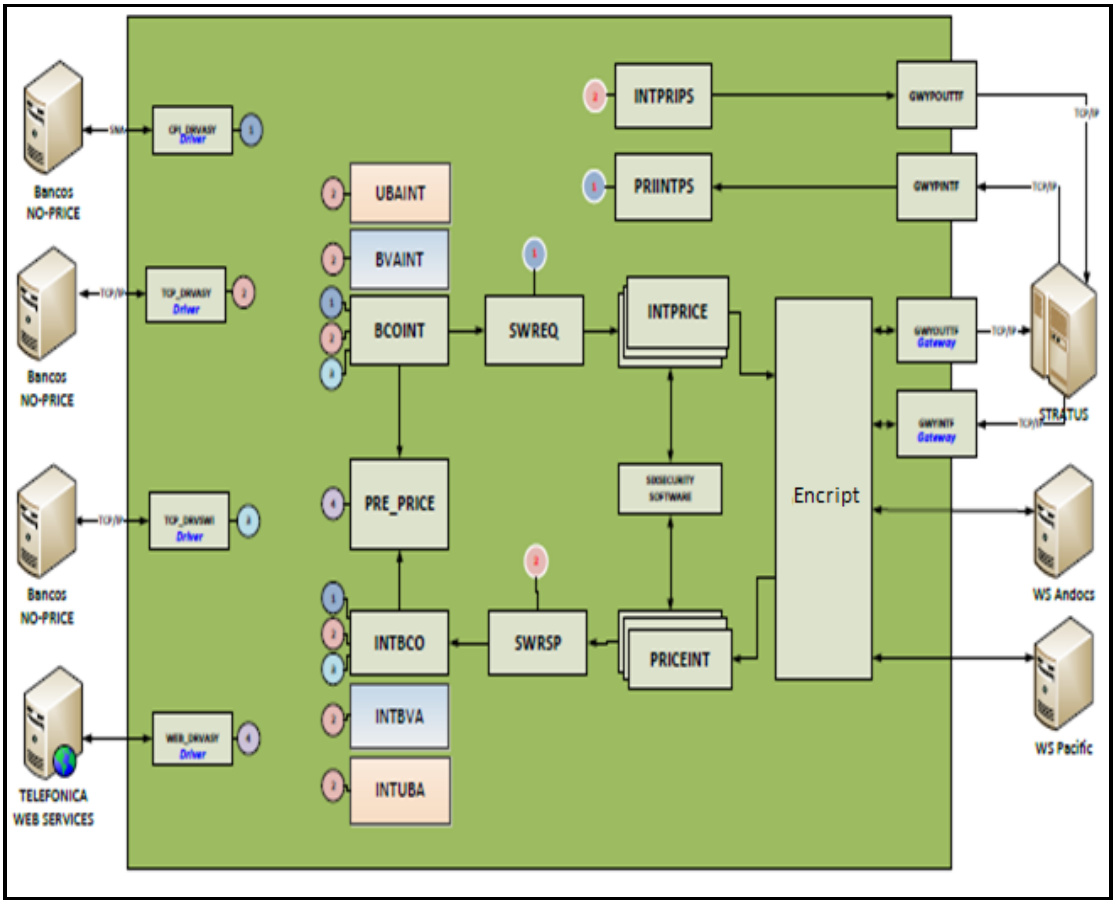
[12] SECURING THE HUMAN. Entendiendo el cifrado [En línea]. [Fecha consulta: febrero 2017]. Disponible en: https://securingthehuman.sans.org/newsletters/ouch/issues/OUCH-201107_sp.pdf

Figura 1: sistema de encriptación como servicios



Fuente: Elaboración propia

Figura 2: sistema de encriptación en una plataforma transaccional

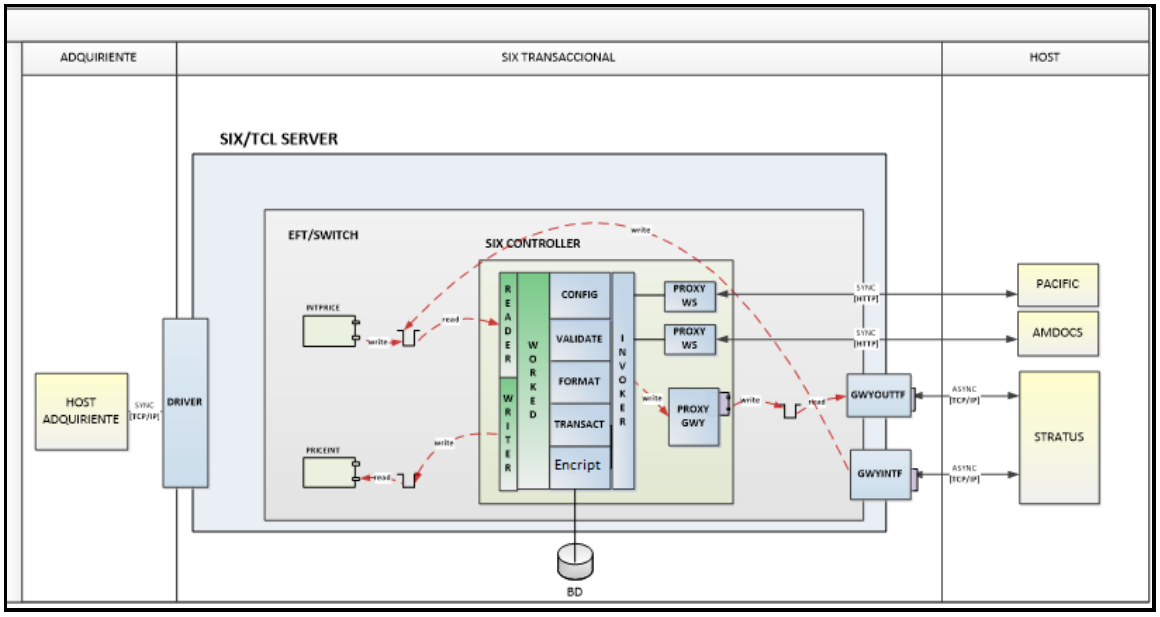


Fuente: elaboración Propia

Arquitectura como componente

el sistema está diseñado para ser utilizado como componentes y des esta forma extender las funcionalidades de las aplicaciones de forma rápida, los desarrolladores deberán utilizar unas librerías, que expondrá unas interfaces, estas exponen los servicios que contienen los componentes y con la implementación de estos, se agrega de forma rápida todas las funcionalidades de criptografía.

Figura 3: Arquitectura como componentes del sistema de encriptación



1.3.2 Firmado Digital (Algoritmo Hash)

Es una tecnología de protección informática que mediante el uso de unos algoritmos(hash) genera un valor (firma) en base a la información contenida en una trama o archivo, esta información es validada por el receptor quien verificará que el contenido de la trama o archivo tengan relación con la firma, asegurando de esta forma su integridad y autenticidad. [13]

Las firmas digitales ayudan a establecer las siguientes garantías:

[13] MICROSOFT. Criptografía y seguridad en computadores. [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://support.microsoft.com/es-pe/kb/257591

Autenticidad**,** una firma digital ayuda a identificar autenticidad de la información. Integridad**,** una firma digital asegura al receptor que la información recibida es la misma que envió el emisor. No renuncia**,** La firma digital ayuda a probar a todas las partes el origen del contenido firmado. Por "renuncia" se entiende el acto de un firmante de negar cualquier asociación con el contenido firmado. Por la garantía que establece la firma digital se usa para autenticar información digital, como documentos, mensajes de correo electrónico y macros, mediante criptografía digital.[14]

1.3.3 Tipos de criptografía: simétrica, asimétrica e híbrida.

Criptografía Simétrica:

Este tipo de criptografía cifra y descifra la información utilizando una misma clave, para ello tanto el emisor como el receptor deben conocer dicha clave.

El punto vulnerable de este tipo de cifrado es la clave, debido a que, si esta es interceptada de algún modo, la información podrá ser sustraída o modificada.

Entre los algoritmos más utilizados están: DES, 3DES, RC5, AES.

Algoritmos de Encriptación Simétrico:

Son funciones matemáticas cuyo objetivo es transformar la información de representación a otro tipo de representación.
Un algoritmo criptográfico, o de cifrado, es una función matemática usada en los procesos de encriptación y desencriptación.[15]

DES (Data Encryption Standard)

Digital Encryption Standard creado en 1975 con ayuda de la NSA (National Security Agency); en 1982 se convirtió en un estándar. Utiliza una clave de 56 bit. En 1999 fue quebrado (violado) en menos de 24 horas por un servidor

[14] MICROSOFT. Criptografía y seguridad en computadores. [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://support.microsoft.com/es-pe/kb/257591

[15] CRYPTOFORGET. Encriptación y seguridad [En Línea]. [Fecha de consulta: 23 enero 2017]. Disponible en: http://www.cryptoforge.com.ar/encriptacion-seguridad-de-la-informacion.htm

dedicado a eso. Esto lo calificó como un algoritmo inseguro y con falencias reconocidas.[16]

3DES (Triple Data Encryption Standard)

Es un algoritmo para cifrar información, fue escogido como un estándar FIPS en los Estados Unidos en 1976, por el avance de la tecnología, 3DES terminó siendo un método inseguro, para dar frente a esta situación se vio como alternativa aplicar el algoritmo DES 3 veces.

La Universidad Nacional Autónoma (2015) menciona que “triple des Fue emitido por el NIST en 1999 como una versión mejorada de DES. La cual consiste en realizar el cifrado DES tres veces utilizando tres claves”.[17]

IDEA (International Data Encryption Algorithm)

Más conocido como un componente de PGP (encriptación de mails), trabaja con claves de 128 bits. Realiza procesos de shift y copiado y pegado de los 128 bits, dejando un total de 52 sub llaves de 16 bits cada una. Es un algoritmo más rápido que el DES, pero al ser nuevo, aún no es aceptado como un estándar, aunque no se le han encontrado debilidades aún[18]

AES (Advanced Encryption Standard)

El algoritmo se basa en varias sustituciones, permutaciones y transformaciones lineales, ejecutadas en bloques de datos de 16 bytes por lo que se le llama blockcipher. Estas operaciones se repiten varias veces, las cuales son llamadas "rondas". En cada ronda, un único “roundkey” se calcula de la clave de encriptación, y es incorporado en los cálculos. [19]

---

[16] JIMÉNEZ, Pablo y QUEZADA, Iván. Desarrollo De Una Aplicación Para La Encriptación Y Desencriptación De La Información De Un Directorio Mediante Autenticación Por Pki Utilizando Tecnología Active Directory [en línea]. Universidad del Azuay. Quito, Ecuador. [Fecha Consulta: julio 2017]. Disponible en: http://dspace.uazuay.edu.ec/bitstream/datos/2685/1/09221.pdf

[17] UNIVERSIDAD NACIONAL AUTÓNOMA. Principales Algoritmos Simétricos - 3DES o TDES) [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en:http://redyseguridad.fi-p.unam.mx/proyectos/criptografia/criptografia/index.php/4-criptografia-simetrica-o-de-clave-secreta/41-introduccion-a-la-criptografia-simetrica/413-principales-algoritmos-simetricos?showall=&start=5

[18] JIMÉNEZ, Pablo y QUEZADA, Iván. Desarrollo De Una Aplicación Para La Encriptación Y Desencriptación De La Información De Un Directorio Mediante Autenticación Por Pki Utilizando Tecnología Active Directory [en línea]. Universidad del Azuay. Quito, Ecuador. [Fecha Consulta: julio 2017]. Disponible en: http://dspace.uazuay.edu.ec/bitstream/datos/2685/1/09221.pdf

[19] BOXCRYPTOR.Cifrado AES y RSA [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://www.boxcryptor.com/es/cifrado

Criptografía Asimétrica:

Este tipo de criptografía cifra la información con una clave privada y descifra la información con una clave pública, esta clave pública es compartida con los receptores y es generada con clave privada, también se utilizan certificados SSL.

Algoritmos de Encriptación Asimétrico:

Son funciones matemáticas cuyo objetivo es transformar la información de representación a otro tipo de representación.

"Un algoritmo criptográfico, o de cifrado, es una función matemática usada en los procesos de encriptación y desencriptación".[20]

SHA (Secure Hash Algorithm)

Es un algoritmo de hash seguro, de síntesis, que genera un hash de 160 bits de un mensaje de tamaño máximo de 264 bits. Es una función de un solo sentido que implica que sea sentido único ya que al mensaje se puede calcular su valor hash, pero no se puede recrear a su mensaje original. [21]

RC5 (Cifrado de Rivest)

RC5 es un algoritmo de cifrado en bloques desarrollado por Ron Rivest, de hecho es lo que indican las siglas que dan nombre al algoritmo y en cuanto al número 5 corresponde a una secuencia de algoritmos de cifrado simétrico desarrollados todos por Rivest y que fueron evolucionando (RC2, RC4 y RC5), donde RC5 se dio a conocer en 1995. Este algoritmo salió en sustitución del esquema RC4, el cual había sido publicado anónimamente en Internet. El candidato para AES, RC6, estaba basado en RC5. [22]

RSA (Rivest, Shamir y Adleman)

[20] CRYPTOFORGET. Encriptación y seguridad [En Línea]. [Fecha de consulta: 23 enero 2017]. Disponible en: http://www.cryptoforge.com.ar/encriptacion-seguridad-de-la-informacion.htm

[21] MOYA. Johanna y ESCOBAR, franklin. Desarrollo De Una Aplicación Para Encriptar Información En La Transmisión De Datos En Un Aplicativo De Mensajería Web. [En línea]. Universidad Católica de Ecuador, Quito. [Fecha de consulta: 23 enero 2017]. Disponible en: http://repositorio.puce.edu.ec/bitstream/handle/22000/8335/Disertacion_MoyaCazaJohannaBeatriz_EscobarErazoFranklinAndres.pdf?sequence=1&isAllowed=y

[22] ECURED. RC5 [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.ecured.cu/index.php/RC5.

RSA es un algoritmo para el cifrado asimétrico, trabaja con dos claves diferentes: una clave "pública", y otra "privada". Ambas son complementarias entre sí (trabajan de manera conjunta) así que un mensaje cifrado con una de ellas sólo puede ser descifrado por su contraparte. Dado que la clave privada no se puede calcular a partir de la clave pública, esta última queda generalmente a disposición del público. Estas propiedades permiten que los cripto-sistemas asimétricos sean utilizados en una amplia variedad de funciones, tales como las firmas digitales. [23]

1.3.4 Cifrado por Software y Hardware.

a. Cifrado por software.

Consiste en utilizar una aplicación para realizar el cifrado de la información, siendo la aplicación la que se encarga de la administración de las claves de seguridad, así como la utilización de los distintos métodos de cifrado y la utilización de los distintos tipos de claves, tales como ZPK (zone per key), TPK (terminar per key), ZMK (Zone Master Key), etc.

Características:

- Comparte los recursos para cifrar los datos con otros programas en el equipo.
- Son tan seguros como el hardware donde se despliega.
- Utiliza el password del usuario como código para cifrar los datos.
- Podría necesitar updates del software
- Económico en aquitecturas de pequeñas aplicaciones
- Es multimedios

b. Cifrado por Hardware.

Consiste en utilizar el hardware que se encargue del cifrado de la información, este hardware se encarga de administrar y proteger los distintos tipos de claves, también se encarga de la ejecución de los distintos comandos de

[23] BOXCRYPTOR.Cifrado AES y RSA [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://www.boxcryptor.com/es/cifrado

seguridad, que permitirán cifrar, validar y generar distintos resultados de seguridad como PINBLOCK, hash, cifrados, etc.

Características:

- Utiliza el procesador en su totalidad para las tareas de seguridad.
- Los servicios de seguridad siempre están activos.
- No necesita ningún tipo de instalador solo hay que llamar a sus comandos de seguridad.
- Protege ante los ataques más frecuentes, como ataques de inicio en frío, códigos malignos y de fuerza bruta.
- Es más seguro, ya que nadie tiene acceso a su lógica de seguridad.

1.3.5 Software de terceros

CryptoForge

Es un programa de encriptación para seguridad personal y profesional. Permite proteger la privacidad de archivos, carpetas, y mensajes confidenciales encriptándolos (cifrándolos) con hasta cuatro algoritmos de encriptación robustos. Una vez que la información ha sido encriptada, puede ser almacenada en un medio inseguro, o transmitida por una red insegura (como Internet), y aun así permanecer secreta. Luego, la información puede ser desencriptada (descifrada) a su formato original.[24]

TrueCrypt

Es un sistema informático para establecer y dar mantenimiento de cifrado sobre un volumen en tiempo real (dispositivo de almacenamiento de datos). Encriptación de volumen en tiempo real significa que los datos se encriptan automáticamente justo antes de ser guardados y se descifran justo después de que se 24 han cargado, sin intervención del usuario. Todo el sistema de

[24] CRYPTOFORGET. Encriptación y seguridad [En Línea]. [Fecha de consulta: 23 enero 2017]. Disponible en: http://www.cryptoforge.com.ar/encriptacion-seguridad-de-la-informacion.htm

archivos se cifra (por ejemplo, nombres de archivos y carpetas, el contenido de cada archivo, el espacio libre, metadatos, etc.).[25]

BestCrypt

Es un producto que encripta automáticamente los datos antes de guardarlos o almacenarlos en un archivo. La información confidencial está codificada mediante un algoritmo, para que los datos sean ilegibles. BestCrypt descifra de forma transparente el archivo una vez que se ha accedido a este al proporcionar la contraseña o tecla correcta.[26]

1.3.6 Seguridad informática.

La seguridad informática es un conjunto de normas y técnicas destinadas a la protección de la información que viaja por medios de sistemas informáticos (software o hardware).

la seguridad de la información es mucho más amplia que la simple protección de los datos a nivel lógico. Para proporcionar una seguridad real hemos de tener en cuenta múltiples factores, tanto internos como externos.[27]

El estándar de niveles de seguridad más utilizado internacionalmente es el TCSEC Orange Book, desarrollado en 1983 de acuerdo a las normas de seguridad en computadoras del Departamento de Defensa de los Estados Unidos. Los niveles describen diferentes tipos de seguridad del Sistema Operativo y se enumeran desde el mínimo grado de seguridad al máximo[28].

Nivel de seguridad:

Está relacionado al nivel de riesgo potencial al cual se está sometido por usar las aplicaciones; a menor riesgo, mayor el nivel de seguridad.[29]

---

[25] TRUECRYPT. encrypted by TrueCryp [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://truecrypt.sourceforge.net/

[26] BESTCRYPT. BestCrypt Volume Encryption [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.jetico.com/products/enterprise-data-protection/bestcrypt-volume-encryption

[27] MICROSOFT. Criptografía y seguridad en computadores. [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://support.microsoft.com/es-pe/kb/257591

[28] SEGU INFO.Seguridad [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.segu-info.com.ar/logica/seguridadlogica.htm.

[29] Project Management Institute, A Guide To The Project Management Body Of Knowledge (PMBOK Guides), 5a. edición.p253

Nivel de Riesgo = probabilidad * Impacto

Probabilidad/ Impacto

1.3.7 Procesos de Desarrollo (Producción) de Software

El proceso de desarrollo de software es un tipo de ingeniería que aplica los principios de la ciencia computacional y de la matemática para alcanzar soluciones con una mejor relación entre el costo y el beneficio para los problemas de software. asimismo se trata de la aplicación sistemática, disciplinada y cuantificable para el desarrollo, operación y mantenimiento de un software.[30]

"Un proceso de desarrollo de software es un conjunto de acciones que permiten transformar de forma eficiente la necesidad de un usuario en una solución de software efectiva. La definición de un proceso de desarrollo de software es una descripción de este proceso en la que se identifican roles, tareas y artefactos, y que permiten guiar a los ingenieros durante el desarrollo de software".[31]

Un proceso de desarrollo de software es la descripción de una secuencia de actividades que deben ser seguida por un equipo de trabajadores para generar un conjunto coherente de productos, uno de los cuales en el programa del sistema deseado[32]

A menudo los procesos de desarrollo de software son definidos usando anotaciones informales o simples documentos de texto en lenguaje natural, debido principalmente a la facilidad con que esto puede ser implementado. Sin embargo, este tipo de anotación hace más difícil la mejora ulterior del proceso. En cambio, las anotaciones formales pueden requerir entrenamiento

[30] NORIEGA Raúl. Proceso de desarrollo de software, 2da edición. on Amazon.com.2010

[31] LAGOS, Alejandro. Mejora Sistemática Del Proceso De Desarrollo De Software De La División De Autoservicio De DTS [en línea]. Universidad de Chile, Santiago. [Fecha consulta: enero 2017], Disponible en: http://repositorio.uchile.cl/bitstream/handle/2250/110974/cf-lagos_as.pdf?sequence=1&isAllowed=y

[32] DRAKE Jose. Proceso de desarrollo de aplicaciones [En Línea]. [Fecha Consulta: octubre 2016]. Disponible en: http://www.ctr.unican.es/asignaturas/Ingenieria_Software_4_F/Doc/M1_08_Proceso.pdf.

previo para desarrollarlas, pero permiten algún tipo de procesamiento automatizado a partir de ellas. [33]

Un proceso de desarrollo de software es una serie de operaciones usadas en la creación de un producto, también se define como un conjunto de tareas, que tienen que ser realizadas para producir un producto de software de alta calidad. Así mismo es el proceso que se sigue para construir el producto de software desde la concepción de una idea, hasta la entrega y el retiro final del sistema[34]

1.3.8 Diseño de procesos basado en patrones de negocio orientado a la gestión, producción y provisión de bienes o servicios

Para el rediseño de los procesos y el posterior diseño e implementación de la herramienta computacional de apoyo hemos elegido la técnica de Diseño de Procesos Basado en Patrones de Negocio. Esta es una metodología creada por el Dr. Oscar Barros V. que ha demostrado ser un excelente modelo de completitud, el cual permite minimizar el riesgo y esfuerzo en cuanto a modelamiento de procesos de negocio se refiere. Esto se logra basado en la experiencia de muchas empresas que han sido estudiadas y analizadas por el Dr. Barros para determinar el patrón de procesos de negocio. Dicho patrón será utilizado como base para modelar la arquitectura de procesos de negocio enfocada en el negocio del proyecto. Esto permite realizar un modelo en el cual evitamos pasar por alto los subprocesos, los flujos y relaciones esenciales entre ellos.[35]

1.3.9 Mejora de Procesos de Software

Existen tres dimensiones críticas en una organización sobre las cuales se enfocan comúnmente los esfuerzos orientados a la mejora del negocio: (1)

[33] LAGOS, Alejandro. Mejora Sistemática Del Proceso De Desarrollo De Software De La División De Autoservicio De DTS [en línea]. Universidad de Chile, Santiago. [Fecha consulta: enero 2017], Disponible en: http://repositorio.uchile.cl/bitstream/handle/2250/110974/cf-lagos_as.pdf?sequence=1&isAllowed=y

[34] Albares, Franklin. Proceso de desarrollo de software [En línea]. [Fecha consulta: febrero 2017]. Disponible en: http://brfranciscoosunaiuty.blogspot.pe/2012/07/proceso-de-desarrollo-de-software.html

[35] ORTUZAR, Juan. Mejoramiento De Procesos De Producción, Mantención Y Soporte de Productos De Software [en línea]. Universidad de Chile, Santiago. [Fecha consulta: enero 2017]. Disponible en: http://www.tesis.uchile.cl/tesis/uchile/2011/cf-ortuzar_je/html/index-frames.html

personas, (2) procedimientos y métodos, y (3) herramientas y equipamiento. Sin embargo, son los procesos los que permiten unir estas dimensiones con un objetivo común, en una misma manera de hacer las cosas dentro de una organización. [36]

CMMI (capability maturity model integration)

CMMI es una colección de mejores prácticas que ayudan a las organizaciones a mejorar sus procesos. Este modelo ha sido desarrollado para la aplicación de mejora de procesos en el desarrollo de los productos y servicios cubriendo todo el ciclo de vida desde su conceptualización hasta su entrega y mantenimiento.

CMMI surge como solución a los problemas que presentaba CMM (capability maturity model), ya que la ejecución de más de un CMM significaba una confrontación entre múltiples definiciones y traslapo de áreas de proceso. En 1998, el gobierno de estados unidos ordenó al instituto de ingeniería de software (SEI) empezar a trabajar sobre la integración CMM, el desarrollo de este proyecto dio inicio a la creación de CMMI, pretende alinear los objetivos para la mejora de procesos con los objetivos de negocio de la organización, siendo utilizado como un marco para organizar y priorizar las actividades, y ayudar a coordinar actividades multidisciplinarias necesarias para el éxito en la construcción del producto o servicio. [37]

1.3.10 Área de producción.

El área productiva o de fabricación es el proceso de mayor generación de valor agregado en cualquier organización. Los sistemas productivos han

[36] LAGOS, Alejandro. Mejora Sistemática Del Proceso De Desarrollo De Software De La División De Autoservicio De DTS [en línea]. Universidad de Chile, Santiago. [Fecha consulta: enero 2017], Disponible en: http://repositorio.uchile.cl/bitstream/handle/2250/110974/cf-lagos_as.pdf?sequence=1&isAllowed=

[37] MÉNDEZ, Ángel. Mejora Del Proceso Software De Una Pequeña Empresa Desarrolladora De Software: Caso Competisoft- Perú - Lim.Omega, Primer Ciclo [en línea]. Universidad la Católica del Perú. Lima, Perú. [Fecha Consulta: abril 2017]. Disponible en: http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/1615/MENDEZ_BAZALAR_ANGEL_COMPETISOFT_LIM_OMEGA.pdf?sequence=1&isAllowed=y

sido el eje de los procesos de desarrollo de las empresas de manufactura e industria alrededor del mundo.[38]

Productividad.

La productividad es la relación entre el resultado de una actividad productiva y los medios que han sido necesarios para obtener dicha producción. Para el campo empresarial la productividad son un conjunto de actividades que se realizan para conseguir los objetivos de una empresa.[39]

[38] INGENIERIAINDUSTRIALONLINE. Producción [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.ingenieriaindustrialonline.com/herramientas-para-el-ingeniero-industrial/producci%C3%B3n

[39] EMPRENDEPYME. La productividad [En línea] [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.emprendepyme.net/que-es-la-productividad-empresarial.html.

Productividad Laboral.

La productividad laboral se mide a través de la relación entre la producción obtenida o vendida y la cantidad de trabajo incorporado en el proceso productivo en un periodo determinado.[40]

Existen dos procedimientos para medirla.

"El método más común es aquél que relaciona la cantidad de producto obtenido con el número de horas trabajadas por el personal durante un periodo determinado, ya sea en una unidad productiva, en un sector de actividad económica o en un país. También la productividad laboral puede medirse a través de la relación entre la cantidad producida y el número de trabajadores ocupados. Esta relación permite evaluar el rendimiento de una unidad productiva en un período determinado. Si en el transcurso del tiempo aumenta la relación entre el volumen vendido y la magnitud del trabajo incorporado, ello significa que el producto promedio del trabajo mejora; si disminuye, entonces el trabajo promedio produce menos".[41]

Figura 4: Productividad laboral o producción por trabajador

$$Produccción\ media\ por\ trabajador\ = \frac{Producción}{Número\ de\ trabajadores}$$

Fuente: "Cálculo del índice de productividad y el costo unitario"; inegi

Productividad de software

La definición tradicional de productividad de software corresponde al número de líneas de código fuente producidas por persona-mes de esfuerzo. Existen muchos problemas con esta definición, partiendo por una definición precisa de lo que es una línea de código: ¿Debe incluir comentarios, líneas en blanco y líneas de declaración de datos? ¿Cómo tratar múltiples instrucciones por línea o código reusado? Después debe definirse claramente qué se determina

[40] INEGI. Calculo de los índices de productividad laboral y del costeo unitario de la mano de obra [En línea] [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.inegi.org.mx/saladeprensa/boletines/ 2017/ipl/ipl 2017_09.pdf

[41] INEGIO. Metodología De Cálculo De Indicadores De Productividad Laboral [En Línea]. [Fecha de consulta: 23 enero 2017]. Disponible en: http://www.inegi.org.mx/inegi/spc/doc/bibliografia/7AC06BCF.pdf

como esfuerzo: ¿Esfuerzo de codificación, de análisis o administrativo? Aun cuando estas cantidades pudieran ser definidas con claridad persiste la paradoja que indica que existe mayor productividad cuando se utiliza un lenguaje de bajo nivel comparado con la utilización de lenguajes de alto nivel.[42]

Costos de Producción.

Los costos de producción (también llamados costos de operación) son los gastos necesarios para mantener un proyecto, línea de procesamiento o un equipo en funcionamiento. En una compañía estándar, la diferencia entre el ingreso (por ventas y otras entradas) y el costo de producción indica el beneficio bruto.[43]

Un indicador complementario al índice de productividad laboral es el costo laboral por unidad producida o costo laboral unitario. Para calcular este indicador, primero se dividen las remuneraciones totales entre el número de horas trabajadas por el personal (o entre el número de trabajadores), obteniéndose las remuneraciones medias, y mediante su comparación en el tiempo se calculan los índices correspondientes. Este índice se divide entre el índice de productividad laboral o producción media.[44]

Figura 5: costo Laboral Unitario

$$Costo\ laboral\ unitario = \frac{Remuneraciones\ medias}{Productividad\ laboral}$$

Fuente: "Cálculo del índice de productividad y el costo unitario"; inegi

[42] VISCONTI, Marcello. Métricas clásicas de Software [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.eici.ucm.cl/Academicos/R_Villarroel/descargas/ingenieriaii/tema8.pdf

[43] ORGANIZACIÓN DE LAS NACIONES UNIDAS. costos de producción [En línea] [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.fao.org/docrep/003/v8490s/v8490s06.htm

[44] INEGIO. Metodología De Cálculo De Indicadores De Productividad Laboral [En Línea]. [Fecha de consulta: 23 enero 2017]. Disponible en: http://www.inegi.org.mx/inegi/spc/doc/bibliografia/7AC06BCF.pdf

1.3.11 Calidad

En los tiempos actuales son muy volátiles desde el punto de vista de los negocios, y para sobrevivir se debe ser capaz de institucionalizar cambios que sean buenos tanto desde el aspecto técnico como desde el aspecto económico, teniendo en cuenta al mismo tiempo los riesgos y las incertidumbres.[45]

Calidad del producto

La calidad de los productos es la percepción de los consumidores respecto a lo bien (o no) que un producto satisface sus deseos y necesidades; constituyen el punto de referencia para la evaluación de la calidad (la calidad se logra como resultado del uso, no de la producción)[46].

Nivel de Calidad

- El nivel de calidad (C) de un servicio consiste en la diferencia (positiva o negativa) que se produce entre la prestación del servicio o producto (P) y las expectativas de los clientes (E). Las expectativas representan lo que los clientes esperan recibir con la prestación del servicio y los clientes hacen la evaluación de la calidad de cualquier tipo de servicio que reciban comparando lo que reciben con lo que esperaban recibir. **El Rendimiento Percibido**: Se refiere al desempeño (en cuanto a la entrega del valor) que el cliente considera haber obtenido luego de adquirir un producto o servicio. **Las Expectativas**: Las expectativas son las "esperanzas" que los clientes tienen por conseguir algo[47]

[45] BARBIERI, Sebastián. Framework de mejora de procesos de desarrollo de software [en línea]. Lic. Alejandro Bianchi. Tesis Magistral. Universidad. Universidad Nacional de la Plata, Argentina. [Fecha de consulta: 02 de Setiembre 2016]. Disponible en: http://postgrado.info.unlp.edu.ar/Carreras/Magisters/Ingenieria_de_Software/Tesis/Sebastian_Barbieri.pdf

[46] BARBIERI, Sebastián. Framework de mejora de procesos de desarrollo de software [en línea]. Lic. Alejandro Bianchi. Tesis Magistral. Universidad. Universidad Nacional de la Plata, Argentina. [Fecha de consulta: 02 de Setiembre 2016]. Disponible en: http://postgrado.info.unlp.edu.ar/Carreras/Magisters/Ingenieria_de_Software/Tesis/Sebastian_Barbieri.pdf

[47] MARKETINGDEPYMES. Calidad de los productos. [En línea] [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.marketingdepymes.com/sala-de-lectura/instrumentos/el-concepto-de-calidad-en-los-productos-tangibles-y-en-los-servicios

Calidad Del Software

La calidad de software es la concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares y procesos de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente.[48]

También es la suma de todos aquellos aspectos o características de un producto o servicio, que influyen en su capacidad para satisfacer las necesidades de los usuarios. La satisfacción del usuario está determinada por la diferencia entre la calidad percibida y la calidad esperada, cuando éste hace uso de un producto o servicio[49]

Métrica

En informática, el término “métrica” hace referencia a la medición del software en base a parámetros predeterminados, como puede ser el número de líneas de código de que consta o el volumen de documentación asociada. A veces en vez de hablar de métrica se usa el término “Indicadores” del software. Algunos ingenieros lo usan como sinónimos mientras que otros les atribuyen significados distintos.[50]

SonarQube

Herramienta de gestión de la calidad del código fuente. Permite recopilar, analizar, y visualizar métricas del código fuente. Está formado por PMD, Checkstyl, Findbugs, Clover y Cobertura. Principalmente es usado con Java, pero da soporte a otros lenguajes. (LGPL)[51].

Métricas de calidad

[48] PRESSMAN Roger S. .Software Engineering: A Practitioner's Approach. McGraw-Hill. Primera edición

[49] CPEIG. Jornada sobre Calidad del Producto Software e ISO 25000, Santiago de Compostela, España, 10 de junio de 2014

[50] APRENDERAPROGRAMAR. Calidad del software. Métricas y fiabilidad de aplicaciones [En Línea]. [Fecha de consulta: 02 de Setiembre 2016]. Disponible en: http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=198:calidad-del-software-metricas-y-fiabilidad-de-aplicaciones-1a-parte-dv00103a&catid=45:tendencias-programacion&Itemid=164

[51] SONARQUBEHISPANO. definición de Métricas. [En Linea]. [Fecha de consulta: 23 febrero 2017]. Disponible en: https://sonarqubehispano.org/pages/viewpage.action?pageId=4980840

Un tipo de medida. Las métricas pueden tener valores variables, o medidas, a lo largo del tiempo. Ejemplos: número de líneas de código, complejidad, etc. Una métrica puede ser cualitativa como densidad de líneas duplicadas, líneas de cobertura por test unitario o cuantitativo por ejemplo número de líneas de código, complejidad.[52]

1. **Cobertura de Código:** mide la cantidad de test de evaluación del método en las clases

   CC= (cantidad de Unit_test / número de métodos) *100

2. **Cumplimiento de Reglas:** mide el cumplimiento a las reglas de programación definidas como estándar de la empresa

   CR = 100-((Violaciones/líneas de código) * 100)

3. **Índice Interdependencia entre paquetes:** mide el número de ciclos entre paquetes, este con la finalidad de no tener dependencias circulares

   IIEP= 2*(dependencia entre archivos/
   número de desentendencia entre directorios) *100

4. **Duplicados**: mide la cantidad de código duplicado en la programación

   D= (Número de líneas duplicadas/Número total de líneas) *100

5. **Complejidad por Método** = mide la complejidad cíclica, con el objetivo que los flujos de código no sean extensos.

   CM= Complejidad/método

6. **LCOM4:** mide la referencia circular entre los métodos, este punto en la codificación es importante para evitar problemas candados de codigo.

   LCOM4= número de atributos comunes de métodos de clases/4

[52] SONARQUBEHISPANO. definición de Métricas. [En Linea]. [Fecha de consulta: 23 febrero 2017]. Disponible en: https://sonarqubehispano.org/pages/viewpage.action?pageId=4980840.

7. **Comentarios:** mide la cantidad de líneas de comentarios en el código, el comentario en el código es importante porque en estas líneas se explica lo codificado.

   COM= (Líneas de comentadas/
   (Líneas de código + líneas comentadas))* 100.

El código fuente se analiza y las métricas y evidencias se almacenan en la base de datos de SonarQube. Los resultados del análisis pueden ser navegados a través de la interfaz web.[53]

1.3.12 Software basado en componentes.

El avance tecnológico, ha provocado que los sistemas computacionales se vuelvan cada vez más complejos, elevando sus costos de implementación, haciéndolos difíciles de desarrollar, instalar y reutilizar debido a estas circunstancias hubo la necesidad de tener un código especializado para las distintas funcionalidades de las aplicaciones, estas se pueden obtener de la Web o pueden desarrollarse.

Los principios fundamentales de la componentizacion son:

Reusable: Los componentes son usualmente diseñados para ser utilizados en escenarios diferentes por diferentes aplicaciones, sin embargo, algunos componentes pueden ser diseñados para tareas específicas. Sin contexto específico: Los componentes son diseñados para operar en diferentes ambientes y contextos. Información específica como el estado de los datos deben ser pasadas al componente en vez de incluirlos o permitir al componente acceder a ellos. Extensible, Un componente puede ser extendido desde un componente existente para crear un nuevo comportamiento. Encapsulado, Los componentes exponen interfaces que permiten al programa usar su funcionalidad. Sin revelar detalles internos, detalles del proceso o estado. Independiente, Los Componentes están diseñados para tener una dependencia mínima de otros componentes. Por lo tanto los componentes

[53] SONARQUBEHISPANO. definición de Métricas. [En Linea]. [Fecha de consulta: 23 febrero 2017]. Disponible en: https://sonarqubehispano.org/pages/viewpage.action?pageId=4980840

pueden ser instalados en el ambiente adecuado sin afectar otros componentes o sistemas[54].

Los beneficios son:

Facilidad de Instalación, cuando una nueva versión esté disponible, usted podrá reemplazar la versión existente sin impacto en otros componentes o el sistema como un todo. Costos reducidos, el uso de componentes de terceros permite distribuir el costo del desarrollo y del mantenimiento. Facilidad de desarrollo, los componentes implementan una interface bien definida para proveer la funcionalidad definida permitiendo el desarrollo sin impactar otras partes del sistema. Reusable, el uso de componentes reutilizables significa que ellos pueden ser usados para distribuir el desarrollo y el mantenimiento entre múltiples aplicaciones y sistemas. Mitigación de complejidad técnica. Los componentes mitigan la complejidad por medio del uso de contenedores de componentes y sus servicios. Ejemplos de servicios de componentes incluyen activación de componentes, gestión de la vida de los componentes, gestión de colas de mensajes para métodos del componente y transacciones.[55]

1.3.13 Comprar o desarrollar software.

Antes que nada, considero que debemos tener claro que todo software estaba basado en un modelo de procesos o negocio. Por tanto, el software se adaptará a "mi proceso o mi negocio" en la medida que el modelo de "mi proceso o mi negocio" se asemeje en parte o gran parte al modelo de negocio o proceso sobre el cual se ha basado el software. Además todo proceso o negocio tiene un modelo definido, correcto o incorrecto, documentado o no, ordenado o no, pero al final es un modelo.[56]

[54] PELÁEZ, Juan. Arquitectura basada en Componentes. [En línea] [Citado el: 05 de octubre de 2015.] . Disponible en: http://geeks.ms/blogs/jkpelaez/archive/2009/04/18/arquitectura-basada-en-componentes.aspx

[55] PELÁEZ, Juan. Arquitectura basada en Componentes. [En línea] [Citado el: 05 de octubre de 2015] . Disponible en: http://geeks.ms/blogs/jkpelaez/archive/2009/04/18/arquitectura-basada-en-componentes.aspx

[56] CASHFLOWCOMPARTE. Comprar o Desarrollar Software [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://cashflowcomparte.blogspot.pe/2009/01/comprar-o-desarrollar-software.html

Ventajas de desarrollar: se establece la funcionalidad del software acorde al modelo de procesos o de negocio de la compañía, resuelven la totalidad de los procesos de la empresa, el software es propiedad del cliente, se puede agrandar y actualizar fácilmente. Desventajas de desarrollar: demasiado tiempo para la implementación de la solución de software, proceso de estabilización del software[57]

Ventajas de Comprar: funcionalidad probada y estable, posibilidad de comprobar la calidad de la solución en implementaciones existentes, menor costo total de implementación, menor tiempo de implementación, Desventajas de compra: funcionalidad cerrada, limitadas posibilidades de modificaciones, por riesgo a cambiar el modelo del software, normalmente adquisición sin código fuentes, necesidad de recursos especializados en el software para el mantenimiento.[58]

¿Cuándo conviene comprar un software a medida?

Menciona que una empresa es competitiva porque su forma de trabajo es particular y diferente a la de las demás empresas, es decir que esa empresa "es" los procesos de gestión que tiene definido. Estos pueden ser formales o informales. Hay empresas con más de 15 años de historia que trabajan de determinada manera, todos los empleados saben que hay que hacer en cada caso que se presenta, y si bien no tienen un manual de procesos definidos, estas empresas tienen procesos de facto muy valiosos porque son esos procesos los que la hacen eficiente y mejor a otras empresas.[59]

1.3.14 Lenguaje de programación Java y C++.

Lenguaje de programación Java

Java es una tecnología que se usa para el desarrollo de aplicaciones que convierten a la Web en un elemento más interesante y útil. Java no es lo

[57] CASHFLOWCOMPARTE. Comprar o Desarrollar Software [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://cashflowcomparte.blogspot.pe/2009/01/comprar-o-desarrollar-software.html
[58] CASHFLOWCOMPARTE. Comprar o Desarrollar Software [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://cashflowcomparte.blogspot.pe/2009/01/comprar-o-desarrollar-software.html
[59] THALU. Software enlatado vs software a medida [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.thalu.net/software-enlatado-vs-software-a-medida/

mismo que javascript, el cual se trata de una tecnología sencilla que se usa para crear páginas web y solamente se ejecuta en el explorador.[60]

Lenguaje de programación C++

C++ es un lenguaje imperativo orientado a objetos derivado del C [1]. En realidad, un supe conjunto de C, que nació para añadirle cualidades y características de las que carecía. El resultado es que como su ancestro, sigue muy ligado al hardware subyacente, manteniendo una considerable potencia para programación a bajo nivel, pero se la han añadido elementos que le permiten también un estilo de programación con alto nivel de abstracción[61]

Java VS C++

C++ es más veloz que Java, aunque, durante los últimos años Java ha disminuido su factor de desempeño contra C++ hasta alrededor del doble en plataformas Linux-Intel. Por lo que es necesario "iluminar a los no iniciados" para que se den cuenta de que la "lentitud de Java" es una leyenda urbana que ha perdurado por más de 15 años sin un verdadero análisis que lo respalde. Por otro lado, es importante reconocer que ningún lenguaje es la "bala de plata", por lo que Java o C++ deben implementarse de acuerdo al problema que queremos resolver. Para acceder a recursos de bajo nivel como drivers, rutinas matemáticas o código embebido de alto desempeño, la respuesta la encontramos en C++. Para programas orientados a negocios o web que requieran facilidad de modificación, portabilidad o simplemente, cuyos problemas de desempeño puedan ser "matados a billetazos" (es decir, pagando por agregar hardware más potente), Java es una buena opción. Para finalizar, esto no debería ser una competencia para ver "quién es el mejor", sino encontrar la manera de complementar ambos lenguajes. Después de todo, ambos son los más populares entre los programadores. En nuestro caso particular, una opción interesante será tener el procesamiento geoespacial en

[60] JAVA. What is java [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://www.java.com/es/about/whatis_java.jsp

[61] ZATOR. Que es C++ [En línea]. [Fecha Consulta: 05 de Octubre de 2016].Disponible en: http://www.zator.com/Cpp/E0_I.htm

C++, mientras dejamos todo lo demás en Java y sus múltiples frameworks de desarrollo.[62]

1.3.15 Conceptos Básicos

Clave o Llave privada: Es la clave generada por el emisor de la información para el cifrado y descifrado, esta clave es compartida con el receptor.

El cifrado simétrico (también conocido como cifrado de clave privada o cifrado de clave secreta) consiste en utilizar la misma clave para el cifrado y el descifrado[63]

Clave o Llave pública: Es una clave generada con la clave privada, esta clave es entregada a los receptores para el descifrado de la información.

En un sistema de cifrado con clave pública, los usuarios eligen una clave aleatoria que sólo ellos conocen (ésta es la clave privada). A partir de esta clave, automáticamente se deduce un algoritmo (la clave pública). Los usuarios intercambian esta clave pública mediante un canal no seguro. [64]

SSL (Secure Sockets Layer): Es un protocolo que protege el medio de comunicación en una red, para ello utiliza cifrado simétrico e intercambio de certificados, esto último ayuda a identificar que la institución que se comunica contigo es quien dice ser.

Sistema de procesamiento de transacciones.

Es un tipo de sistema de información que recolecta, procesa, almacena, exhibe modifica, cancela y recupera toda la información generada por las transacciones producidas en una organización. Son sistemas computarizados que efectúan y registran las transacciones diarias necesarias para dirigir el negocio, es decir, dan servicio a nivel operativo de la organización y se

[62] EVERAC99. Desempeño de C++ vs. Java [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://everac99.wordpress.com/2012/12/13/desempeno-de-c-vs-java-todo-reside-en-la-calidad-del-compilador-y-del-programador-tambien/

[63] CCM. criptografía de clave privada o clave secreta [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://es.ccm.net/contents/126-criptografia-de-clave-privada-o-clave-secreta

[64] CCM. criptografía de clave privada o clave secreta [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://es.ccm.net/contents/126-criptografia-de-clave-privada-o-clave-secreta

utilizan para producir un conjunto de informes programados con regularidad. Los gerentes necesitan los TPS para supervisar el estado de las operaciones internas y las relaciones de la empresa con el entorno externo. Son tan importantes que una falla en ellos puede generar el quiebre de una empresa y de las vinculadas a ella[65].

Sistemas de Automatización de la Oficina y Sistemas de Trabajo del Conocimiento.

Los sistemas de trabajo del conocimiento sirven de apoyo a los trabajadores profesionales, como los científicos, ingenieros y médicos. **Sistema de Procesamiento de datos:** Son aquellos sistemas de información que procesan grandes volúmenes de información generadas en las funciones administrativas, como el control de inventarios. **Sistemas para la administración:** Son sistemas que se basan en los datos obtenidos, el procesamiento de datos. Estos sistemas usan datos para su el análisis, y posterior toma de decisiones. **Sistema de apoyo a la toma de decisiones:** Enfatizan cada etapa de la toma de decisiones gracias a la información obtenida. **Sistemas expertos e inteligencia artificial:** Es el campo principal de los sistemas expertos que permite desarrollar máquinas que cuenten con un desempeño inteligente. [66]

1.3.16 Metodologías de desarrollo

"Un producto del proceso de ingeniería de software que proporciona un enfoque disciplinado para asignar tareas y responsabilidades dentro de una organización del desarrollo. Su meta es asegurar la producción del software de alta calidad que resuelve las necesidades de los usuarios dentro de un presupuesto y tiempo establecidos"[67]

[65] ACADEMIA. Sistema de procesamiento de transacciones fundación universitaria los libertadores [En Línea]. [Fecha de consulta: 02 de Setiembre 2016]. Disponible en: http://www.academia.edu/8957812/SISTEMA_DE_PROCESAMIENTO_DE_TRANSACCIONES_FUNDACI%C3%93N_UNIVERSITARIA_LOS_LIBERTADORES

[66] GRUPO IZAMORAR. Tipos y o clasificación de los sistemas de información. [En Línea]. [Fecha de consulta: 02 de Setiembre 2016]. Disponible en: http://izamorar.com/tipos-y-o-clasificacion-de-los-sistemas-de-informacion/

[67] RUEDA, Julio. Aplicación de la Metodología RUP para el desarrollo rápido de aplicaciones basado en el Estándar J2EE [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en http://biblioteca.usac.edu.gt/tesis/08/08_0308_CS.pdf

Metodología Rational Unified Process (Rup)

El Proceso Unificado de Rational es un proceso de ingeniería del software. Proporciona un acercamiento disciplinado a la asignación de

Tareas y responsabilidades en una organización de desarrollo. Su propósito es asegurar la producción de software de alta calidad que se ajuste a las necesidades de sus usuarios finales con unos costos y calendario predecibles.[68]

Las características principales de RUP son:

Guiado/Manejado por casos de uso: Los casos de uso reemplazan la antigua especificación funcional tradicional y constituyen la guía fundamental establecida para las actividades a realizar durante todo el proceso de desarrollo incluyendo el diseño, la implementación y las pruebas del sistema.

Centrado en arquitectura: La arquitectura involucra los elementos más significativos del sistema y está influenciada entre otros por plataformas software, sistemas operativos, manejadores de bases de datos, protocolos, consideraciones de desarrollo como sistemas heredados y requerimientos no funcionales. [69]

Iterativo e Incremental: Para hacer más manejable un proyecto se recomienda dividirlo en ciclos. Para cada ciclo se establecen fases de referencia, cada una de las cuales debe ser considerada como un mini proyecto cuyo núcleo fundamental está constituido por una o más iteraciones de las actividades principales básicas de cualquier proceso de desarrollo. En concreto RUP divide el proceso en cuatro fases, dentro de las cuales se realizan varias.[70]

[68] MARTÍNEZ Alejandro y MARTÍNEZ Raúl. Guía a Rational Unified Process [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://anaylenlopez.files.wordpress.com/2011/03/trabajo-guia20rup.pdf

[69] MARTÍNEZ Alejandro y MARTÍNEZ Raúl. Guía a Rational Unified Process [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://anaylenlopez.files.wordpress.com/2011/03/trabajo-guia20rup.pdf

[70] MARTÍNEZ Alejandro y MARTÍNEZ Raúl. Guía a Rational Unified Process [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://anaylenlopez.files.wordpress.com/2011/03/trabajo-guia20rup.pdf

Fases: Como ya se ha visto en el apartado anterior, el RUP se divide en cuatro fases, las cuales pasaremos a ver con más detalle.

Inicio

Antes de iniciar un proyecto es conveniente plantearse algunas preguntas: ¿Cuál es el objetivo? ¿Es factible? ¿Lo construimos o lo compramos? ¿Cuánto va a costar? La fase de inicio trata de responder a estas preguntas y a otras más. Sin embargo, no pretendemos una estimación precisa o la captura de todos los requisitos. Más bien se trata de explorar el problema para decidir si vamos a continuar o a dejarlo".[71]

Elaboración

El propósito de la fase de elaboración es analizar el dominio del problema, establecer los cimientos de la arquitectura, desarrollar el plan del proyecto y eliminar los mayores riesgos. Cuando termina esta fase se llega al punto de no retorno del proyecto, a partir de ese momento, pasamos de las relativamente ligeras y de poco riesgo dos primeras fases, a afrontar la fase de construcción, la cual es costosa y arriesgada[72].

Construcción

La finalidad principal de esta fase es alcanzar la capacidad operacional del producto de forma incremental a través de las sucesivas iteraciones. Durante esta fase todos los componentes, características y requisitos, que no lo hayan sido hecho hasta ahora, han de ser implementados, integrados y testeados, obteniéndose una versión del producto que se pueda poner en manos de los usuarios (una versión beta)[73].

Transición

[71] MARTÍNEZ Alejandro y MARTÍNEZ Raúl. Guía a Rational Unified Process [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://anaylenlopez.files.wordpress.com/2011/03/trabajo-guia20rup.pdf

[72] MARTÍNEZ Alejandro y MARTÍNEZ Raúl. Guía a Rational Unified Process [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://anaylenlopez.files.wordpress.com/2011/03/trabajo-guia20rup.pdf

[73] MARTÍNEZ Alejandro y MARTÍNEZ Raúl. Guía a Rational Unified Process [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://anaylenlopez.files.wordpress.com/2011/03/trabajo-guia20rup.pdf

La finalidad de la fase de transición es poner el producto en manos de los usuarios finales, para lo que típicamente se requerirá desarrollar nuevas versiones actualizadas del producto, completar la documentación, entrenar al usuario en el manejo del producto, y en general tareas relacionadas con el ajuste, configuración, instalación y usabilidad del producto.[74]

Metodología Extreme Programming (Xp)

La programación extrema o Extreme Programming (XP) es un enfoque de la ingeniería de software formulado por Kent Beck, autor del primer libro sobre la materia, Extreme Programming Explained: Embrace Change (1999). Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos[75]

PMBOK (Cuerpo del conocimiento de dirección de proyectos)

El cuerpo de conocimiento es reconocido como un conjunto de buenas prácticas en dirección de proyectos, lo cual significa que son aplicables a la mayoría de los procesos y que su aplicación puede contribuir al aumento de las posibilidades de éxito de una amplia variedad de proyectos. Dichas áreas de conocimiento son a la vez llamadas áreas de gestión: integración, alcance, plazos, costos, calidad, recursos humanos, comunicación, riesgos, adquisiciones y stakeholders.Los gestores de proyectos deben tener un conocimiento lo más detallado posible de cada una de las áreas

[74] MARTÍNEZ Alejandro y MARTÍNEZ Raúl. Guía a Rational Unified Process [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://anaylenlopez.files.wordpress.com/2011/03/trabajo-guia20rup.pdf

[75] INGENIERIADESOFTWARE. Extreme-Programing [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html

mencionadas, tanto en los aspectos teóricos como en las cuestiones prácticas vinculadas a procesos, técnicas y herramientas.[76]

Metodología Scrum

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos. En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales[77].

1.3.17 Selección de la Metodología de desarrollo.

La empresa cuenta con una metodología propia de gestión de proyectos, que está basada en los conocimientos del PMBOK, así como en los componentes SCRUM y las herramientas de RUP (ver Anexo 6).

1.3.18 Herramientas utilizadas en el desarrollo.

STARUML: Herramienta UML de licencia gratuita (inicialmente comercial), desarrollada en 1996 y posteriormente en el 2005 modificada por la GLP para el modelamiento de software, basándose en estándares UML y DMA. Muy fácil de usar, debido a la simplicidad y rápida percepción de sus objetos, funciones y características, otra característica fundamental es que su código es compatible con lenguajes como C++ y Java[78]

[76] ESAN. Las Diez áreas de conocimiento del PMI [En Línea]. [Fecha de consulta: 02 de Setiembre 2016]. Disponible en: http://www.esan.edu.pe/apuntes-empresariales/ 2017/08/las-diez-areas-de-conocimiento-segun-el-pmi/
[77] PROYECTOSAGILES. Qué es SCRUM [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://proyectosagiles.org/que-es-scrum
[78] CODIGOPROGRAMACION. starUML [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://codigoprogramacion.com/tag/staruml#.WMMa6m81_IU

ECLIPSE: Eclipse es una plataforma de desarrollo, diseñada para ser extendida de forma indefinida a través de plug-ins. Fue concebida desde sus orígenes para convertirse en una plataforma de integración de herramientas de desarrollo. No tiene en mente un lenguaje específico, sino que es un ide-genérico, aunque goza de mucha popularidad entre la comunidad de desarrolladores del lenguaje java usando el plug-in jdt que viene incluido en la distribución estándar del ide.[79]

1.4 Formulación del problema

Problema General

¿De qué manera influye la Implementación de un sistema de encriptación para optimizar el proceso de desarrollo de software de una empresa de Lima?

Problemas Específicos

PE1: ¿De qué manera influye la implementación del sistema de encriptación sobre la productividad laboral del proceso de desarrollo de software de una empresa de Lima?

PE2: ¿De qué manera influye la implementación del sistema de encriptación en el costo laboral unitario del proceso de desarrollo de software de una empresa de Lima?

PE3: ¿De qué manera influye la implementación del sistema de encriptación en el nivel de cobertura de código de los productos de software de una empresa de Lima?

PE4: ¿De qué manera influye la implementación del sistema de encriptación en el nivel de cumplimiento de reglas de los productos de software de una empresa de Lima?

[79] genbetadev. Eclipse IDE. [En línea] [Citado el: 05 de octubre de 2015.].Disponible en:https://www.genbetadev.com/herramientas/eclipse-ide

PE5: ¿De qué manera influye la implementación del sistema de encriptación en el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima?

PE6: ¿De qué manera influye la implementación del sistema de encriptación en el nivel de código duplicado de los productos de software de una empresa de Lima?

PE7: ¿De qué manera influye la implementación del sistema de encriptación en el nivel de complejidad por método de los productos de software de una empresa de Lima?

PE8: ¿De qué manera influye la implementación del sistema de encriptación en el nivel de LCOM4 de los productos de software de una empresa de Lima?

PE9: ¿De qué manera influye la implementación del sistema de encriptación en el nivel de comentarios de código de los productos de software de una empresa de Lima?

1.5 Justificación

Justificación Teórica

Cuando se señala la importancia que tiene la investigación de un problema en el desarrollo de una teoría científica, ello implica indicar que el estudio realiza una innovación científica para lo cual es necesario hacer un balance o estado de la cuestión del problema, explicar si sirve para refutar resultados de otras investigaciones o ampliar el modelo teórico. [80]

Este estudio se realiza con finalidad de aportar el conocimiento teórico sobre el uso de sistemas de encriptación y sobre el conocimiento del proceso de desarrollo de software, estos conceptos fueron definido por Sergio Talens y Raúl Noriega, en conocimiento aportado, se evidenciará en los resultados de la investigación donde se demostrará el aporte para la mejora de los procesos

[80] ÑAUPAS Humberto. Metodología de investigación. cuarta edición. Bogota, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4.

Justificación Práctica

La justificación práctica se debe de hacer cuando el desarrollo de la investigación ayuda a resolver un problema o por lo menos, propone estrategias que al aplicarse contribuirían a resolverlo[81].

Estas investigaciones se realizan con la finalidad de optimizar el proceso de desarrollo de software de una empresa de lima, con ello se solucionará los problemas de calidad de software y de productividad.

Justificación Metodológica

Cuando se indica que el uso de determinadas técnicas e instrumentos de investigación, pueden servir para otras investigaciones similares, puede tratarse de técnicas o instrumentos novedosos, tales como los test, las pruebas hipótesis, modelos, diagramas, muestreo. [82]

La elaboración y aplicación del sistema de encriptación en el proceso de desarrollo de software, se indicará o validará mediante el uso de métodos científicos, y con ello se demostrará su validez y confiabilidad, esto se realizará mediante el uso de un pre test y post test, con el planteamiento de hipótesis y diagramas estadísticos.

Justificación técnica

Considera que la técnica viene a ser un conjunto de mecanismos, medios y sistemas de dirigir, recolectar, conservar, reelaborar y transmitir los datos. Es también un sistema de principios y normas que auxilian para aplicar los métodos, pero realizan un valor distinto. Las técnicas de investigación se justifican por su utilidad, que se traduce en la optimización de los esfuerzos, la mejor administración de los recursos y la comunicabilidad de los resultados.[83]

[81] BERNAL, César A. Metodología de la investigación. Tercera edición PEARSON EDUCACIÓN, Colombia, 2010

[82] ÑAUPAS Humberto. Metodología de investigación. cuarta edición. Bogota, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4.

[83] VALDERRAMA Santiago. Pasos para elaborar proyectos y tesis de investigación. Tercera edición. Editorial San Marcos Lima, Perú .2013. ISBN: 9786123028787

Para la implementación de un sistema de encriptación y la mejora de los procesos, se utilizará, Rup como metodología de desarrollo de software, PMBOK para gestión de proyecto, Scrum como marco de dirección y para la mejora de los procesos la metodología CMMI, esta última se encuentra implementada en la empresa.

Justificación Económica:

La finalidad de la evaluación económica es la de suministrar suficientes elementos de juicio sobre los costos y beneficios del proyecto, para que se pueda establecer la conveniencia al uso propuesto de los recursos económicos que se solicitan.[84]

El uso del sistema de encriptación, como servicio o como componente, disminuirá los costos, incrementará la producción y aumentará el nivel de calidad y seguridad de los productos. La mejora del software lo vuelve más atractivo al mercado, teniendo como respuesta un incremento en las ventas.

1.6 Hipótesis

Hipótesis General

La Implementación del sistema de encriptación optimiza significativamente el proceso de desarrollo de software de una empresa de Lima.

Hipótesis Específicos

HE1: La implementación del sistema de encriptación incrementa significativamente la productividad laboral del proceso de desarrollo de software de una empresa de Lima.

HE2: La implementación del sistema de encriptación reduce de manera significativa el costo laboral unitario del proceso de desarrollo de software de una empresa de Lima.

[84] AOS, Justificación económica [En línea] [Citado el: 05 de octubre de 2015.]. Disponible en: http://www.oas.org/dsd/publications/unit/oea42s/ch06.htm#f. justificación económica

HE3: La implementación del sistema de encriptación incrementa de manera significativa el nivel de cobertura de código de los productos de software de una empresa de Lima.

HE4: La implementación del sistema de encriptación incrementa significativamente el nivel de cumplimiento de reglas de los productos de software de una empresa de Lima.

HE5: La implementación del sistema de encriptación disminuye significativamente el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima.

HE6: La implementación del sistema de encriptación Disminuye significativamente el nivel de código duplicado de los productos de software de una empresa de Lima.

HE7: La implementación del sistema de encriptación disminuye significativamente el nivel de complejidad por método de los productos de software de una empresa de Lima.

HE8: La implementación del sistema de encriptación disminuye significativamente el nivel de LCOM4 de los productos de software de una empresa de Lima.

HE9: La implementación del sistema de encriptación incrementa significativamente el nivel de comentarios de código de los productos de una empresa de Lima.

1.7 Objetivos

Objetivo General

Determinar la influencia de la Implementación del sistema de encriptación para optimizar el proceso de desarrollo de software de una empresa de Lima.

Objetivos Específicos

OE1: Determinar la influencia de la implementación del sistema de encriptación sobre la productividad laboral del proceso de desarrollo de software de una empresa de Lima.

OE2: Determinar la influencia de la implementación del sistema de encriptación en el costo laboral unitario del proceso de desarrollo de software de una empresa de Lima.

OE3: Determinar la influencia de la implementación del sistema de encriptación en el nivel de cobertura de código de los productos de software de una empresa de Lima.

OE4: Determinar la influencia de la implementación del sistema de encriptación en el nivel de cumplimiento de reglas de los productos de software de una empresa de Lima.

OE5: Determinar la influencia de la implementación del sistema de encriptación en el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima.

OE6: Determinar la influencia de la implementación del sistema de encriptación en el nivel de código duplicado de los productos de software de una empresa de Lima.

OE7: Determinar la influencia de la implementación del sistema de encriptación en el nivel de complejidad por método de los productos de software de una empresa de Lima.

OE8: Determinar la influencia de la implementación del sistema de encriptación en el nivel de LCOM4 de los productos de software de una empresa de Lima.

OE9: Determinar la influencia de la implementación del sistema de encriptación en el nivel de comentarios de código de los productos de software de una empresa de Lima.

II. METODO

2.1 Tipo de investigación

Investigación Aplicada: Es aquella que está orientada a resolver objetivamente los problemas del proceso de producción, distribución, circulación y consumo de bienes y servicios, de cualquier actividad humana, comercial, comunicacional, servicios, etc.[85]

Porque la investigación busca dar solución al problema de la empresa mediante la implementación de sistema de encriptación y mediante el incremento de conocimiento de los temas relacionados.

2.2 Diseño de investigación

El diseño de la investigación es pre experimental que consiste en: "son aquellos que no reúnen los requisitos de experimentos puros y por lo tanto no tiene validez interna, pero realizan un control mínimo[86]: G X O

Donde:

G: Grupo de sujetos o casos.

X: Tratamiento, estímulo o condición experimental (presencia de algún nivel o modalidad de la variable independiente).

0: Medición de los sujetos de un grupo o de la variable dependiente

Son cuatro las características principales de estos diseños:

En este diseño se efectúa una observación antes de introducir la variable independiente (O1) y otra después de su aplicación (O2). Por lo general las observaciones se obtienen a través de la aplicación de una prueba u observación directa, cuyo nombre asignado depende del momento de aplicación. Si la prueba se administra antes de la introducción de la variable

[85] ÑAUPAS Humberto. Metodología de investigación. cuarta edición. Bogota, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4.

[86] ÑAUPAS Humberto. Metodología de investigación. cuarta edición. Bogota, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4.

independiente se le denomina pre test y si se administra después entonces se llama post test.[87]

Dónde:

O: una medición a los sujetos de un grupo (pre: prueba previa al tratamiento, post: prueba posterior al tratamiento)
O1: Medición Previa al tratamiento (sin el sistema informático)
O2: Medición Posterior al tratamiento (con el sistema informático)
X: tratamiento, estímulo o condición experimental (Sistema Informático).

En este diseño se aplica un pre-test (O1) a una variable, después la aplicación de la variable independiente Sistema Web (X) y finalmente el post-test (O2). El resultado es el cambio ocurrido desde el pre-test hasta el post-test". [88]

2.3 Variables, Operacionalización

Definición Conceptual.

Variable independiente (VI): Sistema de encriptación.

Es un sistema para cifrar y descifrar información compuesto por un conjunto de algoritmos criptográficos, claves y, posiblemente, varios textos en claro con sus correspondientes versiones en texto cifrado. Los sistemas criptográficos actuales se basan en tres tipos de algoritmos criptográficos: de clave secreta o simétrica, de clave pública o asimétrica y de resumen de mensajes (funciones de dispersión)[89]

Variable dependiente (VD): Proceso de Desarrollo de software

El proceso de desarrollo de software es un tipo de ingeniería que aplica los principios de la ciencia computacional y de la matemática para alcanzar soluciones con una mejor relación entre el costo y el beneficio para los problemas de software. Asimismo se trata de la aplicación sistemática,

[87] CAMPBELL, Donald y STANLEY, Julian. Diseños experimentales y cuasi experiméntales en la investigación social, 1978
[88] CAMPBELL, Donald y STANLEY, Julian. Diseños experimentales y cuasi experiméntales en la investigación social, 1978,
[89] TALENS Sergio. Criptografía [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.uv.es/sto/articulos/BEI-2003-04/criptologia.pdf

disciplinada y cuantificable para el desarrollo, operación y mantenimiento de un software.[90]

Definición Operacional

Variable independiente (VI): Sistema de Encriptación.

Es un sistema transaccional, que mediante el uso de métodos de encriptación volverá ilegible la información que se desee proteger, este sistema también cuenta protegerá la integridad de la información mediante la generación y validación de las firmas.

Variable dependiente (VD): Proceso de producción de software

El proceso de desarrollo de software es un conjunto de procesos que tiene como finalidad mejorar la productividad, disminuir los costos e incrementar la calidad del software, esto se mide mediante un análisis numérico de los indicadores, teniendo como instrumento la ficha de observación.

[90] TALENS Sergio. Criptografía [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.uv.es/sto/articulos/BEI-2003-04/criptologia.pdf

Tabla 1: Variable dependiente

Variables	Definicion conceptual	Definicion operacional	Dimensiones	Indicadores	Escala
Proceso de desarrollo de software	Un proceso de desarrollo de software es la descripción de una secuencia de actividades que deben ser seguida por un equipo de trabajadores para generar un conjunto coherente de productos, uno de los cuales en el programa del sistema deseado[91]	El proceso de desarrollo de software es un conjunto de procesos que tiene como finalidad mejorar la productividad, disminuir los costos e incrementar la calidad del software, esto se mide mediante un análisis numérico de los indicadores, teniendo como instrumento la ficha de observación.	Productividad	**Productividad laboral**=Producción/ Número de trabajadores	Razón
				Costo laboral unitario = Remuneración Media/Productividad laboral	Razón
			Calidad de software	1. **Cobertura de Código** = (#Unit_test/# métodos) *100 2. **Cumplimiento de Reglas** = 100-((Violaciones/líneas de código) * 100) 3. **Índice Interdependencia entre paquetes**=2*(dependencia entre archivos/número de desentendencia entre directorios) *100 4. **Duplicados** = (Número de líneas duplicadas/Número total de líneas) *100 5. **Complejidad por Método** = #Complejidad/#método 6. **LCOM4**=#atributos comunes de métodos de clases/4 7. **Comentarios**=(Líneas de comentadas/(líneas de código + líneas comentadas))* 100	Razón

Fuente: Elaboración Propia

[91] DRAKE Jose. Proceso de desarrollo de aplicaciones [En Línea]. [Fecha Consulta: octubre 2016]. Disponible en: http://www.ctr.unican.es/asignaturas/Ingenieria_Software_4_F/Doc/M1_08_Proceso.pdf.

2.4 Población, muestra

Población.

La población es un conjunto de objetos, eventos o hechos que se van a estudiar, también es un conjunto de individuos, personas o instituciones"[92]

La población que se considerará para el desarrollo de la investigación está conformada por las 10 implementaciones de productos de software de una empresa de Lima.

Muestra

Es el sub conjunto o parte del universo o población, seleccionados por métodos diversos, pero siempre teniendo en cuenta la representatividad del universo. [93]

Fórmula para el cálculo de la Muestra:

Con un error tolerado del 5% poblaciones de 10 y un nivel de significación de 1.73.

$$n = \frac{N * Z^2 * p * (1 - p)}{(N - 1) * e^2 + Z^2 * p * (1 - p)}$$

Dónde:

Z = 1.73 = nivel de significación. (Confianza)

P = 0.5 = Proporción de éxito.

N = 10 = Número de población.

E = 0.05 = Error de Estimación de la muestra.

Si la población es menor a cincuenta (50) individuos, la población es igual a la muestra[94].

[92] ÑAUPAS Humberto. Metodología de investigación. cuarta edición. Bogotá, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4.

[93] ÑAUPAS Humberto. Metodología de investigación. cuarta edición. Bogotá, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4.

[94] HERNANDEZ SAMPIERI, Roberto. Metodología de la Investigación. 6a. México : McGRAW W-HILL, 2014.P69

Como la población es 10 implementaciones y esta es menor a 50, la muestra será 10.

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

La recolección de datos implica: “Elaborar un plan detallado de procedimientos que nos conduzcan a reunir datos con un propósito específico.[95]

Las técnicas que vamos a utilizar en este proyecto de investigación son las Observaciones de campo y fichas de evaluación.

2.5.1 Observación

La observación es un proceso intencional de captación de las características, cualidades y propiedades de los objetos y sujetos de la realidad, a través de nuestros sentidos o con la ayuda de poderosos instrumentos que amplían su limitada capacidad.[96]

2.5.2 Instrumentos

La recolección de datos implica “elaborar un plan detallado de procedimientos que nos conduzcan a reunir información con un propósito específico.[97]

2.5.3 Ficha de Observación:

Según Arias (2012) menciona que “es aquella que además de realizarse en correspondencia con unos objetivos, utiliza una guía diseñada previamente, en la que se especifican los elementos que serán observados.[98]

[95] HERNANDEZ SAMPIERI, Roberto. Metodología de la Investigación. 6a. Mexico : McGRAW W-HILL, 2014.p198
[96] CARRASCO Díaz S. Metodología de la Investigación Científica Lima Segunda Reimpresión 2009 ASTIVERA T. Armando Metodología de la Investigación. Buenos Aires, Editorial Kapelusz (2001).
[97] ÑAUPAS Humberto. Metodología de investigación. cuarta edición. Bogotá, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4
[98] FIDIAS G., Arias. El Proyecto de Investigación: Introducción a la metodología científica. 6a Edición. Caracas, Venezuela: Editorial Episteme, C.A., 2012.p70

2.5.4 Validación

La validación de contenido se refiere: Al grado en que un instrumento refleja un dominio específico de contenido de lo que se mide. [99]

En el presente proyecto de investigación se realizará la validación de contenido por juicio de expertos.

Cuadro de Validación de expertos:

Tabla 2: Validación de instrumentos Juicio experto

ítem	Apellidos y Nombres del validador	DNI	Especialidad
1	Marín Verastegui, Wilson Ricardo	45801046	Mgtr. Gestión de tecnología de información
2	Gálvez Tapia, Orteaus Murses	16798332	Mgtr. En Ingeniería de sistemas
3	Perez Rojas , Even	43776841	Mgtr. Gestión de tecnología de información

Fuente: Elaboración Propia

2.5.5 Confiabilidad

La confiabilidad de un instrumento es:

El grado en que su aplicación repetida al mismo individuo u objeto produce resultados iguales[100]

Según sea el caso se aplicará el alfa de cronbach para determinar la confiabilidad de los instrumentos definidos.

[99] ÑAUPAS Humberto. Metodología de investigación. cuarta edición. Bogotá, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4

[100] HERNANDEZ SAMPIERI, Roberto. Metodología de la Investigación. 6a. México: McGRAW W-HILL, 2014.

$$\alpha = \left(\frac{k}{k-1}\right)\left[1 - \frac{\sum S_i^2}{S_t^2}\right]$$

Donde:

S^2_i : varianza del item i-avo
S^2_t : varianza de la escala
K : número de ítemes

"Cabe destacar que, existen instrumentos para recabar datos que por su naturaleza no ameritan el cálculo de la confiabilidad, como son: entrevistas, escalas de estimación, listas de cotejo, guías de observación, hojas de registros, inventarios, rúbricas, otros. A este tipo de instrumentos, sin embargo, debe estimarse o comprobarse su validez, a través del juicio de expertos, para establecer si los reactivos que los configuran o integran se encuentran bien redactados y miden lo que se pretende medir".[101]

2.6 Métodos de análisis de datos

Consiste en describir los datos, los valores con las puntuaciones obtenidas para cada variable.

Los métodos a aplicar serán estadística descriptiva, con el cual se recopilará y analizará los datos, con el objeto de resumir y describir los resultados a través de tablas y gráficas[102].

Según los métodos seleccionados: Estadística Descriptiva y Estadística Inferencial.

- Histogramas, Gráfico de Barras
- Prueba de Normalidad Kolmogórov-Smirnov, T-Student

[101] Coral, Yadira. Validez y confiabilidad de los instrumentos de investigación para la recolección de datos, Carabobo. Venezuela. 2008.
[102] HERNANDEZ SAMPIERI, Roberto. Metodología de la Investigación. 6a. México : McGRAW W-HILL, 2014

- Planteamiento de las Hipótesis Estadísticas Nulas ($H1_0$) y su correspondiente Hipótesis Estadística Alterna ($H1_a$).

2.7 Aspectos éticos

La presente investigación respetó la propiedad intelectual y los derechos de autor, de igual manera guarda la confidencialidad de la información de una empresa de Lima con respecto al riesgo comercial y/o estratégico. También, se tiene en consideración la confidencialidad de las personas que han participado en este proyecto, la veracidad y la confiabilidad de los resultados. Cabe mencionar que las fuentes son fidedignas y con rigor científico, y que los resultados de los mismos reflejan una evaluación objetiva.

III. RESULTADOS

3.1 Análisis Descriptivo

Tiene como objetivo fundamental, procesar, resumir y analizar un consunto de datos de la variable a estudiar, estudia un conjunto de medidas o estadígrafos mediante el cual es posible comprender la magnitud de la variable estudiada. [103]

En el estudio se empleó un sistema de información para determinar la mejora significativa de la productividad laboral, el costo laboral y en la calidad de la implementación de los productos de software de una empresa de Lima, para ello se aplicó un pre test que permitió conocer las condiciones iniciales de los indicadores del estudio, posteriormente implementado el sistema se realizó un post test para determinar la mejora u optimización del proceso de software.

1° Indicador: Productividad laboral

Tabla 3: Medias descriptivas de la productividad laboral antes y después de implementado el sistema

Variable	Media	Desv.Est.	CoefVar	Mínimo	Mediana	Máximo
Productividad Laboral (Pre)	0.038	0.014	0.3694	0.020	0.039	0.067
Productividad Laboral(Post)	0.051	0.0146	0.2891	0.033	0.047	0.077

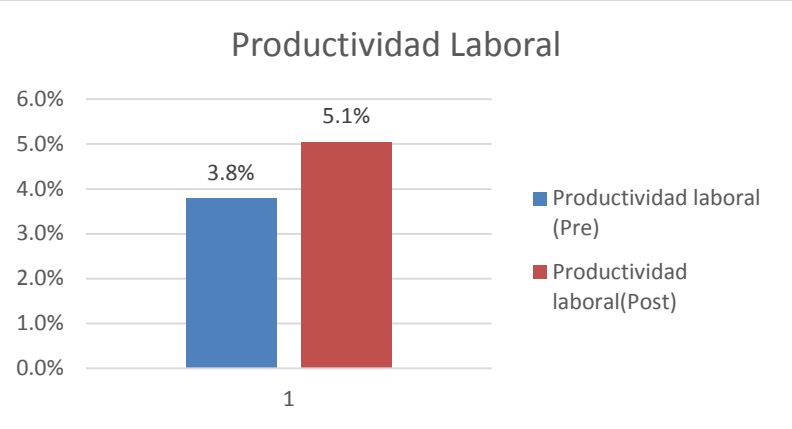
Fuente: Elaboración Propia

En el pre test se obtuvo como la media de la productividad laboral de la muestra, el valor de 3.8%, mientras que para el post test el valor fue de 5.1%; esto indica una gran diferencia antes y después de la implementación de la aplicación; así mismo, los niveles de productividad laboral mínimos fueron de 2% antes y de 3.3% después. La dispersión, en el pre test fue de 1.4 y en el post test de 1.46%, se demuestra que la variabilidad con respecto a los datos no difiere en gran medida porque están juntos, por lo tanto, la comparación de medidas se considera adecuada.

[103] ÑAUPAS Humberto. Metodología de investigación. cuarta edición. Bogotá, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4.

En la tabla 31 y 32 (ver anexo 3) se muestra los valores del Nivel de productividad laboral antes y después de la implementación del aplicativo. La medición se realizó en 10 muestras para cada prueba (pre y post test).

Figura 6: Nivel de productividad laboral antes y después de implementado la aplicación



Fuente: Elaboración propia.

2° Indicador: Costo laboral unitario

Tabla 4: Medidas descriptivas del costo laboral unitario antes y después de implementado la aplicación

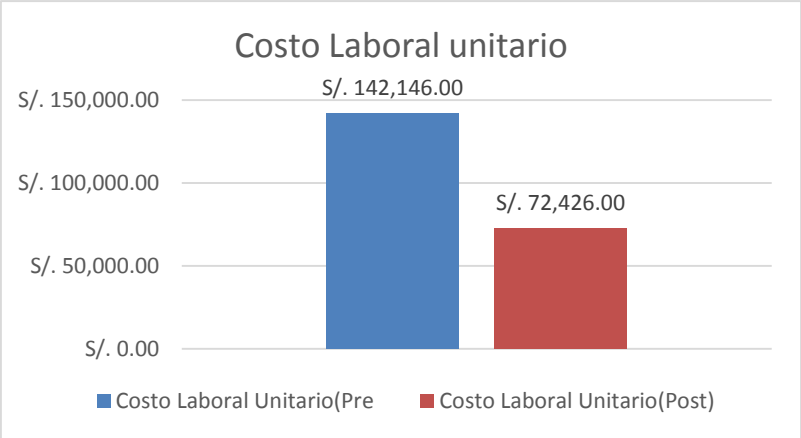
Variable	Media	Desv.Est.	CoefVar	Mínimo	Mediana	Máximo
Costo Laboral Unitario(Pre)	S/. 142,146.00	103903.2	0.731	S/. 34,800.00	S/. 91,820.00	S/. 343,000.00
Costo Laboral Unitario(Post)	S/. 72,426.00	38237.256	0.5279	S/. 27,040.00	S/. 66,200.00	S/. 141,600.00

Fuente: Elaboración propia.

En el pre test de la muestra se obtuvo como la media del costo laboral de la muestra, el valor de S/. 142,146, mientras que para el post test el valor fue de S/. 72,426; esto indica una gran diferencia antes y después de la implementación de la aplicación; así mismo, los niveles de costo laboral mínimos fueron de S/. 34,800.00 antes y de S/. 27,040.00 después. La dispersión del costo laboral unitario, en el pre test fue de S/.103903.2 y en el post test de S/.38237.25, con ello se demuestra que la variabilidad con respecto a los datos no difiere en gran medida porque están juntos; por lo tanto, la comparación de medias se considera adecuada.

En la tabla 33 y 34 (ver anexo 3) se muestra los valores del costo laboral unitario antes y después de implementado la aplicación. La medición se realizó en 10 muestras para cada prueba (pre y post test).

Figura 7: Nivel del Costo Laboral Unitario antes y después de la implementación



Fuente: Elaboración propia.

3° Indicador: Cobertura de código

Tabla 5: Medidas descriptivas del indicador de cobertura de código antes y después de implementado la aplicación

Variable	Media	Desv.Est.	CoefVar	Mínimo	Mediana	Máximo
Cobertura de Código (60 % - 100 %)(Pre)	43%	0.0678	0.1569	30%	45%	50%
Cobertura de Código (60 % - 100 %) (Post)	63%	0.1211	0.1926	40%	65%	80%

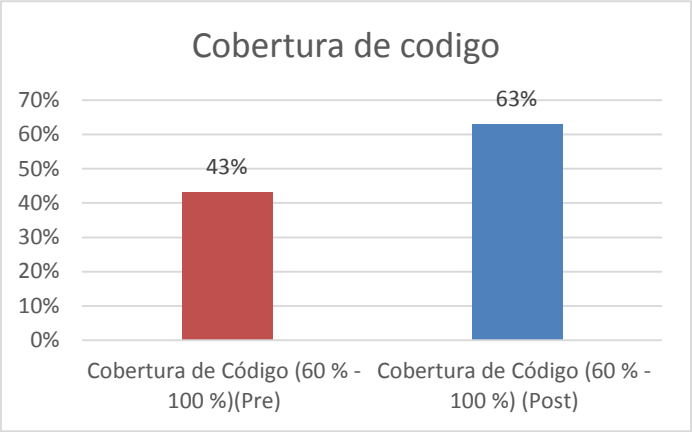
Fuente: Elaboración propia.

En el pre test de la muestra se obtuvo como media de cobertura de código, el valor de 43% de cobertura mientras para el post test el valor fue de 63%; esto indica una gran diferencia antes y después de la implementación del sistema; así mismo, la cobertura mínima antes es de 30% y de 40% después. La dispersión de cobertura de código en el pre test fue de 6.78% y en el post test de 12.11, se demuestra que la variabilidad con respecto a los datos no difiere en gran medida porque están juntos; por lo tanto, la comparación de medias se considera adecuada.

En la tabla 35 y 36 (ver anexo 3) se muestra los valores de cobertura de código antes y después de implementado la aplicación. La medición se

realizó en 10 muestras para cada prueba (pre y post test). En el caso del pre test, la cobertura de código fue entre 30% y 50% mientras que para el post test los valores aumentan entre 40% y 80%.

Figura 8: Nivel de cobertura de código del software antes y después de implementado la aplicación



Fuente: Elaboración propia.

4° Indicador: Cumplimiento de Reglas

Tabla 6: Medidas descriptivas del indicador de cumplimiento de reglas antes y después de implementado la aplicación

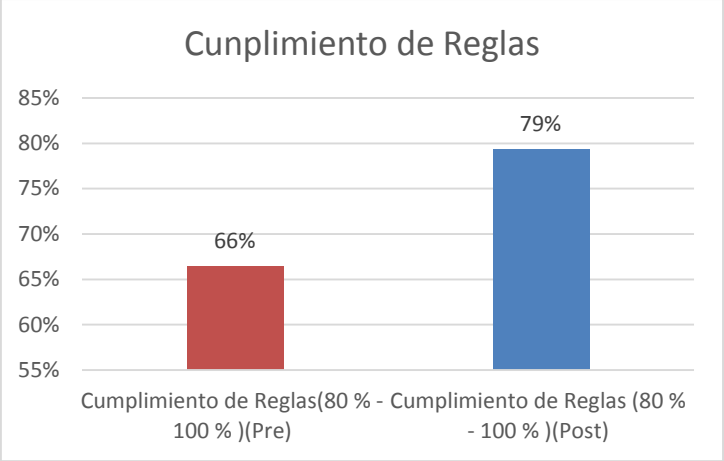
Variable	Media	Desv.Est.	CoefVar	Mínimo	Mediana	Máximo
Cumplimiento de Reglas(80 % - 100 %)(Pre)	66.4%	0.0811	0.1221	51.0%	65.0%	80.0%
Cumplimiento de Reglas (80 % - 100 %)(Post)	79%	0.0667	0.0841	68%	81%	90%

Fuente: Elaboración propia.

En el pre test de la muestra se obtuvo como media del cumplimiento de reglas de programación, el valor de 66.4% mientras para el post test el valor fue de 79%; esto indica una gran diferencia antes y después de la implementación del sistema; así mismo, el cumplimiento de reglas mínimo antes es de 51% y de 68% después. La dispersión del cumplimiento de reglas en el pre test fue de 8.11% y en el post test de 6.67%, se demuestra que la variabilidad con respecto a los datos no difiere en gran medida porque están juntos; por lo tanto, la comparación de medias se considera adecuada.

En la tabla 37 y 38 (ver anexo 3) se muestra los valores de cumplimiento de reglas de programación antes y después de implementado la aplicación. La medición se realizó en 10 muestras para cada prueba (pre y post test).

Figura 9: Nivel de cumplimiento de reglas del software antes y después de implementado la aplicación



Fuente: Elaboración propia.

5° Indicador: Índice Interdependencia entre paquetes

Tabla 7: Medidas descriptivas del indicador de interdependencia entre paquetes antes y después de implementado la aplicación

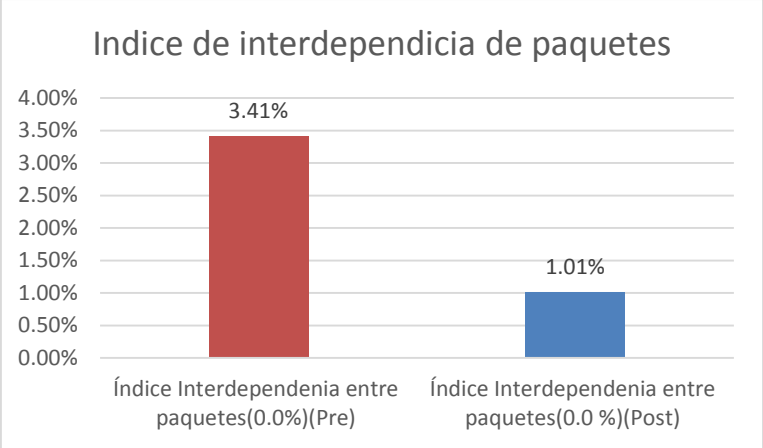
Variable	Media	Desv.Est.	CoefVar	Mínimo	Mediana	Máximo
Índice Interdependencia entre paquetes(0.0%)(Pre)	3.4%	0.0245	0.7185	0.5%	2.3%	6.5%
Índice Interdependencia entre paquetes(0.0 %)(Post)	1.0%	0.0103	1.0166	0.0%	1.0%	2.8%

Fuente: Elaboración propia.

En el pre test de la muestra se obtuvo como media de la interdependencia de paquetes, el valor de 3.4% mientras para el post test el valor fue de 1%; esto indica una gran diferencia antes y después de la implementación del sistema; así mismo, la interdependencia de paquetes mínimo antes es de 0.5% y de 0% después. La dispersión de la interdependencia de paquetes en el pre test fue de 2.45% y en el post test de 1.03%, se demuestra que la variabilidad con respecto a los datos no difiere en gran medida porque están juntos; por lo tanto, la comparación de medias se considera adecuada.

En la tabla 39 y 40 (ver anexo 3) se muestra los valores de interdependencia de paquetes antes y después de implementado la aplicación. La medición se realizó en 10 muestras para cada prueba (pre y post test).

Figura 10: Nivel de cumplimiento de interdependencia de paquetes antes y después de implementado la aplicación



Fuente: Elaboración propia.

6° Indicador: Duplicados

Tabla 8: Medidas descriptivas del indicador de duplicados antes y después de implementado la aplicación

Variable	Media	Desv.Est.	CoefVar	Mínimo	Mediana	Máximo
Duplicados (0 % – 10 %)(Pre)	15%	0.0381	0.2628	10%	14%	20%
Duplicados (0 % – 10 %) (Post)	9%	0.0218	0.2453	5%	10%	12%

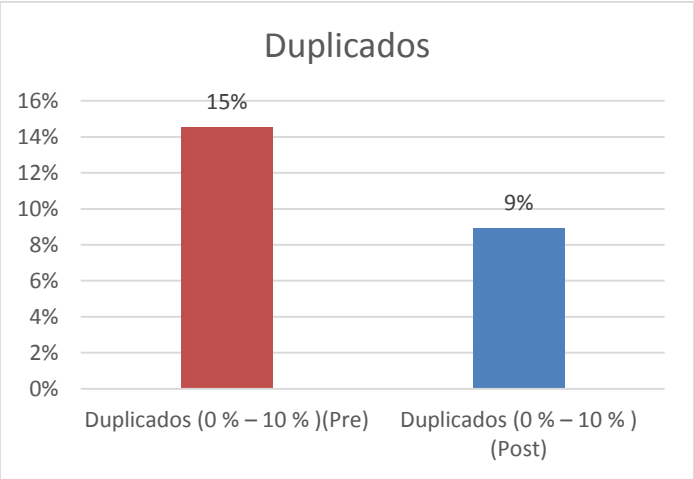
Fuente: Elaboración propia.

En el pre test de la muestra se obtuvo como media de códigos duplicados, el valor de 15% mientras para el post test el valor fue de 9%; esto indica una gran diferencia antes y después de la implementación del sistema; así mismo, la cantidad del código duplicado mínimos antes es de 10% y de 5% después. La dispersión de la cantidad del código duplicado en el pre test fue de 3.81% y en el post test de 2.18%, se demuestra que la variabilidad con respecto a los datos no difiere en gran medida porque están juntos; por lo tanto, la comparación de medias se considera adecuada.

En la tabla 41 y 42 (ver anexo 3) se muestra los valores del código duplicado antes y después de implementado la aplicación. La medición se realizó en

10 muestras para cada prueba (pre y post test). En el caso del pre test, el código duplicado, fue entre 10% y 20% mientras que para el post test los valores aumentan entre 5% y 12%.

Figura 11: Nivel de códigos duplicados antes y después de implementado la aplicación



Fuente: Elaboración propia.

7° Indicador: Complejidad del Método

Tabla 9: Medidas descriptivas del indicador de complejidad de metodo antes y después de implementado la aplicación

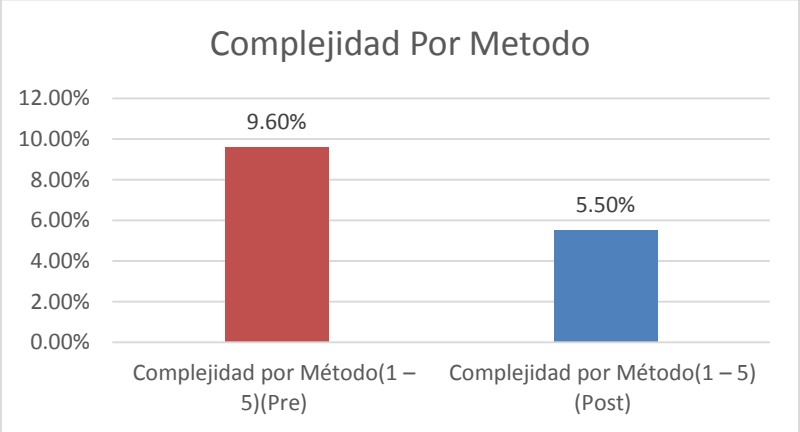
Variable	Media	Desv.Est.	CoefVar	Mínimo	Mediana	Máximo
Complejidad por Método(1 – 5)(Pre)	10%	0.0448	0.4667	5%	8%	20%
Complejidad por Método(1 – 5) (Post)	6%	0.0127	0.2308	4%	5%	8%

Fuente: Elaboración propia.

En el pre test de la muestra se obtuvo como media de la complejidad por método, el valor de 10% mientras para el post test el valor fue de 6%; esto indica una gran diferencia antes y después de la implementación del sistema; así mismo, el nivel de complejidad del método mínimos antes es de 5% y de 4% después. La dispersión de la complejidad por método en el pre test fue de 4.48% y en el post test de 1.27%, se demuestra que la variabilidad con respecto a los datos no difiere en gran medida porque están juntos; por lo tanto, la comparación de medias se considera adecuada.

En la tabla 43 y 44 (ver anexo 3) se muestra los valores de complejidad por método antes y después de implementado la aplicación. La medición se realizó en 10 muestras para cada prueba (pre y post test). En el caso del pre test, la complejidad por método, fue entre 5% y 20% mientras que para el post test los valores aumentan entre 4% y 8%.

Figura 12: Nivel de complejidad por método antes y después de implementado la aplicación



Fuente: Elaboración propia.

8° Indicador: LCOM4

Tabla 10: Medidas descriptivas del indicador LCOM4 antes y después de implementado la aplicación

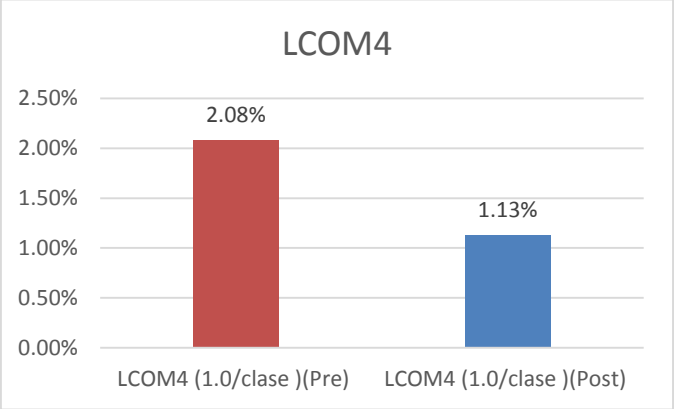
Variable	Media	Desv.Est.	CoefVar	Mínimo	Mediana	Máximo
LCOM4 (1.0/clase)(Pre)	2.1%	0.0071	0.3413	1.1%	2.1%	3.0%
LCOM4 (1.0/clase)(Post)	1.1%	0.0018	0.1618	1.0%	1.0%	1.5%

Fuente: Elaboración propia.

En el pre test de la muestra se obtuvo como media de LCOM4, el valor de 2.1% mientras para el post test el valor fue de 1.1%; esto indica una gran diferencia antes y después de la implementación del sistema; así mismo, valor de LCOM4 antes fue de 1.1% y de 1.0% después. La dispersión del LCOM4 en el pre test fue de 0.71% y en el post test de 0.18%, se demuestra que la variabilidad con respecto a los datos no difiere en gran medida porque están juntos; por lo tanto, la comparación de medias se considera adecuada.

En la tabla 45 y 46 (ver anexo 3) se muestra los valores del LCOM4 antes y después de implementado la aplicación. La medición se realizó en 10 muestras para cada prueba (pre y post test). En el caso del pre test, el LCOM4, fue entre 1.1% y 3% mientras que para el post test los valores aumentan entre 1% y 1.5%.

Figura 13: Nivel de LCOM4 antes y después de implementado la aplicación



Fuente: Elaboración propia.

9° Indicador: Comentarios

Tabla 11: Medidas descriptivas de los comentarios en el codigo antes y después de implementado la aplicación

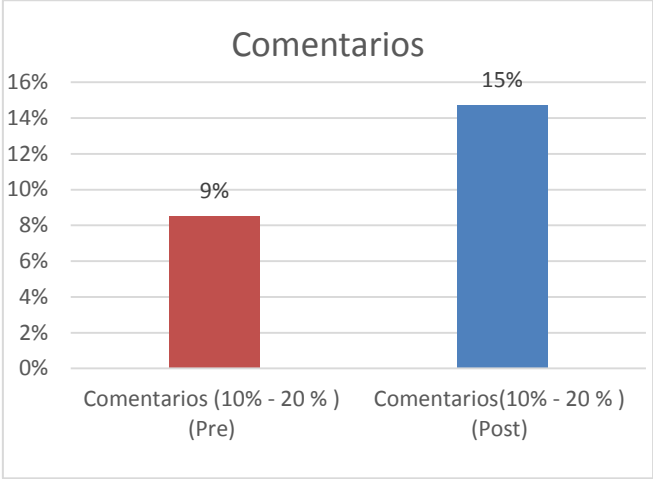
Variable	Media	Desv.Est.	CoefVar	Mínimo	Mediana	Máximo
Comentarios (10% - 20 %) (Pre)	8.5%	0.0222	0.2612	5.0%	9.0%	11.0%
Comentarios(10% - 20 %) (Post)	15%	0.02406	0.1637	11%	15%	18%

Fuente: Elaboración propia.

En el pre test de la muestra se obtuvo como media de los comentarios en el código, el valor de 8.5% mientras para el post test el valor fue de 15%; esto indica una gran diferencia antes y después de la implementación del sistema; así mismo, valor del porcentaje de comentarios antes fue de 5% y de 11% después.La dispersión del porcentaje de comentarios del código en el pre test fue de 2.22% y en el post test de 2.4%, se demuestra que la variabilidad con respecto a los datos no difiere en gran medida porque están juntos; por lo tanto, la comparación de medias se considera adecuada.

En la tabla 47 y 48 (ver anexo 3) se muestra los valores del porcentaje de comentarios del código antes y después de implementado la aplicación. La medición se realizó en 10 muestras para cada prueba (pre y post test).

Figura 14: Nivel de comentarios de código antes y después de implementado la aplicación



Fuente: Elaboración propia.

3.2 Análisis Inferencial

La estadística inferencial es aquella que ayuda al investigador a encontrar la significatividad de sus resultados, esta estadística compara los datos de dos muestras para poder determinar las posibles diferencias[104]. Con el objetivo de seleccionar la prueba de hipótesis; los datos fueron sometidos a la comprobación de su distribución, específicamente si los datos contaban con distribución normal.

Se elige a Kolmogórov-Smirnov debido a la presencia de datos continuos y la ausencia de categorías.

[104] ÑAUPAS Humberto. Metodología de investigación. cuarta edición. Bogota, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4.

1° Indicador: Productividad Laboral

Con el objetivo de seleccionar la prueba de hipótesis; los datos fueron sometidos a la comprobación de su distribución, para validar tienen un con distribución normal.

Tabla 12: Prueba de normalidad de la productividad laboral antes de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra

		Productividad
N		10
Parámetros normales[a,b]	Media	.037800
	Desviación estándar	.0141091
Máximas diferencias extremas	Absoluta	.156
	Positivo	.156
	Negativo	-.106
Estadístico de prueba		.156
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.

b. Se calcula a partir de datos.

c. Corrección de significación de Lilliefors.

d. Esto es un límite inferior de la significación verdadera.

Fuente: Elaboración propia.

Tabla 13: Prueba de normalidad la productividad laboral después de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra

		Productividad
N		10
Parámetros normales[a,b]	Media	.050500
	Desviación estándar	.0148043
Máximas diferencias extremas	Absoluta	.194
	Positivo	.194
	Negativo	-.119
Estadístico de prueba		.194
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.

b. Se calcula a partir de datos.

c. Corrección de significación de Lilliefors.

d. Esto es un límite inferior de la significación verdadera.

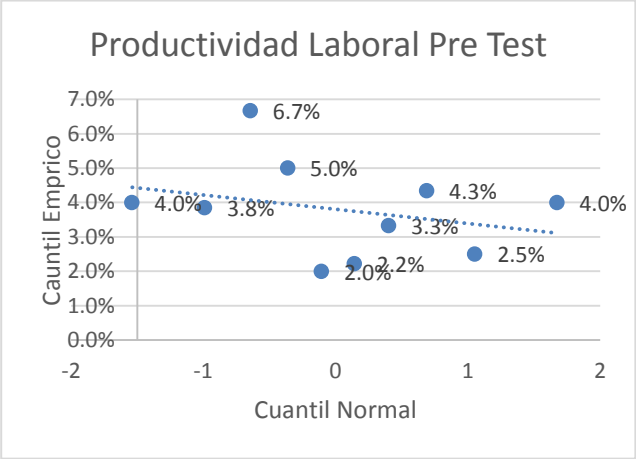
Fuente: Elaboración propia.

Pre-Test: El valor Sig. = 0.2 > P-Valor = 0.05, por lo tanto, esta muestra tiene el comportamiento Normal.

Post-Test: El valor Sig. = 0.2 > P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.

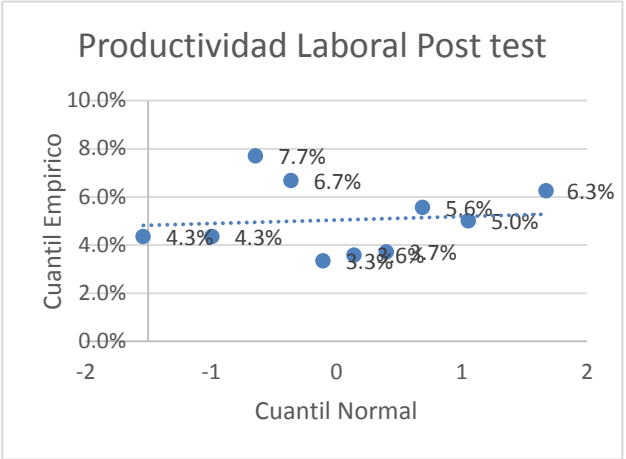
Las Figuras 16 y 17, comprueban la prueba de normalidad, con la tendencia en la concentración de puntos en torno a la recta. Lo que confirma la distribución normal de los datos de la muestra.

Figura 15: Prueba de Normalidad del Nivel de productividad laboral antes de implementado la aplicación



Fuente: Elaboración propia.

Figura 16: Prueba de Normalidad del Nivel de productividad laboral después de implementado la aplicación



Fuente: Elaboración propia.

2° Indicador: Costo laboral unitario

Con el objetivo de seleccionar la prueba de hipótesis; los datos fueron sometidos a la comprobación de su distribución, para validar tienen un con distribución normal.

Tabla 14: Prueba de normalidad del costo laboral unitario antes y después de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Costo
N		10
Parámetros normales[a,b]	Media	142146.00
	Desviación estándar	103903.197
Máximas diferencias extremas	Absoluta	.276
	Positivo	.276
	Negativo	-.151
Estadístico de prueba		.276
Sig. asintótica (bilateral)		,290[c]

a. La distribución de prueba es normal.
b. Se calcula a partir de datos.
c. Corrección de significación de Lilliefors.

Fuente: Elaboración propia

Tabla 15: Prueba de normalidad del costo laboral unitario antes y después de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Productividad
N		10
Parámetros normales[a,b]	Media	72426.00
	Desviación estándar	38237.256
Máximas diferencias extremas	Absoluta	.183
	Positivo	.183
	Negativo	-.120
Estadístico de prueba		.183
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.
b. Se calcula a partir de datos.
c. Corrección de significación de Lilliefors.

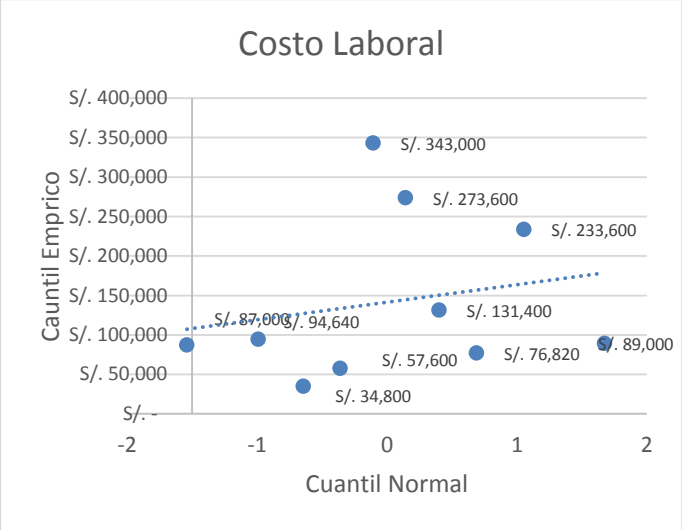
Fuente: Elaboración propia

Pre-Test: El valor Sig. = 0.517 > P-Valor = 0.05, por lo tanto, esta muestra tiene el comportamiento Normal.

Post-Test: El valor Sig. = 0.29 > P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.

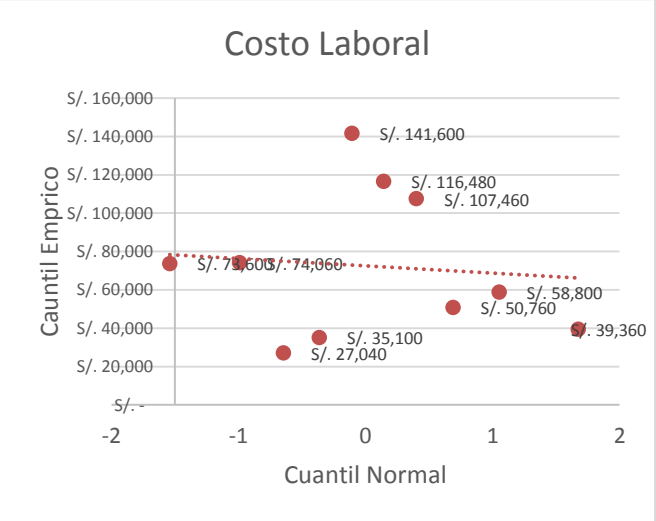
Las Figuras 18 y 19, se comprueban la prueba de normalidad, con la tendencia en la concentración de puntos en torno a la recta. Lo que confirma la distribución normal de los datos de la muestra.

Figura 17: Prueba de Normalidad del costo laboral unitario antes de implementado el sistema



Fuente: Elaboración propia.

Figura 18: Prueba de Normalidad del costo laboral unitario antes y después de implementado la aplicación.



Fuente: Elaboración propia

3° Indicador: Cobertura de Código

Con el objetivo de seleccionar la prueba de hipótesis; los datos fueron sometidos a la comprobación de su distribución, específicamente si los datos de cobertura de código contaban con distribución normal.

Tabla 16: Prueba de normalidad del indicador de cobertura de código antes de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Cobertura
N		10
Parámetros normales[a,b]	Media	43.200
	Desviación estándar	6.779
Máximas diferencias extremas	Absoluta	.205
	Positivo	.158
	Negativo	-.205
Estadístico de prueba		.205
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.

b. Se calcula a partir de datos.

c. Corrección de significación de Lilliefors.

d. Esto es un límite inferior de la significación verdadera.

Fuente: Elaboración propia

Tabla 17: Prueba de normalidad del indicador de cobertura de código después de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Cobertura
N		10
Parámetros normales[a,b]	Media	62.900
	Desviación estándar	12.115
Máximas diferencias extremas	Absoluta	.205
	Positivo	.152
	Negativo	-.205
Estadístico de prueba		.205
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.

b. Se calcula a partir de datos.

c. Corrección de significación de Lilliefors.

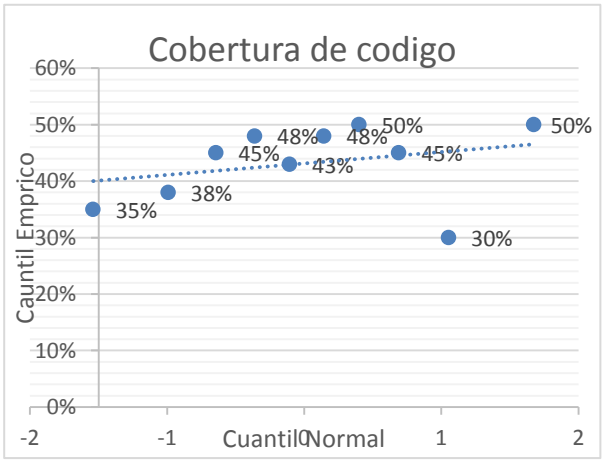
d. Esto es un límite inferior de la significación verdadera.

Fuente: Elaboración propia

Pre-Test: El valor Sig. = 0.2 > P-Valor = 0.05, por lo tanto, esta muestra tiene el comportamiento Normal.

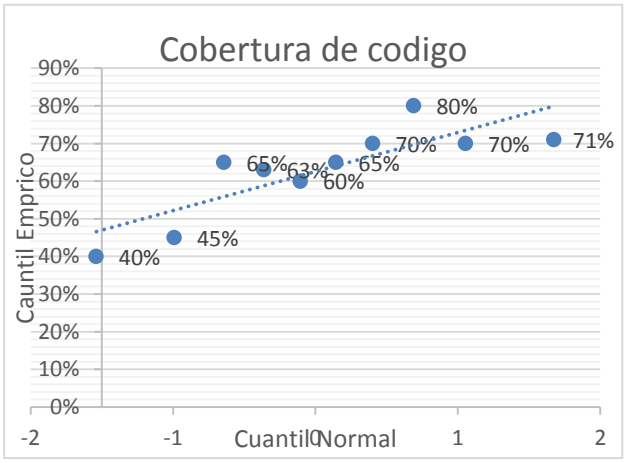
Post-Test: El valor Sig. = 0.2 > P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.

Figura 19: Prueba de Normalidad del indicador de cobertura de código antes de implementado el sistema



Fuente: Elaboración propia.

Figura 20: Prueba de Normalidad del indicador de cobertura de código después de implementado la aplicación



Fuente: Elaboración propia.

Las Figuras 18 y 19, se comprueban la prueba de normalidad, con la tendencia en la concentración de puntos en torno a la recta. Lo que confirma la distribución normal de los datos de la muestra.

4° Indicador: Cumplimiento de Reglas

Con el objetivo de seleccionar la prueba de hipótesis; los datos fueron sometidos a la comprobación de su distribución, específicamente si los datos de cobertura de código contaban con distribución normal.

Tabla 18: Prueba de normalidad de los indicador de cumplimiento de reglas antes de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Reglas
N		10
Parámetros normales[a,b]	Media	66.400
	Desviación estándar	8.113
Máximas diferencias extremas	Absoluta	.169
	Positivo	.169
	Negativo	-.138
Estadístico de prueba		.169
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.
b. Se calcula a partir de datos.

Fuente: Elaboración propia

Tabla 19: Prueba de normalidad del indicador de cumplimiento de reglas después de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Reglas
N		10
Parámetros normales[a,b]	Media	79.300
	Desviación estándar	6.667
Máximas diferencias extremas	Absoluta	.242
	Positivo	.118
	Negativo	-.242
Estadístico de prueba		.242
Sig. asintótica (bilateral)		,101[c]

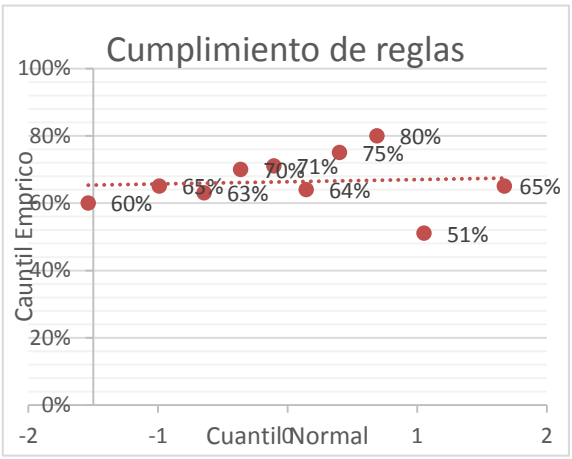
a. La distribución de prueba es normal.
b. Se calcula a partir de datos.

Fuente: Elaboración propia.

Pre-Test: El valor Sig. = 0.2 > P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.
Post-Test: El valor Sig. = 0.1 > P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.

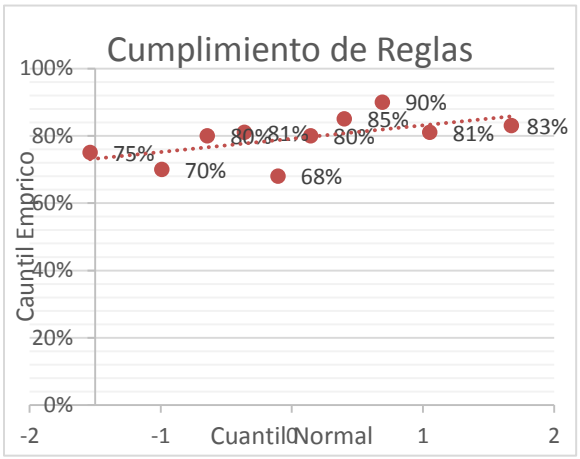
Las Figuras 22 y 23, se comprueban la prueba de normalidad, con la tendencia en la concentración de puntos en torno a la recta. Lo que confirma la distribución normal de los datos de la muestra.

Figura 21: Prueba de Normalidad del indicador de cumplimiento de reglas antes de implementado el sistema



Fuente: Elaboración propia

Figura 22: Prueba de Normalidad del indicador de cumplimiento de reglas después de implementado la aplicación



Fuente: Elaboración propia.

5° Indicador: Índice Interdependencia entre paquetes

Con el objetivo de seleccionar la prueba de hipótesis; los datos fueron sometidos a la comprobación de su distribución, específicamente si los datos de cobertura de código contaban con distribución normal.

Tabla 20: Prueba de normalidad del indicador del índice de interdependencia entre paquetes antes de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Interdependencia
N		10
Parámetros normales[a,b]	Media	3.410
	Desviación estándar	2.452
Máximas diferencias extremas	Absoluta	.245
	Positivo	.245
	Negativo	-.235
Estadístico de prueba		.245
Sig. asintótica (bilateral)		,091[c]

a. La distribución de prueba es normal.
b. Se calcula a partir de datos.
c. Corrección de significación de Lilliefors.
d. Esto es un límite inferior de la significación verdadera.

Fuente: Elaboración propia.

Tabla 21: Prueba de normalidad del indicador del índice de interdependencia de paquetes después de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Interdependencia
N		10
Parámetros normales[a,b]	Media	1.010
	Desviación estándar	1.027
Máximas diferencias extremas	Absoluta	.237
	Positivo	.237
	Negativo	-.163
Estadístico de prueba		.237
Sig. asintótica (bilateral)		,117[c]

a. La distribución de prueba es normal.
b. Se calcula a partir de datos.
c. Corrección de significación de Lilliefors.
d. Esto es un límite inferior de la significación verdadera.

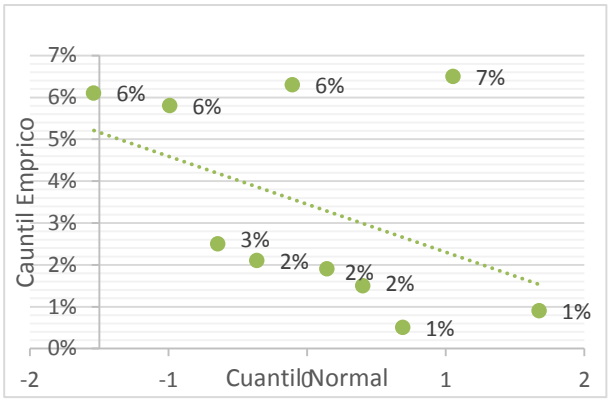
Fuente: Elaboración propia.

Pre-Test: El valor Sig. = 0.09> P-Valor = 0.05, por lo tanto, esta muestra tiene el comportamiento Normal.

Post-Test: El valor Sig. = 0.11 > P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.

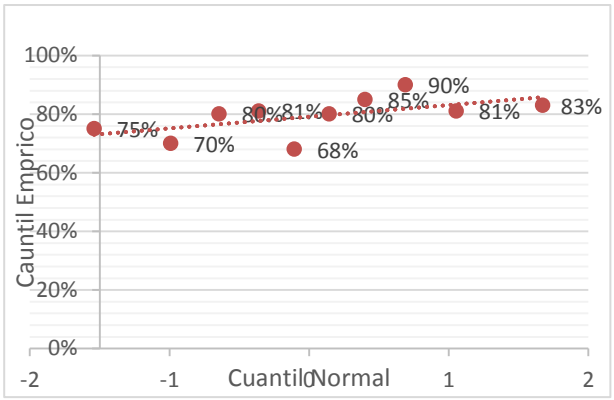
Las Figuras 24 y 25, se comprueban la prueba de normalidad, con la tendencia en la concentración de puntos en torno a la recta. Lo que confirma la distribución normal de los datos de la muestra.

Figura 23: Prueba de Normalidad del indicador índice de interdependencia de paquetes antes de implementado el sistema



Fuente: Elaboración propia

Figura 24: Prueba de Normalidad del indicador índice de interdependencia de paquetes después de implementado la aplicación



Fuente: Elaboración propia.

6° Indicador: Duplicados

Con el objetivo de seleccionar la prueba de hipótesis; los datos fueron sometidos a la comprobación de su distribución, específicamente si los datos de cobertura de código contaban con distribución normal.

Tabla 22: Prueba de normalidad del indicador códigos duplicados antes de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Duplicados
N		10
Parámetros normales[a,b]	Media	14.500
	Desviación estándar	3.808
Máximas diferencias extremas	Absoluta	.153
	Positivo	.153
	Negativo	-.144
Estadístico de prueba		.153
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.

b. Se calcula a partir de datos.

Fuente: Elaboración propia

Tabla 23: Prueba de normalidad del indicador de códigos duplicados después de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Duplicados
N		10
Parámetros normales[a,b]	Media	8.900
	Desviación estándar	2.183
Máximas diferencias extremas	Absoluta	.193
	Positivo	.108
	Negativo	-.193
Estadístico de prueba		.193
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.

b. Se calcula a partir de datos.

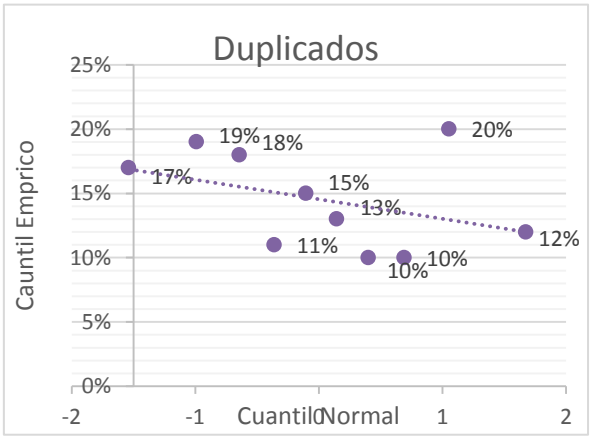
Fuente: Elaboración propia.

Pre-Test: El valor Sig. = 0.2> P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.

Post-Test: El valor Sig. = 0.2 > P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.

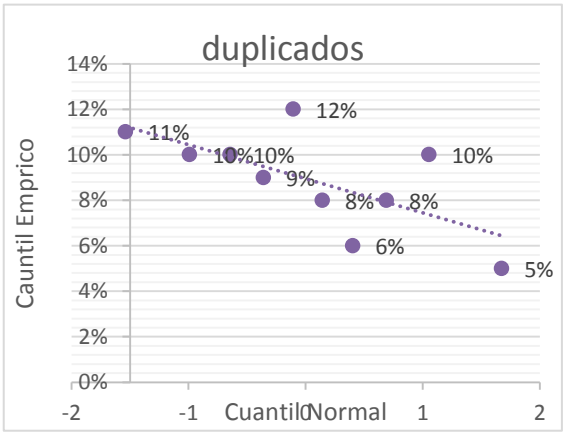
Las Figuras 26 y 27, se comprueban la prueba de normalidad, con la tendencia en la concentración de puntos en torno a la recta. Lo que confirma la distribución normal de los datos de la muestra.

Figura 25: Prueba de Normalidad del indicador de códigos duplicados antes de implementado el sistema



Fuente: Elaboración propia

Figura 26: Prueba de Normalidad del indicador de códigos duplicados después de implementado la aplicación



Fuente: Elaboración propia

7° Indicador: Complejidad por Método

Con el objetivo de seleccionar la prueba de hipótesis; los datos fueron sometidos a la comprobación de su distribución, específicamente si los datos de cobertura de código contaban con distribución normal.

Tabla 24: Prueba de normalidad del indicador de complejidad por método antes de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Complejidad
N		10
Parámetros normales[a,b]	Media	9.600
	Desviación estándar	4.477
Máximas diferencias extremas	Absoluta	.219
	Positivo	.219
	Negativo	-.152
Estadístico de prueba		.219
Sig. asintótica (bilateral)		,190[c]

a. La distribución de prueba es normal.
b. Se calcula a partir de datos.
c. Corrección de significación de Lilliefors.

Fuente: Elaboración propia.

Tabla 25: Prueba de normalidad del indicador de complejidad por método después de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Complejidad
N		10
Parámetros normales[a,b]	Media	5.500
	Desviación estándar	1.269
Máximas diferencias extremas	Absoluta	.253
	Positivo	.253
	Negativo	-.147
Estadístico de prueba		.253
Sig. asintótica (bilateral)		,069[c]

a. La distribución de prueba es normal.
b. Se calcula a partir de datos.
c. Corrección de significación de Lilliefors.

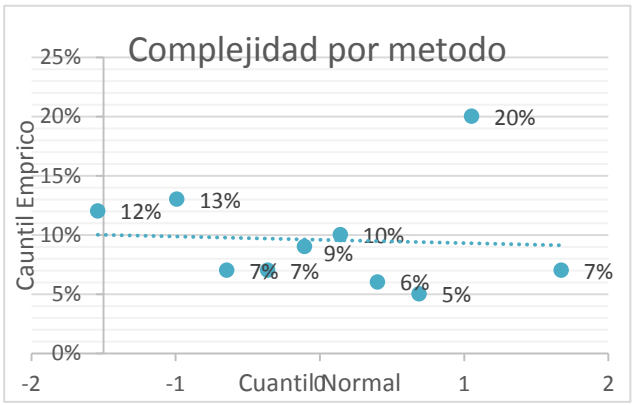
Fuente: Elaboración propia

Pre-Test: El valor Sig. = 0.19> P-Valor = 0.05, por lo tanto, esta muestra tiene el comportamiento Normal.

Post-Test: El valor Sig. = 0.06 > P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.

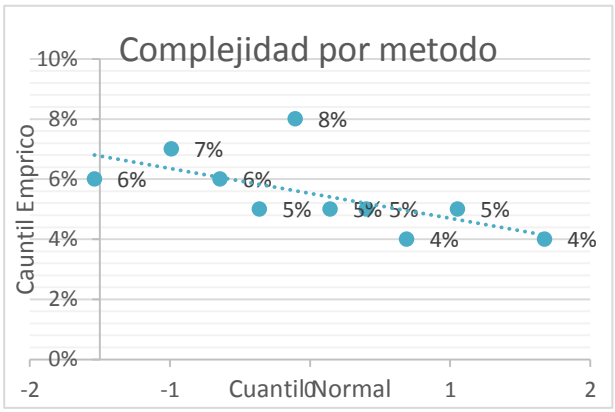
Las Figuras 28 y 29, se comprueban la prueba de normalidad, con la tendencia en la concentración de puntos en torno a la recta. Lo que confirma la distribución normal de los datos de la muestra.

Figura 27: Prueba de Normalidad del indicador de complejidad por método antes de implementado el sistema



Fuente: Elaboración propia

Figura 28: Prueba de Normalidad del indicador de complejidad por método después de implementado la aplicación



Fuente: Elaboración propia

8° Indicador: LCOM4

Con el objetivo de seleccionar la prueba de hipótesis; los datos fueron sometidos a la comprobación de su distribución, específicamente si los datos de cobertura de código contaban con distribución normal.

Tabla 26: Prueba de normalidad del indicador LCOM4 antes de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		LCOM4
N		10
Parámetros normales[a,b]	Media	2.090
	Desviación estándar	0.709
Máximas diferencias extremas	Absoluta	.167
	Positivo	.167
	Negativo	-.142
Estadístico de prueba		.167
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.
b. Se calcula a partir de datos.
c. Corrección de significación de Lilliefors.

Fuente: Elaboración propia.

Tabla 27: Prueba de normalidad del indicador LCOM4 después de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		LCOM4
N		10
Parámetros normales[a,b]	Media	1.130
	Desviación estándar	0.183
Máximas diferencias extremas	Absoluta	.361
	Positivo	.361
	Negativo	-.239
Estadístico de prueba		.361
Sig. asintótica (bilateral)		,0061[c]

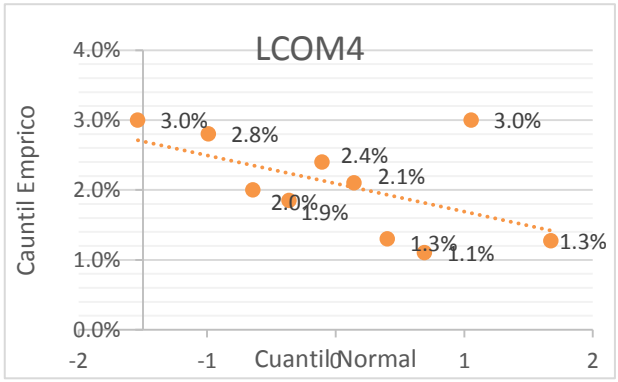
a. La distribución de prueba es normal.
b. Se calcula a partir de datos.
c. Corrección de significación de Lilliefors.

Fuente: Elaboración propia.

Pre-Test: El valor Sig. = 0.2> P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.
Post-Test: El valor Sig. = 0.061 > P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.

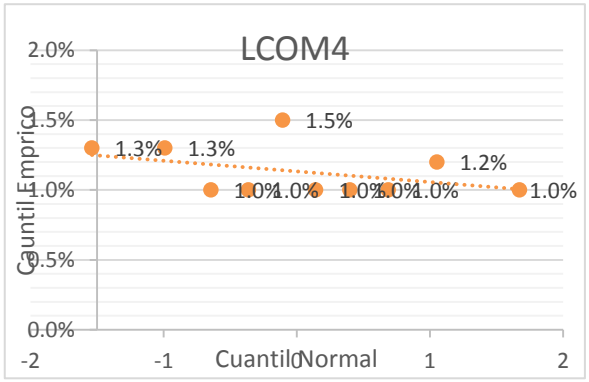
Las Figuras 30 y 31, se comprueban la prueba de normalidad, con la tendencia en la concentración de puntos en torno a la recta. Lo que confirma la distribución normal de los datos de la muestra.

Figura 29: Prueba de Normalidad del indicador LCOM4 antes de implementado el sistema



Fuente: Elaboración propia

Figura 30: Prueba de Normalidad del indicador LCOM4 después de implementado la aplicación



Fuente: Elaboración propia

9° Indicador: Comentarios

Con el objetivo de seleccionar la prueba de hipótesis; los datos fueron sometidos a la comprobación de su distribución, específicamente si los datos de cobertura de código contaban con distribución normal.

Tabla 28: Prueba de normalidad del indicador comentarios del código antes de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Comentarios
N		10
Parámetros normales[a,b]	Media	8.500
	Desviación estándar	2.224
Máximas diferencias extremas	Absoluta	.189
	Positivo	.142
	Negativo	-.189
Estadístico de prueba		.189
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.

b. Se calcula a partir de datos.

c. Corrección de significación de Lilliefors.

d. Esto es un límite inferior de la significación verdadera.

Fuente: Elaboración propia.

Tabla 29: Prueba de normalidad del indicador de comentarios del código después de implementado la aplicación

Prueba de Kolmogorov-Smirnov para una muestra		
		Comentarios
N		10
Parámetros normales[a,b]	Media	14.700
	Desviación estándar	2.406
Máximas diferencias extremas	Absoluta	.150
	Positivo	.150
	Negativo	-.130
Estadístico de prueba		.150
Sig. asintótica (bilateral)		,200[c,d]

a. La distribución de prueba es normal.

b. Se calcula a partir de datos.

c. Corrección de significación de Lilliefors.

d. Esto es un límite inferior de la significación verdadera.

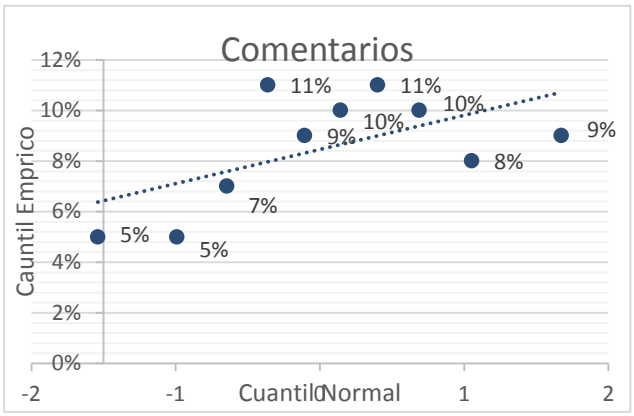
Fuente: Elaboración propia.

Pre-Test: El valor Sig. = 0.2> P-Valor = 0.05, por lo tanto, esta muestra tiene el comportamiento Normal.

Post-Test: El valor Sig. = 0.2 > P-Valor = 0.05, por lo tanto esta muestra tiene el comportamiento Normal.

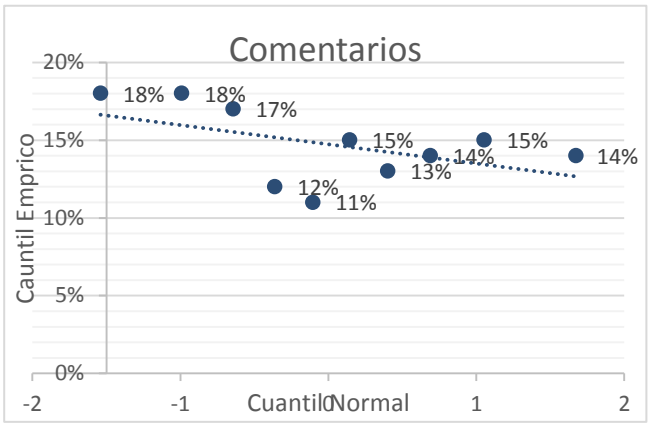
Las Figuras 32 y 33, se comprueban la prueba de normalidad, con la tendencia en la concentración de puntos en torno a la recta. Lo que confirma la distribución normal de los datos de la muestra.

Figura 31: Prueba de Normalidad del indicador de comentarios del código antes de implementado el sistema



Fuente: Elaboración propia.

Figura 32: Prueba de Normalidad del indicador comentarios del codigo después de implementado la aplicación



Fuente: Elaboración propia.

3.3 Prueba de Hipótesis

Como la prueba de normalidad, dio como resultado que los datos de los indicadores tienen distribución normal, los valores del pre test y post test fueron comparados utilizando la prueba de T-Student con una significancia del 5%.

Para la constatación de la hipótesis se realiza la prueba T de Student para la diferencia de medias muéstrales, ya que las muestras son menores a 10.

Los contrastes de hipótesis son capaces de responder a preguntas concretas que nos podemos formular sobre los parámetros poblacionales de interés, por ejemplo: ¿La cantidad media diaria de sal ingerida por hipertensos es mayor que la que ingieren las personas con presión arterial normal?, ¿La temperatura corporal de los pacientes que han sufrido cierta infección bacteriana es superior a los 36.7 grados centígrados?, ¿La proporción de personas diabéticas con problemas de vista es superior a la de la población general?. Resulta evidente que un mecanismo capaz de dar respuesta a cuestiones como las anteriores sería una herramienta muy valiosa, en consecuencia los contrastes o tests de hipótesis son una de las

utilidades más valoradas y extendidas en la realización de estudios estadísticos[105]

A. Hipótesis de Investigación 1

Hipótesis Específica

HE: La implementación del sistema de encriptación incrementa significativamente la productividad laboral del proceso de desarrollo de software de una empresa de Lima.

Indicador: Nivel de productividad laboral.

Validación de la Hipótesis

Se toma la T de Student para la diferencia de medias, ya que las muestras son menores a 30. Obteniéndose los siguientes resultados:

Definición de Variables:

MPL_a: Media para el nivel de productividad laboral sin el aplicativo.

MPL_d: Media para el nivel de productividad laboral con el aplicativo.

Hipótesis Nula:

H_0: La implementación del sistema de encriptación no incrementa significativamente la productividad laboral del proceso de desarrollo de software de una empresa de Lima.

H_0: $MPL_a >= MPL_d$

Hipótesis Alterna:

H_a: La implementación del sistema de encriptación incrementa significativamente la productividad laboral del proceso de desarrollo de software de una empresa de Lima.

H_a: $MPL_a < MPL_d$

[105] UNIVERSIDAD CARDENAL HERRERA. Inferencia estadística (intervalos de confianza y p-alor).Comparación de dos poblaciones (test t de comparación de medias, comparación de dos proporciones, comparación de dos varianzas). P8

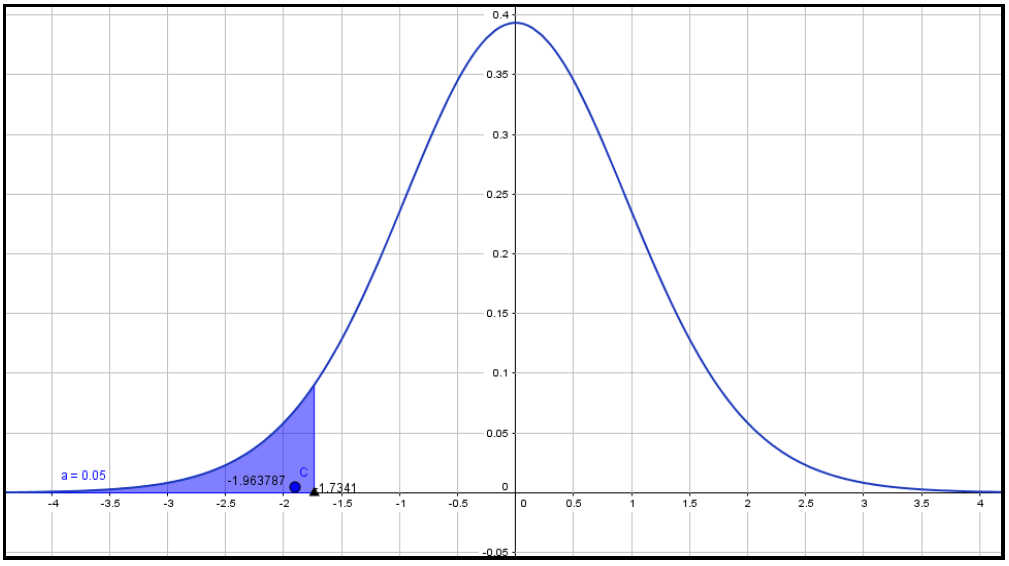
Con el gl (grados de Libertad) de 18 y α (porcentaje de error) de 0.05, se tiene el valor del estadístico t = 1.734, con ello se procede a calcular el t de la diferencia de medias:

ítem	X	S	N
1	0.038	0.014	10
2	0.051	0.0146	10

$t = \frac{(\overline{X}_1 - \bar{X}_2) - (u_1 - u_2)}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$	$t = \frac{(0.038 - 0.051) - 0}{\sqrt{\left(\frac{0.014^2}{10} + \frac{0.0146^2}{10}\right)}} = -1.964$

$u_1 - u_2 = 0, debido\ a\ que\ esa\ diferencia\ encaja\ en\ el\ origen\ de\ coordenadas.$

Figura 33: Gráfica de regla de decisión producción por trabajador



Fuente: Elaboración propia.

Conclusión:

Debido a que el valor calcula para el estadístico t = - 1.964 cae en el área de rechazo de la hipótesis nula, entonces se rechaza la Hipótesis Nula y **se acepta la Hipótesis Alterna.** Así, se da evidencia de que con un 95% de confiabilidad, el Indicador 1 se incrementa favorable mente al tener sistema implementado y esto se corrobora con la subida porcentual de este indicador **(1.3%)**

B. Hipótesis de Investigación 2

Hipótesis Específica

HE: La implementación del sistema de encriptación reduce de manera significativa el costo laboral unitario del proceso de desarrollo de software de una empresa de Lima.

Indicador: Costo laboral unitario

Validación de la Hipótesis

Se toma la T de Student para la diferencia de medias, ya que las muestras son menores a 30. Obteniéndose los siguientes resultados:

Definición de Variables:

$MCLU_a$: Media para el costo laboral unitario sin la aplicación.

$MCLU_d$: Media para el costo laboral unitario con la aplicación.

Hipótesis Nula:

H_0: La implementación del sistema de encriptación no reduce de manera significativa el costo laboral unitario del proceso de desarrollo de software de una empresa de Lima.

H_0: $\mathbf{MCLU_a < MCLU_d}$

Hipótesis Alterna:

H_a: La implementación del sistema de encriptación reduce, de manera significativa, el costo laboral unitario del proceso de desarrollo de software de una empresa de Lima

La media para el costo laboral unitario sin la aplicación es mayor a la Media con la aplicación.

H_a: $\mathbf{MCLU_a > MCLU_d}$

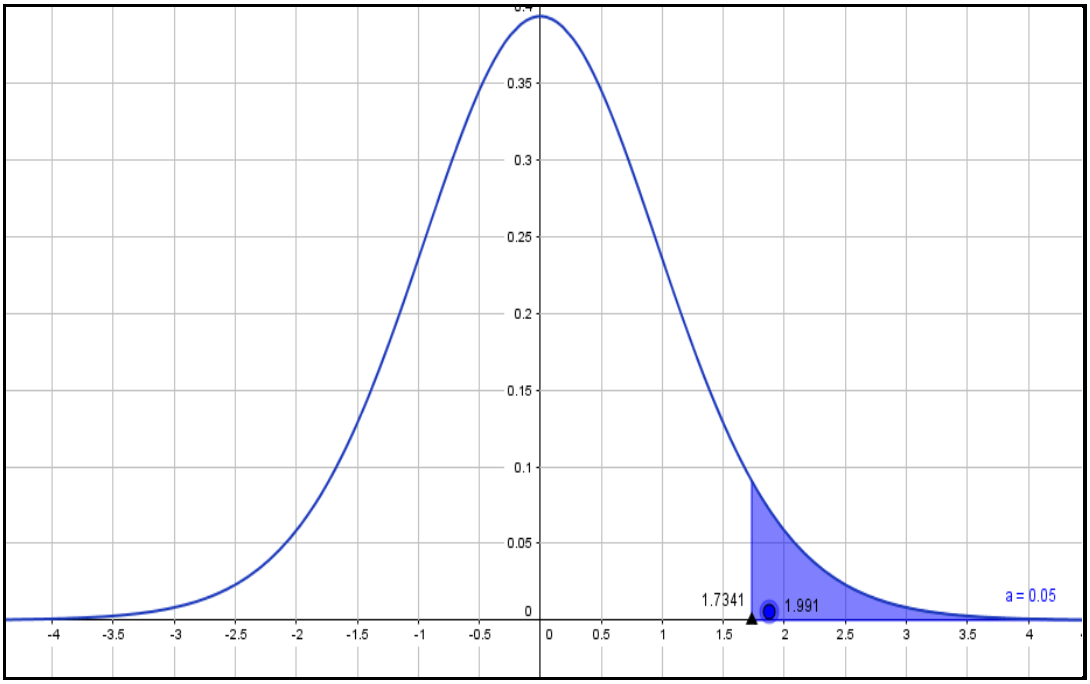
Con el gl (grados de Libertad) de 18 y α (porcentaje de error) de 0.05, se tiene el valor del estadístico t = 1.734, con ello se procede a calcular el t de la diferencia de medias:

ítem	X	S	N
1	142146	103903.2	10
2	72426	38237.256	10

$t = \frac{(\overline{X_1} - \bar{X}_2) - (u_1 - u_2)}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$	$t = \frac{(142146 - 103903) - 0}{\sqrt{\left(\frac{103903.2^2}{10} + \frac{38237.256^2}{10}\right)}} = 1.991$

$u_1 - u_2 = 0, debido\ a\ que\ esa\ diferencia\ encaja\ en\ el\ origen\ de\ coordenadas.$

Figura 34: Gráfica de regla de decisión del costo laboral unitario



Fuente: Elaboración propia.

Conclusión:

Debido a que el valor calculado para el estadístico t = 1.991 cae en el área de rechazo, entonces se rechaza la Hipótesis Nula y **se acepta la Hipótesis Alterna.** Así, se da evidencia de que con un 95% de confiabilidad, el

Indicador 2 se favorece al tener en ejecución nuestro sistema, y esto se corrobora con la caída porcentual de este indicador **(S/. 69720).**

C. Hipótesis de Investigación 3

Hipótesis Específica

HE: La implementación del sistema de encriptación incrementa de manera significativa el nivel de cobertura de código de los productos de software de una empresa de Lima.

Validación de la Hipótesis

Se toma la T de Student para la diferencia de medias, ya que las muestras son menores a 30. Obteniéndose los siguientes resultados:

Definición de Variables:

MCC_a: Media para el nivel de cobertura de código sin la aplicación.

MCC_d: Media para el nivel de cobertura de código con la aplicación.

Hipótesis Nula:

H_0: La implementación del sistema de encriptación no incrementa de manera significativa el nivel de cobertura de código de los productos de software de una empresa de Lima.

H_0: $MCC_a >= MCC_d$

Hipótesis Alterna:

H_a: La implementación del sistema de encriptación incrementa de manera significativa el nivel de cobertura de código de los productos de software de una empresa de Lima.

H_a: $MCC_a < MCC_d$

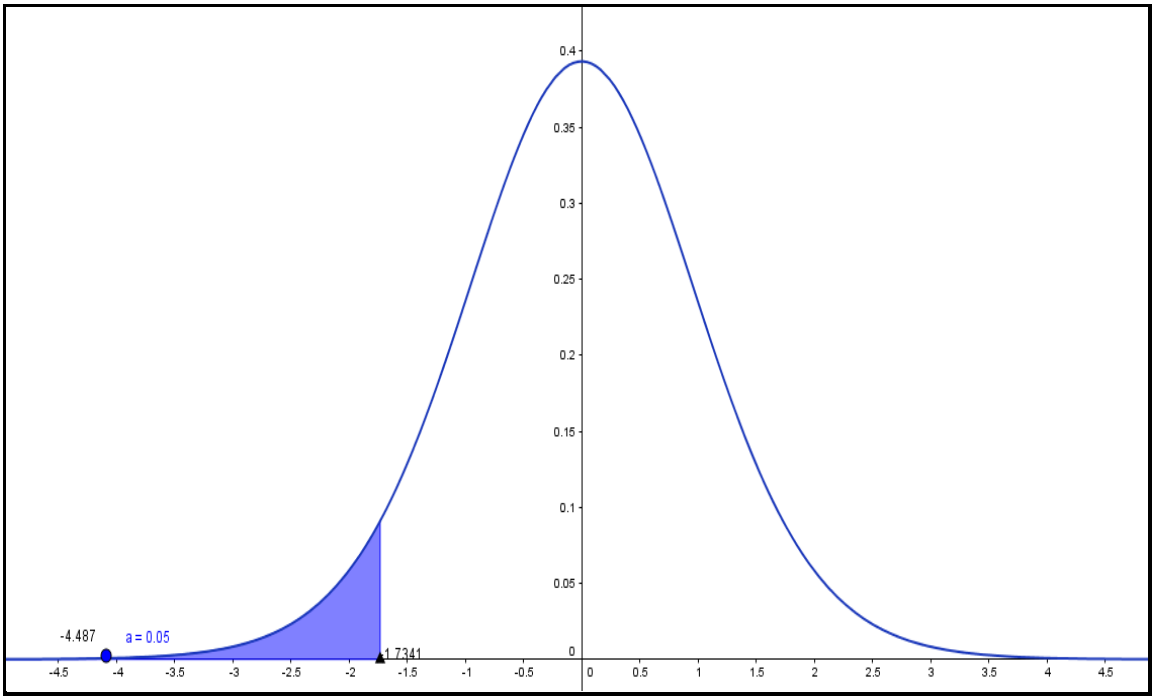
Con el gl (grados de Libertad) de 18 y α (porcentaje de error) de 0.05, se tiene el valor del estadístico t = 1.734, con ello se procede a calcular el t de la diferencia de medias:

ítem	X	S	N
1	0.43	0.0678	10
2	0.63	0.1211	10

$t = \frac{(\overline{X}_1 - \bar{X}_2) - (u_1 - u_2)}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$	$t = \frac{(0.43 - 0.63) - 0}{\sqrt{\left(\frac{0.0678^2}{10} + \frac{0.1211^2}{10}\right)}} = -4.487$

$u_1 - u_2 = 0, debido\ a\ que\ esa\ diferencia\ encaja\ en\ el\ origen\ de\ coordenadas.$

figura 35: Gráfica de Regla de Decisión de la cobertura de código



Fuente: Elaboración propia.

Conclusión:

Debido a que el valor calculado para el estadístico t = -4.487 cae en el área de rechazo, entonces se rechaza la Hipótesis Nula y **se acepta la Hipótesis Alterna.** Así, se da evidencia de que con un 95% de confiabilidad, el Indicador 3 se favorece al tener en ejecución nuestro sistema y esto se corrobora con el incremento del indicador **(20%).**

D. Hipótesis de Investigación 4

Hipótesis Específica

HE: La implementación del sistema de encriptación incrementa significativamente el nivel de cumplimiento de reglas de los productos de software de una empresa de Lima.

Indicador: Cumplimiento de Reglas de codificación

Validación de la Hipótesis

Se toma la T de Student para la diferencia de medias, ya que las muestras son menores a 30. Obteniéndose los siguientes resultados:

Definición de Variables:

MCR_a: Media para el nivel de cumplimiento de reglas sin la aplicación.

$MCRS_d$: Media para el nivel de cumplimiento de reglas con la aplicación.

Hipótesis Nula:

H_0: La implementación del sistema de encriptación no incrementa significativamente el nivel de cumplimiento de reglas de los productos de software de una empresa de Lima.

H_0: $\mathbf{MCR_a > MCR_d}$

Hipótesis Alterna:

H_a: La implementación del sistema de encriptación incrementa significativamente el nivel de cumplimiento de reglas de los productos de software de una empresa de Lima.

H_a: $\mathbf{MCC_a < MCC_d}$

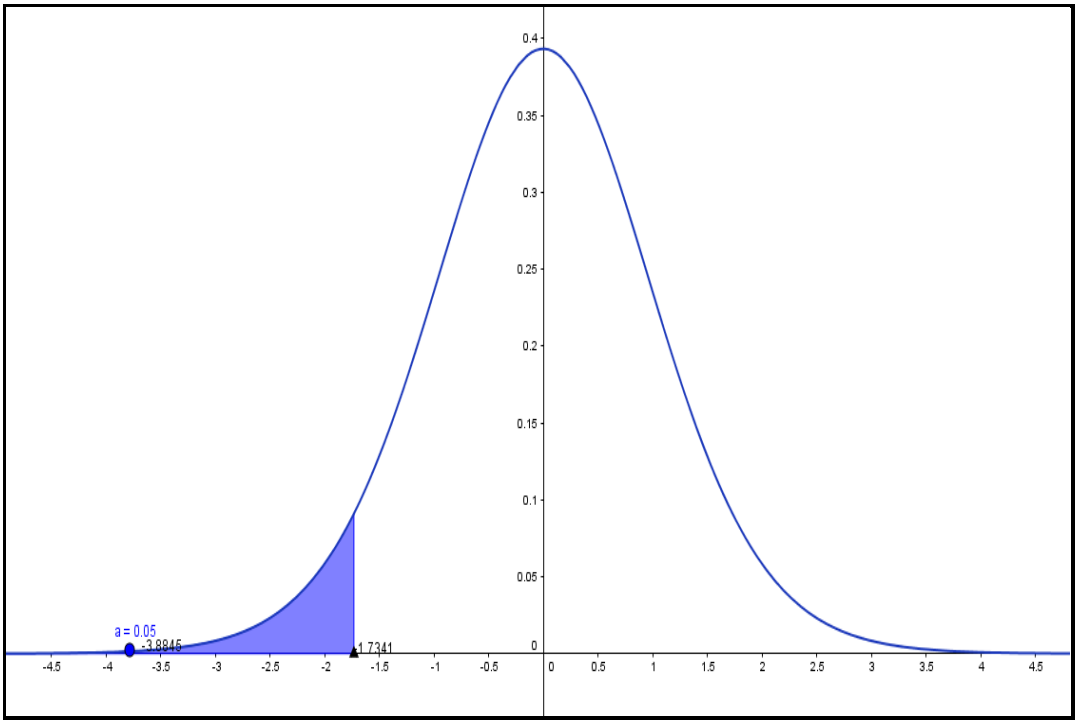
Con el gl (grados de Libertad) de 18 y α (porcentaje de error) de 0.05, se tiene el valor del estadístico t = 1.734, con ello se procede a calcular el t de la diferencia de medias:

ítem	X	S	N
1	0.664	0.0811	10
2	0.79	0.0667	10

$t = \frac{(\overline{X}_1 - \bar{X}_2) - (u_1 - u_2)}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$	$t = \frac{(0.664 - 0.79) - 0}{\sqrt{\left(\frac{0.0811^2}{10} + \frac{0.0667^2}{10}\right)}} = -3.885$

$u_1 - u_2 = 0, debido\ a\ que\ esa\ diferencia\ encaja\ en\ el\ origen\ de\ coordenadas.$

figura 36: Gráfica de Regla de decisión del cumplimiento de reglas



Fuente: Elaboración propia.

Conclusión:

Debido a que el valor calculado para el estadístico t = -3.885 cae en el área de rechazo, entonces se rechaza la Hipótesis Nula y **se acepta la Hipótesis Alterna.** Así, se da evidencia de que con un 95% de confiabilidad, el Indicador 4 se favorece al tener en ejecución nuestro sistema y esto se corrobora con el incremento del indicador **(13%).**

E. Hipótesis de Investigación 5

Hipótesis Específica

HE: La implementación del sistema de encriptación disminuye significativamente el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima.

Indicador: índice Interdependencia entre paquetes

Validación de la Hipótesis

Se toma la T de Student para la diferencia de medias, ya que las muestras son menores a 30. Obteniéndose los siguientes resultados:

Definición de Variables:

IPP_a: Media para el índice Interdependencia entre paquetes sin la aplicación.

IPP_d: Media para el índice Interdependencia entre paquetes con la aplicación.

Hipótesis Nula:

H_0: La implementación del sistema de encriptación no disminuye significativamente el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima.

H_0: M**IPP_a** <= M**IPP_d**

Hipótesis Alterna:

H_a: La implementación del sistema de encriptación disminuye significativamente el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima.

H_a: **$MIPP_a$** > **$MIPP_d$**

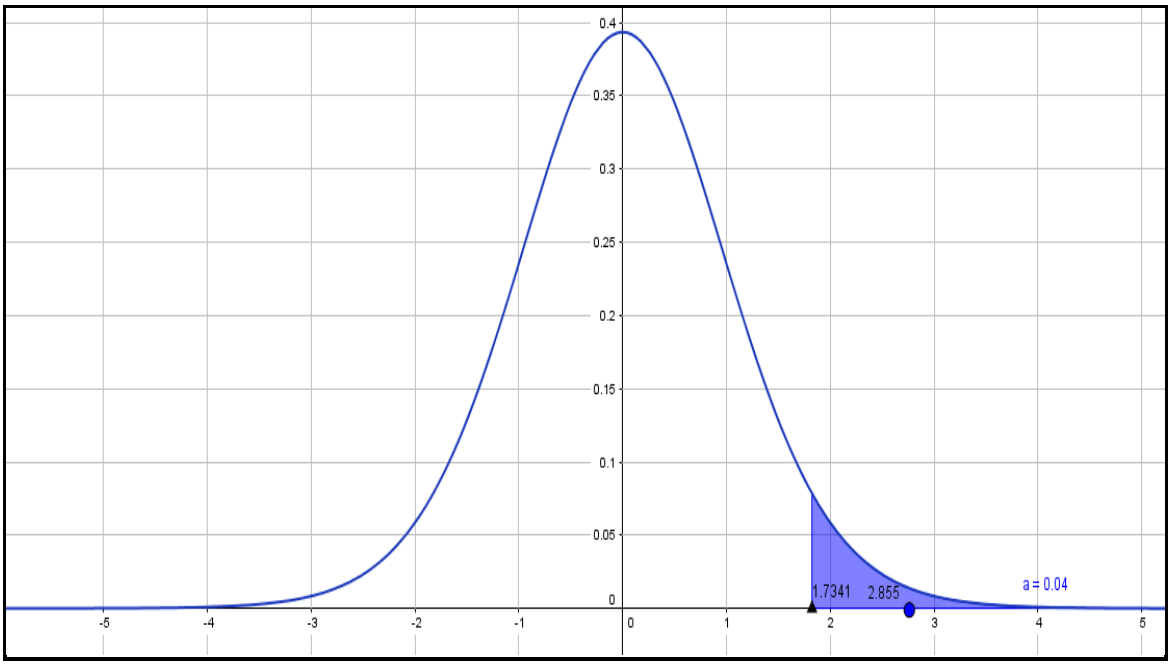
Con el gl (grados de Libertad) de 18 y α (porcentaje de error) de 0.05, se tiene el valor del estadístico t = 1.734, con ello se procede a calcular el t de la diferencia de medias:

ítem	X	S	N
1	0.034	0.0245	10
2	0.01	0.0103	10

$t = \frac{(\overline{X}_1 - \bar{X}_2) - (u_1 - u_2)}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$	$t = \frac{(0.034 - 0.01) - 0}{\sqrt{\left(\frac{0.0245^2}{10} + \frac{0.0103^2}{10}\right)}} = 2.855$

$u_1 - u_2 = 0, debido\ a\ que\ esa\ diferencia\ encaja\ en\ el\ origen\ de\ coordenadas.$

figura 37: Gráfica de Regla de decisión del índice Interdependencia entre paquetes



Fuente: Elaboración propia.

Conclusión:

Debido a que el valor calculado para el estadístico t = 2.855 cae en el área de rechazo, entonces se rechaza la Hipótesis Nula y **se acepta la Hipótesis Alterna.** Así, se da evidencia de que con un 95% de confiabilidad, el Indicador 5 se favorece al tener en ejecución nuestro sistema y esto se corrobora con el decremento del indicador **(2%).**

F. Hipótesis de Investigación 6

Hipótesis Específica

HE: La implementación del sistema de encriptación Disminuye significativamente el nivel de código duplicado de los productos de software de una empresa de Lima.

Indicador: Código duplicado

Validación de la Hipótesis

Se toma la T de Student para la diferencia de medias, ya que las muestras son menores a 30. Obteniéndose los siguientes resultados:

Definición de Variables:

$MNCD_a$: Media para el nivel de código duplicado sin la aplicación.

$MNCD_d$: Media para nivel de código duplicado con la aplicación.

Hipótesis Nula:

H_0: La implementación del sistema de encriptación no disminuye significativamente el nivel de código duplicado de los productos de software de una empresa de Lima.

H_0: $MNCD_a <= MNCD_d$

Hipótesis Alterna:

H_a: La implementación del sistema de encriptación disminuye significativamente el nivel de código duplicado de los productos de software de una empresa de Lima.

H_a: $MNCD_a > MNCD_d$

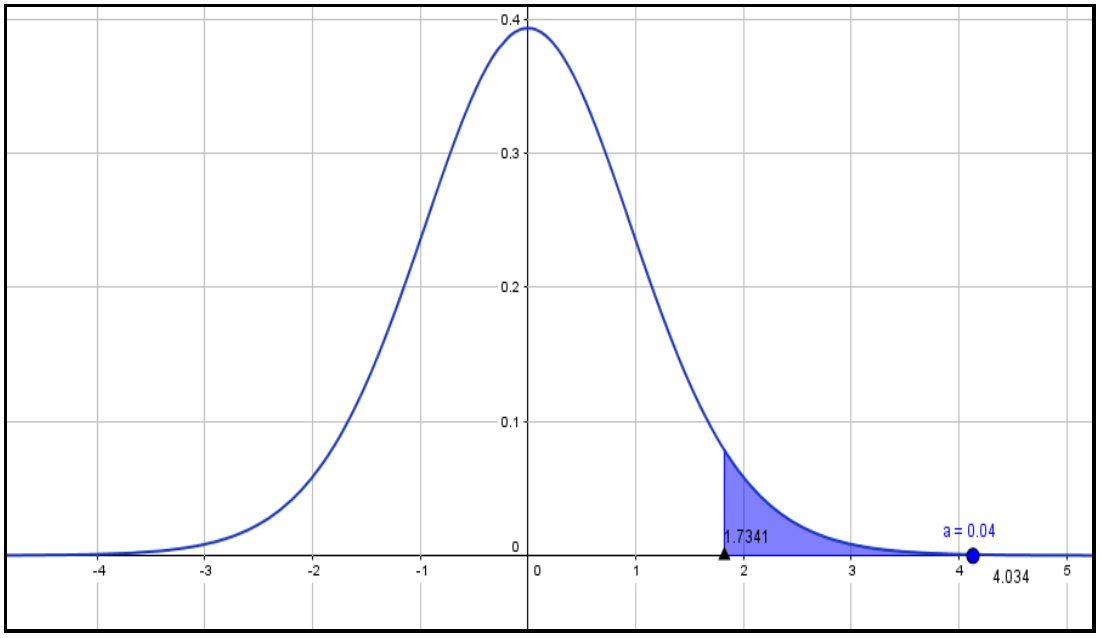
Con el gl (grados de Libertad) de 18 y α (porcentaje de error) de 0.05, se tiene el valor del estadístico t = 1.734, con ello se procede a calcular el t de la diferencia de medias:

ítem	X	S	N
1	0.15	0.0381	10
2	0.09	0.0218	10

$t = \frac{(\overline{X}_1 - \bar{X}_2) - (u_1 - u_2)}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$	$t = \frac{(0.15 - 0.09) - 0}{\sqrt{\left(\frac{0.0381^2}{10} + \frac{0.0218^2}{10}\right)}} = 4.034$

$u_1 - u_2 = 0, debido\ a\ que\ esa\ diferencia\ encaja\ en\ el\ origen\ de\ coordenadas.$

figura 38: Gráfica de Regla de decisión del nivel de código duplicado



Fuente: Elaboración propia.

Conclusión:

Debido a que el valor calculado para el estadístico t = 4.034 cae en el área de rechazo, entonces se rechaza la Hipótesis Nula y **se acepta la Hipótesis Alterna.** Así, se da evidencia de que con un 95% de confiabilidad, el Indicador 6 se favorece al tener en ejecución nuestro sistema y esto se corrobora con el decremento del indicador **(6%).**

G. Hipótesis de Investigación 7

Hipótesis Específica

HE: La implementación del sistema de encriptación disminuye significativamente el nivel de complejidad por método de los productos de software de una empresa de Lima.

Indicador: complejidad por método

Validación de la Hipótesis

Se toma la T de Student para la diferencia de medias, ya que las muestras son menores a 30. Obteniéndose los siguientes resultados:

Definición de Variables:

$MCPM_a$: Media para el nivel de complejidad por método sin la aplicación.

$MCPM_d$: Media para nivel de complejidad por método con la aplicación.

Hipótesis Nula:

H_0: La implementación del sistema de encriptación no disminuye significativamente el nivel de código duplicado de los productos de software de una empresa de Lima.

H_0: $MCPM_a <= MCPM_d$

Hipótesis Alterna:

H_a: La implementación del sistema de encriptación disminuye significativamente el nivel de código duplicado de los productos de software de una empresa de Lima.

H_a: $MCPM_a > MCPM_d$

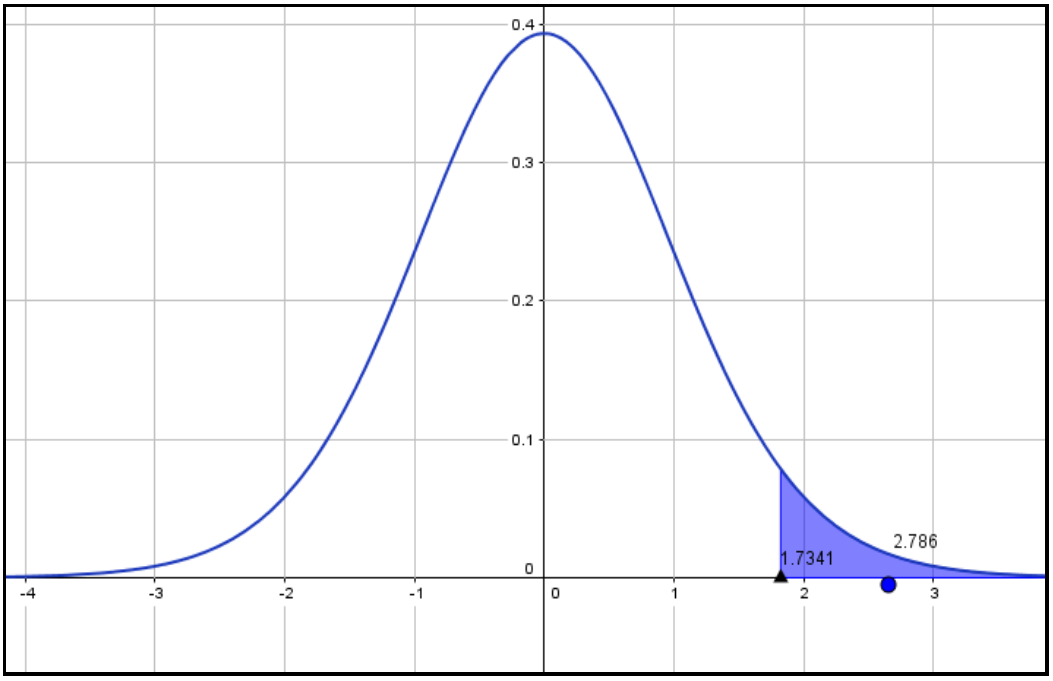
Con el gl (grados de Libertad) de 18 y α (porcentaje de error) de 0.05, se tiene el valor del estadístico t = 1.734, con ello se procede a calcular el t de la diferencia de medias:

ítem	X	S	N
1	0.1	0.0448	10
2	0.06	0.0127	10

$t = \frac{(\overline{X}_1 - \bar{X}_2) - (u_1 - u_2)}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$	$t = \frac{(0.1 - 0.06) - 0}{\sqrt{\left(\frac{0.0448^2}{10} + \frac{0.0127^2}{10}\right)}} = 2{,}786$

$u_1 - u_2 = 0, debido\ a\ que\ esa\ diferencia\ encaja\ en\ el\ origen\ de\ coordenadas.$

figura 39: Gráfica de Regla de decisión del nivel de complejidad por método



Fuente: Elaboración propia.

Conclusión:

Debido a que el valor calculado para el estadístico t = 2.786 cae en el área de rechazo, entonces se rechaza la Hipótesis Nula y **se acepta la Hipótesis Alterna.** Así, se da evidencia de que con un 95% de confiabilidad, el Indicador 7 se favorece al tener en ejecución nuestro sistema y esto se corrobora con el decremento del indicador **(4%).**

H. Hipótesis de Investigación 8

Hipótesis Específica

HE: La implementación del sistema de encriptación disminuye significativamente el nivel de LCOM4 de los productos de software de una empresa de Lima.

Indicador: Nivel LCOM4

Validación de la Hipótesis

Se toma la T de Student para la diferencia de medias, ya que las muestras son menores a 30. Obteniéndose los siguientes resultados:

Definición de Variables:

$NLC4_a$: Media para el nivel de LCOM4 sin la aplicación.

$NLC4_d$: Media para nivel de LCOM4 con la aplicación.

Hipótesis Nula:

H_0: La implementación del sistema de encriptación no disminuye significativamente el nivel de LCOM4 de los productos de software de una empresa de Lima.

H_0: $NLC4_a <= NLC4_d$

Hipótesis Alterna:

H_a: La implementación del sistema de encriptación disminuye significativamente el nivel de LCOM4 de los productos de software de una empresa de Lima.

H_a: $NLC4_a > NLC4_d$

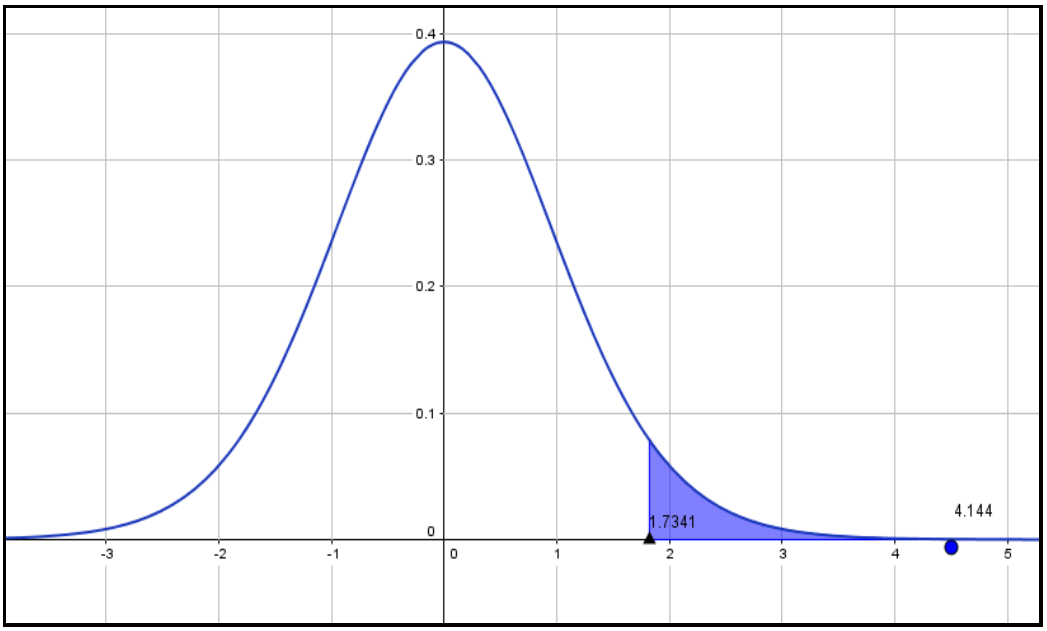
Con el gl (grados de Libertad) de 18 y α (porcentaje de error) de 0.05, se tiene el valor del estadístico t = 1.734, con ello se procede a calcular el t de la diferencia de medias:

ítem	X	S	N
1	0.02	0.0071	10
2	0.011	0.0018	10

$t = \frac{(\overline{X}_1 - \bar{X}_2) - (u_1 - u_2)}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$	$t = \frac{(0.02 - 0.011) - 0}{\sqrt{\left(\frac{0.0071^2}{10} + \frac{0.0018^2}{10}\right)}} = 4.144$

$u_1 - u_2 = 0, debido\ a\ que\ esa\ diferencia\ encaja\ en\ el\ origen\ de\ coordenadas.$

figura 40: Gráfica de Regla de decisión del nivel de LCOM4



Fuente: Elaboración propia.

Conclusión:

Debido a que el valor calculado para el estadístico t = 4.144 cae en el área de rechazo, entonces se rechaza la Hipótesis Nula y **se acepta la Hipótesis Alterna.** Así, se da evidencia de que con un 95% de confiabilidad, el Indicador 8 se favorece al tener en ejecución nuestro sistema y esto se corrobora con el decremento del indicador **(1%).**

I. Hipótesis de Investigación 9

Hipótesis Específica

HE: La implementación del sistema de encriptación incrementa significativamente el nivel de comentarios de código de los productos de software de una empresa de Lima.

Indicador: comentarios de código

Validación de la Hipótesis

Se toma la T de Student para la diferencia de medias, ya que las muestras son menores a 30. Obteniéndose los siguientes resultados:

Definición de Variables:

$MCCO_a$: Media para el nivel de LCOM4 sin la aplicación.

$MCCO_d$: Media para nivel de LCOM4 con la aplicación.

Hipótesis Nula:

H_0: La implementación del sistema de encriptación no incrementa significativamente el nivel de comentarios de código de los productos de software de una empresa de Lima.

H_0: $MCCO_a > MCCO_d$

Hipótesis Alterna:

H_a: La implementación del sistema de encriptación incrementa significativamente el nivel de comentarios de código de los productos de software de una empresa de Lima.

H_a: $MCCO_a < MCCO_d$

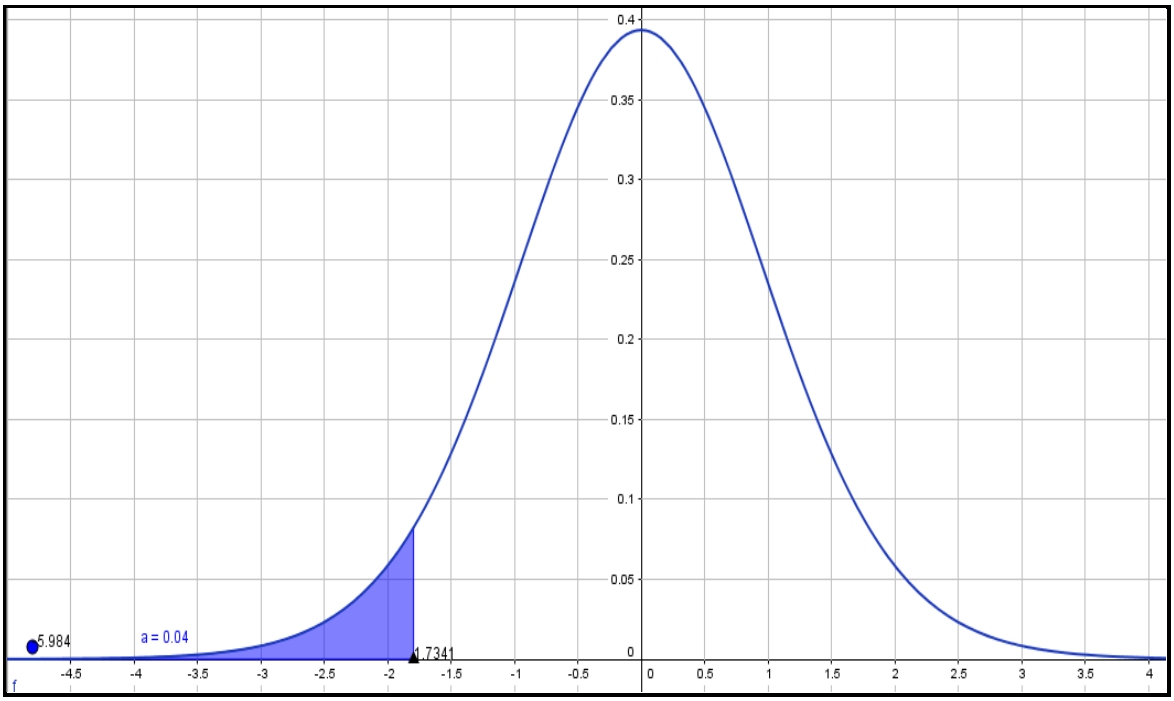
Con el gl (grados de Libertad) de 18 y α (porcentaje de error) de 0.05, se tiene el valor del estadístico t = 1.734, con ello se procede a calcular el t de la diferencia de medias:

ítem	X	S	N
1	0.085	0.0222	10
2	0.15	0.024	10

$t = \frac{(\overline{X}_1 - \bar{X}_2) - (u_1 - u_2)}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}}$	$t = \frac{(0.085 - 0.15) - 0}{\sqrt{\left(\frac{0.0222^2}{10} + \frac{0.024^2}{10}\right)}} = -5.984$

$u_1 - u_2 = 0, debido\ a\ que\ esa\ diferencia\ encaja\ en\ el\ origen\ de\ coordenadas.$

figura 41: Gráfica de Regla de decisión del nivel de comentarios del código



Fuente: Elaboración propia.

Conclusión:

Debido a que el valor calculado para el estadístico t = -5.984 cae en el área de rechazo, entonces se rechaza la Hipótesis Nula y **se acepta la Hipótesis Alterna.** Así, se da evidencia de que con un 95% de confiabilidad, el Indicador 9 se favorece al tener en ejecución nuestro sistema y esto se corrobora con el incremento del indicador **(7%).**

IV. DISCUSIÓN

La investigación presento como propósito identificar la influencia de sistema encriptación en el proceso de desarrollo de software de una empresa de lima,

Con los resultados obtenidos en la presente investigación se comprueba que el sistema de encriptación optimiza el proceso desarrollo de software, en las dimensiones de productividad y calidad de software, en tal sentido se demuestra que el software es adecuado para la empresa, sin embargo el éxito de un software de encriptación según Moya Johanna (2015), "depende de la base conceptual, siendo ello lo más importante para comprender el funcionamiento de los algoritmos de encriptación"[106]. Sobre lo mencionado el conocimiento a un nivel experto de algoritmos y métodos de encriptación asegura que el software se adapte a las necesidades del entorno, por ello estoy de acuerdo en lo indicado por Moya, pues el éxito de cualquier software depende del conocimiento teórico.

También podemos deducir que la implementación de metodologías o herramientas que estén orientados a mejorar el proceso de desarrollo de software, afectaran de forma positiva o negativa los indicadores de productividad laboral, costo laboral y calidad de software. Por otra parte Martinez Yulkeidi (2012) menciona, que "el uso de una metodología o herramienta podría duplicar el nivel productividad del equipo de desarrollo y mejora en 5 veces la practicas de codificación"[107], aunque no concuerdo al 100% en lo indicado, según los resultados se observa un incremento en la productividad y en la calidad del software, pero no de acuerdo a lo especificado, más bien creo que la mejora dependerá mucho del tipo de problema, así como estado actual de sus procesos.

[106] MOYA. Johanna y ESCOBAR, franklin. Desarrollo De Una Aplicación Para Encriptar Información En La Transmisión De Datos En Un Aplicativo De Mensajería Web. [En línea]. [Quito, Ecuador], abril 2017, Disponible en: http://repositorio.puce.edu.ec/bitstream/handle/22000/8335/Disertacion_MoyaCazaJohannaBeatriz_EscobarErazoFranklinAndres.pdf?sequence=1&isAllowed=y

[107] MARTÍNEZ Yulkeidi. Impacto del desarrollo de software dirigido por modelos en la mejora de productividad, mantenibilidad y satisfacción de aplicaciones Web [En línea]. Alicante, España. [Fecha de consulta: 23 enero 2017]. Disponible en: http://rua.ua.es/dspace/handle/10045/26222

Ruiz Ivan (2013), “menciona que la ejecución de un esquema adecuado de pruebas asegura, la calidad del software”[108], estoy de acuerdo con lo indicado, aunque su enfoque está orientado validar el software al cierre de un módulo o sistema, pero no asegura la calidad del software durante el proceso de desarrollo. El tener una codificación de calidad que se pueda generar por ciclos, sin tener la necesidad de esperar que se finalice el desarrollo del software, es el complemento perfecto para asegurar la calidad del producto de software, esto se puede realizar mediante la implementación de herramientas que midan la calidad de código.

Horna Lilly (2014) menciona que “el objetivo de usar modelos de calidad de procesos, conllevará a una mejora en todas las áreas y niveles de la empresa; lo que se puede lograr comprometiendo inicialmente a la gerencia general con su apoyo y participación, y esta a su vez al resto de la organización. La empresa evidenciará mejora en la reducción costos, mejora de procesos y aumento de la calidad del producto final”[109]. Es cierto, para lograr una mejora en los procesos los procesos, la gerencias y las distintas áreas deberán estar comprometidas con los objetivos de la implementación, sea una metodología o una herramienta, al igual Martinez y Horno, creo por resultados de esta investigación, que toda mejora en los procesos, aumenta la calidad y reduce costos, esto se evidencia en los siguientes resultados:

Se ha determinar que la implementación del sistema de encriptación incrementa la productividad laboral en 1.3%, también se determinó que el costo laboral disminuye en S/. 69720. Con respecto a las métricas de calidad se determinó que el uso de los componentes de encriptación eleva de forma significativa el nivel de calidad de los productos de software de la empresa Novatronic, se evidencia un incremento de 20% en nivel de cobertura de Código, un incremento de 13% en el nivel de

[108] RUIZ, Iván. Un framework para el despliegue y evaluación de procesos software [En línea]. Cádiz, España. [Fecha de consulta: 23 enero 2017]. Disponible en: http://rodin.uca.es/xmlui/handle/10498/15725

[109] HORNA Lilly. Implementación de la ISO/IEC 12207:2008 para mejorar los procesos asociados al ciclo de vida de software en una micro empresa peruana cuyo objeto social es el desarrollo de sistemas de información [En línea]. Lima, Peru. [Fecha de consulta: 23 enero 2017]. Disponible en: http://tesis.pucp.edu.pe/repositorio/handle/123456789/6298

cumplimiento de reglas, un incremento de 7% en el nivel de comentarios en el código, un decremento de 2% en el Índice Interdependencia entre paquetes, un decremento de 6% en el nivel de código duplicado, un decremento de 4% en el nivel de complejidad por método, un decremento de 1% en el nivel de LCOM4, siendo aceptada cada una de la hipótesis relacionadas a la métricas de calidad.

Por lo mencionado, es claro que se podrá utilizar las dimensiones, indicadores e instrumentos utilizados en este estudio para la implementación de metodologías o herramientas que busquen la mejora de los procesos de desarrollo de software. Aunque cabe recalcar que este investigación debió incluir otras dimensiones e indicadores orientados a medir los cambios en cada proceso de desarrollo de software, tales como el proceso de dirección de proyecto, el de calidad, el de análisis y diseño, así como el indicador de rentabilidad o el tasa de defectos de software, etc.

También tal como evidencias los resultados el sistema de encriptación, podrían utilizarse en cualquier empresa de software, ya sea como una herramienta, como un componente o como un servicio, cualquiera sea el caso mejora la productividad y calidad del software, producto de la utilización de este software se verán forzados trabajar bajo un arquitectura de componentizacion y bajo un esquema de mediación de calidad del código, esto debido a las dependencias del componente.

El aporte de esta investigación beneficia de forma significativa a la empresa, porque está demostrado que el sistema de encriptación le agrega funcionalidades de seguridad a sus productos, disminuyendo los costos de implementación, en tiempo meno y de forma rápida y sencilla, aunque cabe recalcar que la investigación no solo beneficiara a la empresa, sino a cualquier empresa del rubro de TIC, Financiero que esté interesado en agregar mecanismos de encriptación a sus sistemas.

V. CONCLUSIÓN

1. Se determinó que la implementación del sistema de encriptación influye en productividad laboral del proceso de desarrollo de software de una empresa de lima, incrementado la productividad laboral en 1.3%, esto demuestra que la hipótesis que indica que la implementación del sistema de encriptación incrementa significativamente la productividad laboral del proceso de desarrollo de software de una empresa de lima.

2. Se concluye que la influencia de la implementación del sistema de encriptación en el costo laboral unitario del proceso de desarrollo de software de una empresa de Lima, disminución del costo laboral en S/. 69720, da por aceptada la hipótesis que menciona que la implementación del sistema de encriptación reduce de manera significativa el costo laboral unitario del proceso de desarrollo de software de una empresa de lima.

3. Se observó la influencia de la implementación del sistema de encriptación en el nivel de cobertura de código de los productos de software de una empresa de Lima, esta incrementa en 20% en nivel de cobertura de Código, reconociendo que la hipótesis que menciona que la implementación del sistema de encriptación incrementa de manera significativa el nivel de cobertura de código de los productos de software de una empresa de Lima, es correcta.

4. Se determinó la influencia de la implementación del sistema de encriptación en el nivel de cumplimiento de reglas de los productos de software de una empresa de Lima, este incremento en 13% el nivel de cumplimiento de reglas, corroboran que la hipótesis que menciona que la implementación del sistema de encriptación incrementa significativamente el nivel de cumplimiento de reglas de los productos de software de una empresa de Lima, es aceptada.

5. Se concluyó que la influencia de la implementación del sistema de encriptación en el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima, decremento en 2% el Índice Interdependencia entre paquetes, con ello se da como válida la hipótesis que menciona que la implementación del sistema de encriptación disminuye

significativamente el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima.

6. Se observó la influencia de la implementación del sistema de encriptación en el nivel de código duplicado de los productos de software de una empresa de Lima, este decremento en 6% el nivel de código duplicado, por ello se reconoce que la hipótesis que menciona que la implementación del sistema de encriptación Disminuye significativamente el nivel de código duplicado de los productos de software de una empresa de Lima, es aceptada.

7. Se determinó la influencia de la implementación del sistema de encriptación en el nivel de complejidad por método de los productos de software de una empresa de Lima. Este decremento en 4% el nivel de complejidad por método por ello se reconoce que la hipótesis que indica que la implementación del sistema de encriptación disminuye significativamente el nivel de complejidad por método de los productos de software de una empresa de Lima, es correcta.

8. Se concluyó que la influencia de la implementación del sistema de encriptación en el nivel de LCOM4 de los productos de software de una empresa de Lima, decremento en 1% el nivel de LCOM4, de esta forma se acepta la hipótesis que menciona que la implementación del sistema de encriptación disminuye significativamente el nivel de LCOM4 de los productos de software de una empresa de Lima.

9. Se observó la influencia de la implementación del sistema de encriptación en el nivel de comentarios de código de los productos de software de una empresa de Lima, está incremento en 7% el nivel de comentarios en el código con ello se acepta la hipótesis que menciona que la implementación del sistema de encriptación incrementa significativamente el nivel de comentarios de código de los productos de software de una empresa de Lima.

Los resultados aseveran el cumplimiento de los objetivos de la investigación, esto se evidencia de forma clara en resultados de análisis estadístico.

VI. RECOMENDACIONES

Se recomienda que:

El área de operaciones de la empresa implemente una arquitectura de componentizacion software, así como repositorio de componentes o librerías, para que el desarrollador reutilice las funcionalidades codificadas en estos y no tenga que implementar algo ya desarrollados, con estas mejoras su tiempo de desarrollo disminuiría y la productividad se incrementaría.

El área de RR.HH. realice actividades orientadas a mejorar el clima laboral, para que los factores del entorno, no disminuyan la productividad laboral.

El área de desarrollo, deberá implementa una herramientas que permitan medir y cuantificar la calidad del software desarrollado tales como el sonarQuBe (Código libre), con ello se estandarizara las prácticas de codificación y se incrementara la calidad software.

El área de aseguramiento de la calidad, debe agregar como parte de los procesos de desarrollo de software, puntos de control en los procedimientos de desarrollo, en los cuales norme calidad del software debe ser medida mediante el uso de una herramienta como el sonar.

El área de desarrollo, debe de realizar una evaluación de calidad a cada uno de los productos de la empresa con el fin determinar las mejoras posibles al código.

El área de operaciones debe realizar exposiciones de webinar, donde se traten temas de calidad de software, métricas y herramientas como el sonarQuBe.

.

VII. REFERENCIAS

1. **ACADEMIA.** Sistema de procesamiento de transacciones fundación universitaria los libertadores [En Línea]. [Fecha de consulta: 02 de Setiembre 2016]. Disponible en: http://www.academia.edu/8957812/SISTEMA_DE_PROCESAMIENTO_DE_TRANSACCIONES_FUNDACI%C3%93N_UNIVERSITARIA_LOS_LIBERTADORES

2. **ABDALEIS,** Gretel. Encriptación de Datos [En línea]. [Fecha consulta: febrero 2017]. Disponible en: http://encripdedatos.blogspot.pe/

3. **ANCELIT**, Analida. Encriptación de Datos [En línea]. [Fecha consulta: febrero 2017]. Disponible en: http://encriptaciondedatos.blogspot.pe/2007/09/encriptacion-de-datos.html

4. **ALBARES**, Franklin. Proceso de desarrollo de software [En línea]. [Fecha consulta: febrero 2017]. Disponible en: http://brfranciscoosunaiuty.blogspot.pe/2012/07/proceso-de-desarrollo-de-software.html

5. **AOS**. Justificación económica [En línea] [Citado el: 05 de octubre de 2015.]. Disponible en: http://www.oas.org/dsd/publications/unit/oea42s/ch06.htm#f. justificación económica

6. **APRENDERAPROGRAMAR.** Calidad del software. Métricas y fiabilidad de aplicaciones [En Línea]. [Fecha de consulta: 02 de setiembre 2016]. Disponible en: http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=198:calidad-del-software-metricas-y-fiabilidad-de-aplicaciones-1a-parte-dv00103a&catid=45:tendencias-programacion&Itemid=164

7. **BARBIERI**, Sebastián. Framework de mejora de procesos de desarrollo de software [en línea]. Lic. Alejandro Bianchi. Tesis Magistral. Universidad. Universidad Nacional de la Plata, Argentina. [Fecha de consulta: 02 de Setiembre 2016]. Disponible en: http://postgrado.info.unlp.edu.ar/Carreras/Magisters/Ingenieria_de_Software/Tesis/Sebastian_Barbieri.pdf

8. **BERNAL**, César A. Metodología de la investigación. Tercera edición PEARSON EDUCACIÓN, Colombia, 2010

9. **BOXCRYPTOR**. Cifrado AES y RSA [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://www.boxcryptor.com/es/cifrado

10. **CAMPBELL**, Donald y STANLEY, Julian. Diseños experimentales y cuasi experiméntales en la investigación social, 1978

11. **CHICHARRO** Jesús y **BLASCO** Jorge. Desarrollo De Una Aplicación Móvil Con Cifrado De Datos Por Geo Posición [En línea]. Madrid, España .Fecha de consulta: 23 enero 2017]. Disponible en: https://e-archivo.uc3m.es/handle/10016/18303

12. **CORAL,** Yadira**.** Validez y confiabilidad de los instrumentos de investigación para la recolección de datos, Carabobo. Venezuela. 2008.

13. **CASHFLOWCOMPARTE**. Comprar o Desarrollar Software [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://cashflowcomparte.blogspot.pe/2009/01/comprar-o-desarrollar-software.html

14. **CODIGOPROGRAMACION**. starUML [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://codigoprogramacion.com/tag/staruml#.WMMa6m81_IU

15. **CCM**. criptografía de clave privada o clave secreta [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://es.ccm.net/contents/126-criptografia-de-clave-privada-o-clave-secreta

16. **CPEIG**. Jornada sobre Calidad del Producto Software e ISO 25000, Santiago de Compostela, España, 10 de junio de 2014

17. **CRYPTOFORGET.** Encriptación y seguridad [En Línea]. [Fecha de consulta: 23 enero 2017]. Disponible en: http://www.cryptoforge.com.ar/encriptacion-seguridad-de-la-informacion.htm

18. **DRAKE** Jose. Proceso de desarrollo de aplicaciones [En Línea]. [Fecha Consulta: octubre 2016]. Disponible en: http://www.ctr.unican.es/asignaturas/Ingenieria_Software_4_F/Doc/M1_08_Proceso.pdf

19. **ECURED**. Calidad en el Desarrollo Software Basado en Componentes [En Línea]. [Fecha de consulta: 23 febrero 2017]. Disponible en: https://www.ecured.cu/Calidad_en_el_Desarrollo_Software_Basado_en_Componentes

20. **ECURED**. RC5 [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.ecured.cu/index.php/RC5.

21. **EMPRENDEPYME**. La productividad [En línea] [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.emprendepyme.net/que-es-la-productividad-empresarial.html.

22. **ESAN**. Las Diez áreas de conocimiento del PMI [En Línea]. [Fecha de consulta: 02 de setiembre 2016]. Disponible en: http://www.esan.edu.pe/apuntes-empresariales/ 2017/08/las-diez-areas-de-conocimiento-segun-el-pmi/

23. **EVERAC99**. Desempeño de C++ vs. Java [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://everac99.wordpress.com/2012/12/13/desempeno-de-c-vs-java-todo-reside-en-la-calidad-del-compilador-y-del-programador-tambien/

24. **FIDIAS G**., Arias. El Proyecto de Investigación: Introducción a la metodología científica. 6a Edición. Caracas, Venezuela: Editorial Episteme, C.A., 2012.

25. **GARZAS**, Javier. Primeras fábricas de software, concepto e historia [En Línea]. [Fecha de consulta: 23 mayo 2017]. Disponible en: http://www.javiergarzas.com/2007/05/primeras-fbricas-software-concepto-e.html

26. **Genbetadev**. Eclipse IDE. [En línea] [Citado el: 05 de octubre de 2015.].Disponible en: https://www.genbetadev.com/herramientas/eclipse-ide

27. **GRUPO IZAMORAR**. Tipos y o clasificación de los sistemas de información. [En Línea]. [Fecha de consulta: 02 de setiembre 2016]. Disponible en: http://izamorar.com/tipos-y-o-clasificacion-de-los-sistemas-de-informacion/

28. **HERNANDEZ SAMPIERI**, Roberto. Metodología de la Investigación. 6a. México: McGRAW W-HILL, 2014

29. **HORNA** Lilly. Implementación de la ISO/IEC 12207:2008 para mejorar los procesos asociados al ciclo de vida de software en una micro empresa peruana cuyo objeto social es el desarrollo de sistemas de información [En línea]. Lima, Peru. [Fecha de consulta: 23 enero 2017]. Disponible en: http://tesis.pucp.edu.pe/repositorio/handle/123456789/6298

30. **INGENIERIAINDUSTRIALONLINE**. Producción [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.ingenieriaindustrialonline.com/herramientas-para-el-ingeniero-industrial/producci%C3%B3n

31. **INEGI**. Calculo de los índices de productividad laboral y del costeo unitario de la mano de obra [En línea] [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.inegi.org.mx/saladeprensa/boletines/ 2017/ipl/ipl2017_09.pdf

32. **INEGIO**. Metodología De Cálculo De Indicadores De Productividad Laboral [En Línea]. [Fecha de consulta: 23 enero 2017]. Disponible en: http://www.inegi.org.mx/inegi/spc/doc/bibliografia/7AC06BCF.pdf

33. **INGENIERIADESOFTWARE**. Extreme-Programing [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html

34. **JAVA.** What is java [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://www.java.com/es/about/whatis_java.jsp

35. **JIMÉNEZ**, Pablo y **QUEZADA**, Iván. Desarrollo De Una Aplicación Para La Encriptación Y Desencriptación De La Información De Un Directorio Mediante Autenticación Por Pki Utilizando Tecnología Active Directory [en línea]. Universidad del Azuay. Quito, Ecuador. [Fecha Consulta: julio 2017]. Disponible en: http://dspace.uazuay.edu.ec/bitstream/datos/2685/1/09221.pdf

36. **LAGOS**, Alejandro**.** Mejora Sistemática Del Proceso De Desarrollo De Software De La División De Autoservicio De DTS [en línea]. Universidad de Chile, Santiago. [Fecha consulta: enero 2017], Disponible en: http://repositorio.uchile.cl/bitstream/handle/2250/110974/cf-lagos_as.pdf?sequence=1&isAllowed=

37. **MARKETINGDEPYMES**. Calidad de los productos. [En línea] [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.marketingdepymes.com/sala-de-lectura/instrumentos/el-concepto-de-calidad-en-los-productos-tangibles-y-en-los-servicios

38. **MARTÍNEZ** Alejandro y **MARTÍNEZ** Raúl. Guía a Rational Unified Process [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://anaylenlopez.files.wordpress.com/2011/03/trabajo-guia20rup.pdf

39. **MARTÍNEZ** Yulkeidi. Impacto del desarrollo de software dirigido por modelos en la mejora de productividad, mantenibilidad y satisfacción de aplicaciones Web [En línea]. Alicante, España. [Fecha de consulta: 23 enero 2017]. Disponible en: http://rua.ua.es/dspace/handle/10045/26222

40. **MÉNDEZ**, Ángel**.** Mejora Del Proceso Software De Una Pequeña Empresa Desarrolladora De Software: Caso Competisoft- Perú - Lim.Omega, Primer Ciclo [en línea]. Universidad la Católica del Perú. Lima, Perú. [Fecha Consulta: abril 2017]. Disponible en: http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/1615/MENDEZ_BAZALAR_ANGEL_COMPETISOFT_LIM_OMEGA.pdf?sequence=1&isAllowed=y

41. **MINISTERIO DE JUSTICIA**, Ley de protección de datos personales [en línea]. [Fecha de consulta: 23 enero 2017]. Disponible

en:https://www.minjus.gob.pe/wp-content/uploads/2013/04/DS-3-2013-JUS.REGLAMENTO.LPDP_.pdf

42. **MICROSOFT**. Criptografía y seguridad en computadores. [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://support.microsoft.com/es-pe/kb/257591

43. **MOYA**. Johanna y **ESCOBAR**, franklin. Desarrollo De Una Aplicación Para Encriptar Información En La Transmisión De Datos En Un Aplicativo De Mensajería Web. [En línea]. Universidad Católica de Ecuador, Quito. [Fecha de consulta: 23 enero 2017]. Disponible en: http://repositorio.puce.edu.ec/bitstream/handle/22000/8335/Disertacion_MoyaCazaJohannaBeatriz_EscobarErazoFranklinAndres.pdf?sequence=1&isAllowed=y

44. **NORIEGA** Raúl. Proceso de desarrollo de software, 2da edición. on Amazon.com

45. **Novatronic**. Acerca de [En línea]. [Fecha de consulta: 23 enero 2017]. Disponible en: http://www.novatronic.com/archives/sobre_novatronic/quienes_somos/index.php

46. **ÑAUPAS** Humberto. Metodología de investigación. Cuarta edición. Bogotá, Colombia. Ediciones de la U. 2013. ISBN 978-958-762-188-4.

47. **ORGANIZACIÓN DE LAS NACIONES UNIDAS**. Costos de producción [En línea] [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.fao.org/docrep/003/v8490s/v8490s06.htm

48. **ORTUZAR**, Juan**.** Mejoramiento De Procesos De Producción, Mantención Y Soporte de Productos De Software [en línea]. Universidad de Chile, Santiago. [Fecha consulta: enero 2017]. Disponible en: http://www.tesis.uchile.cl/tesis/uchile/2011/cf-ortuzar_je/html/index-frames.html

49. **PELÁEZ**, Juan. Arquitectura basada en Componentes. [En línea] [Citado el: 05 de octubre de 2015.] . Disponible en: http://geeks.ms/blogs/jkpelaez/archive/2009/04/18/arquitectura-basada-en-componentes.aspx

50. **PRESSMAN** Roger. Software Engineering: A Practitioner's Approach. McGraw-Hill. 1ra edición

51. **PROYECTOSAGILES**. Qué es SCRUM [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: https://proyectosagiles.org/que-es-scrum

52. **Project Management Institute,** A Guide To The Project Management Body Of Knowledge (PMBOK Guides), 5a. edition.

53. **RUIZ,** Iván. Un framework para el despliegue y evaluación de procesos software [En línea]. Cádiz, España. [Fecha de consulta: 23 enero 2017]. Disponible en: http://rodin.uca.es/xmlui/handle/10498/16725

54. **RUEDA**, Julio. Aplicación de la Metodología RUP para el desarrollo rápido de aplicaciones basado en el Estándar J2EE [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en http://biblioteca.usac.edu.gt/tesis/08/08_0308_CS.pdf

55. **SECURING THE HUMAN**. Entendiendo el cifrado [En línea]. [Fecha consulta: febrero 2017]. Disponible en: https://securingthehuman.sans.org/newsletters/ouch/issues/OUCH-201107_sp.pdf

56. **SEGU INFO**. Seguridad [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.segu-info.com.ar/logica/seguridadlogica.htm.

57. **SONARQUBEHISPANO**. Definición de Métricas. [En Línea]. [Fecha de consulta: 23 febrero 2017]. Disponible en: https://sonarqubehispano.org/pages/viewpage.action?pageId=4980840

58. **TALENS** Sergio. Criptografía [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.uv.es/sto/articulos/BEI-2003-04/criptologia.pdf

59. **THALU**. Software enlatado vs software a medida [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.thalu.net/software-enlatado-vs-software-a-medida/

60. **TRINIDAD**, Pablo. **SEGURA,** Sergio y **RUIZ**, Antonio. Evaluación y seguimiento de trabajos en equipo de desarrollo de software a través de la calidad del código fuente [En línea]. Sevilla, España. [Fecha de consulta: 23 enero 2017]. Disponible en: https://upcommons.upc.edu/bitstream/handle/2099/16038/016.pdf

61. **TRUECRYPT**. Encrypted by TrueCryp [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://truecrypt.sourceforge.net/

62. **UNIVERSIDAD NACIONAL AUTÓNOMA**. Principales Algoritmos Simétricos - 3DES o TDES) [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://redyseguridad.fi-p.unam.mx/proyectos/criptografia/criptografia/index.php/4-criptografia-simetrica-o-de-clave-secreta/41-introduccion-a-la-criptografia-simetrica/413-principales-algoritmos-simetricos?showall=&start=5

63. **VALDERRAMA** Santiago. Pasos para elaborar proyectos y tesis de investigación. Tercera edición. Editorial San Marcos Lima, Perú .2013. ISBN: 9786123028787

64. **VISCONTI**, Marcello. Métricas clásicas de Software [En línea]. [Fecha Consulta: 05 de octubre de 2016]. Disponible en: http://www.eici.ucm.cl/Academicos/R_Villarroel/descargas/ingenieriaii/tema8.pdf

65. **ZATOR**. Que es C++ [En línea]. [Fecha Consulta: 05 de Octubre de 2016].Disponible en: http://www.zator.com/Cpp/E0_I.htm

ANEXO

Anexo 1: Organización de la Empresa

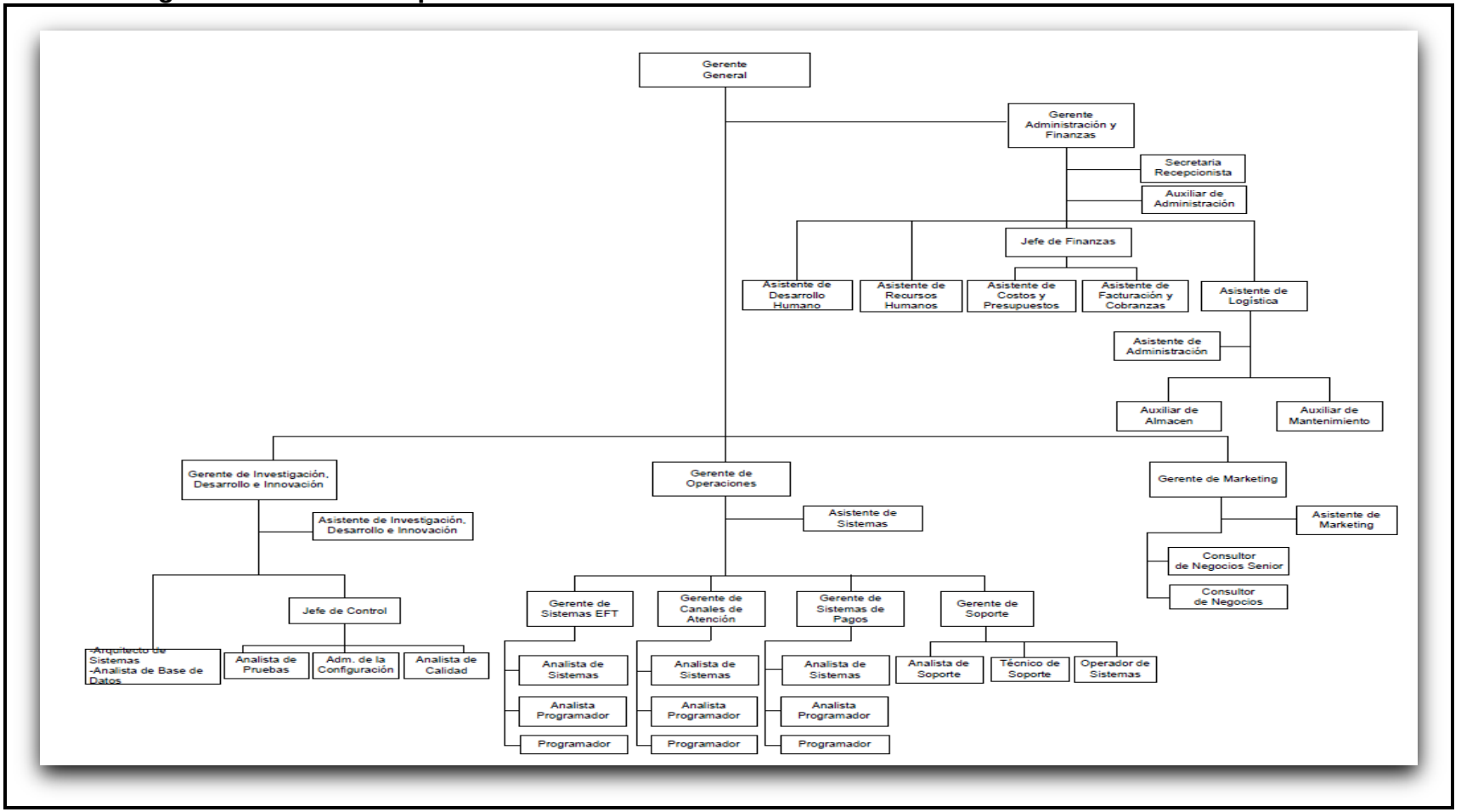


Figura 42: Organigrama de la empresa
Fuente: Elaboración propia

PROBLEMA	OBJETIVOS	HIPÓTESIS	VARIABLES	DIMENSIONES	INDICADORES	ESCALA	METODOS	TECNICAS	INSTRUMENTOS
Problema general	Objetivo general	Hipótesis General	sistema de encriptación	Nivel de seguridad	Nivel de Riesgo = probabilidad(1) * Impacto(2) (1) (2) Insignificante= 1; Baja = 2; Mediana = 3; Alta = 4	razón	Tipo de Investigación: Aplicada Descriptiva Nivel de Investigación: Correlacional Metodología de la Investigación: Cuantitativa Diseño de la Investigación: Pre Experimental Universo: 30 colaboradores del área de desarrollo Muestra: 30 colaboradores del área de desarrollo	Observaciones De Campo	Fichas De observación
¿De qué manera influye la Implementación de un sistema de encriptación para optimizar el proceso de desarrollo de software de una empresa de Lima?	Determinar la influencia de la Implementación del sistema de encriptación para optimizar el proceso de desarrollo de software de una empresa de Lima	La Implementación del sistema de encriptación optimiza significativamente el proceso de desarrollo de software de una empresa de Lima.							
Problema específico	Objetivo específico	Hipótesis específica	Proceso de desarrollo de software	Productividad	Productividad laboral = Producción/ Número de trabajadores	Razón			
¿De qué manera influye la implementación del sistema de encriptación sobre la productividad laboral en el proceso de desarrollo de software de una empresa de Lima?	Determinar la influencia de la implementación del sistema de encriptación sobre la productividad laboral del proceso de desarrollo de software de una empresa de Lima.	La implementación del sistema de encriptación incrementa significativamente la productividad laboral del proceso de desarrollo de software de una empresa de Lima.							
¿De qué manera influye la implementación del sistema de encriptación en el costo laboral unitario del proceso de desarrollo de software de una empresa de Lima?	Determinar la influencia de la implementación del sistema de encriptación en el costo laboral unitario del proceso de desarrollo de software de una empresa de Lima.	La implementación del sistema de encriptación reduce de manera significativa el costo laboral unitario del proceso de desarrollo de software de una empresa de Lima			Costo laboral unitario = Remuneración Media/Productividad laboral	Razón			
¿De qué manera influye la implementación del sistema de encriptación en el nivel de cobertura de código de los productos de software de una empresa de Lima?	Determinar la influencia de la implementación del sistema de encriptación en el nivel de cobertura de código de los productos de software de una empresa de Lima.	La implementación del sistema de encriptación incrementa de manera significativa el nivel de cobertura de código de los productos de software de una empresa de Lima.		calidad de Desarrollo de software	1. Cobertura de Código = (#Unit_test/# métodos)*100 2. Cumplimiento de Reglas = 100-((Violaciones/líneas de código)* 100) 3. Índice Interdependencia entre paquetes=2*(dependencia entre archivos/número de desentendencia entre directorios)*100 4. Duplicados =(Número de líneas duplicadas/Número total de líneas) *100 5. Complejidad por Método = #Complejidad/#método 6. LCOM4=#atributos comunes de métodos de clases/4 7. Comentarios=(Líneas de comentadas/(líneas de código + líneas comentadas))* 100	Razón			
¿De qué manera influye la implementación del sistema de encriptación en el nivel de cumplimiento de Reglas de los productos de software de una empresa de Lima?	Determinar la influencia de la implementación del sistema de encriptación en el nivel de cumplimiento de reglas de los productos de software de una empresa de Lima.	La implementación del sistema de encriptación incrementa significativamente el nivel de cumplimiento de reglas de los productos de software de una empresa de Lima.							
¿De qué manera influye la implementación del sistema de encriptación en el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima?	Determinar la influencia de la implementación del sistema de encriptación en el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima.	La implementación del sistema de encriptación disminuye significativamente el índice Interdependencia entre paquetes de los productos de software de una empresa de Lima.							
¿De qué manera influye la implementación del sistema de encriptación en el nivel de código duplicado de los productos de software de una empresa de Lima?	Determinar la influencia de la implementación del sistema de encriptación en el nivel de código duplicado de los productos de software de una empresa de Lima.	La implementación del sistema de encriptación Disminuye significativamente el nivel de código duplicado de los productos de software de una empresa de Lima.							
¿De qué manera influye la implementación del sistema de encriptación en el nivel de complejidad por método de los productos de software de una empresa de Lima?	Determinar la influencia de la implementación del sistema de encriptación en el nivel de complejidad por método de los productos de software de una empresa de Lima.	La implementación del sistema de encriptación disminuye significativamente el nivel de complejidad por método de los productos de software de una empresa de Lima.							
¿De qué manera influye la implementación del sistema de encriptación en el nivel de LCOM4 de los productos de software de una empresa de Lima?	Determinar la influencia de la implementación del sistema de encriptación en el nivel de LCOM4 de los productos de software de una empresa de Lima.	La implementación del sistema de encriptación disminuye significativamente el nivel de LCOM4 de los productos de software de una empresa de Lima.							
¿De qué manera influye la implementación del sistema de encriptación en el nivel de comentarios de código de los productos de software de una empresa de Lima?	Determinar la influencia de la implementación del sistema de encriptación en el nivel de comentarios de código de los productos de software de una empresa de Lima.	La implementación del sistema de encriptación incrementa significativamente el nivel de comentarios de código de los productos de software de una empresa de Lima.							

Anexo 2: Matriz de consistencia

Tabla 30: Matriz de consistencia - Fuente: Elaboración propia

Anexo 3: Ficha de observación resultados

Tabla 31: ***Nivel de productividad laboral antes de la implementación del aplicativo***

Productividad Laboral Pre				
Ítem	Implementación de productos	Productos	Número de trabajadores	Productividad
1	Producto1	1	25	0.040
2	Producto2	1	26	0.038
3	Producto3	1	15	0.067
4	Producto4	1	20	0.050
5	Producto5	1	50	0.020
6	Producto6	1	45	0.022
7	Producto7	1	30	0.033
8	Producto8	1	23	0.043
9	Producto9	1	40	0.025
10	Producto10	1	25	0.040

Fuente: Elaboración propia.

Tabla 32: ***Nivel de productividad laboral despues de la implementación del aplicativo***

Productividad Laboral Post				
Ítem	Implementación de productos	Productos	Número de trabajadores	Productividad
1	Producto1	1	23	0.043
2	Producto2	1	23	0.043
3	Producto3	1	13	0.077
4	Producto4	1	15	0.067
5	Producto5	1	30	0.033
6	Producto6	1	28	0.036
7	Producto7	1	27	0.037
8	Producto8	1	18	0.056
9	Producto9	1	20	0.050
10	Producto10	1	16	0.063

Fuente: Elaboración propia.

Tabla 33: Costo Laboral unitario antes de implementado la aplicación

Costo Laboral Unitario Pre				
Ítem	**Implementación de productos**	**Productividad**	**Sueldo promedio**	**Costo laboral unitario**
1	Producto1	0.040	3480	S/. 87,000.00
2	Producto2	0.038	3640	S/. 94,640.00
3	Producto3	0.067	2320	S/. 34,800.00
4	Producto4	0.050	2880	S/. 57,600.00
5	Producto5	0.020	6860	S/. 343,000.00
6	Producto6	0.022	6080	S/. 273,600.00
7	Producto7	0.033	4380	S/. 131,400.00
8	Producto8	0.043	3340	S/. 76,820.00
9	Producto9	0.025	5840	S/. 233,600.00
10	Producto10	0.040	3560	S/. 89,000.00

Fuente: Elaboración propia.

Tabla 34: Costo Laboral unitario después de implementado la aplicación

Costo Laboral unitario Post				
Ítem	**Implementación de productos**	**Productividad**	**Sueldo promedio**	**Costo laboral unitario**
1	Producto1	0.043	3200	S/. 73,600.00
2	Producto2	0.043	3220	S/. 74,060.00
3	Producto3	0.077	2080	S/. 27,040.00
4	Producto4	0.067	2340	S/. 35,100.00
5	Producto5	0.033	4720	S/. 141,600.00
6	Producto6	0.036	4160	S/. 116,480.00
7	Producto7	0.037	3980	S/. 107,460.00
8	Producto8	0.056	2820	S/. 50,760.00
9	Producto9	0.050	2940	S/. 58,800.00
10	Producto10	0.063	2460	S/. 39,360.00

Fuente: Elaboración propia.

Tabla 35: Cobertura de código antes de la implementación del sistema

Ítem	Implementación de productos	Cobertura de Código (60 % - 100 %)
1	Producto1	35%
2	Producto2	38%
3	Producto3	45%
4	Producto4	48%
5	Producto5	43%
6	Producto6	48%
7	Producto7	50%
8	Producto8	45%
9	Producto9	30%
10	Producto10	50%
		Unit test

Fuente: Elaboración propia.

Tabla 36: Cobertura de código después de la implementación del sistema

Ítem	Implementación de productos	Cobertura de Código (60 % - 100 %)
1	Producto1	40%
2	Producto2	45%
3	Producto3	65%
4	Producto4	63%
5	Producto5	60%
6	Producto6	65%
7	Producto7	70%
8	Producto8	80%
9	Producto9	70%
10	Producto10	71%
		Unit test

Fuente: Elaboración propia.

Tabla 37: Cumplimiento de reglas antes de la implementación del sistema

Ítem	Implementación de productos	Cumplimiento de Reglas (80 % - 100 %)
1	Producto1	60%
2	Producto2	65%
3	Producto3	63%
4	Producto4	70%
5	Producto5	71%
6	Producto6	64%
7	Producto7	75%
8	Producto8	80%
9	Producto9	51%
10	Producto10	65%
		Reglas(PMD)

Fuente: Elaboración propia

Tabla 38: Cumplimiento de reglas después de la implementación del sistema

Ítem	Implementación de productos	Cumplimiento de Reglas (80 % - 100 %)
1	Producto1	75%
2	Producto2	70%
3	Producto3	80%
4	Producto4	81%
5	Producto5	68%
6	Producto6	80%
7	Producto7	85%
8	Producto8	90%
9	Producto9	81%
10	Producto10	83%
		Reglas(PMD)

Fuente: Elaboración propia**.**

Tabla 39: Interdependencia de paquetes antes de la implementación del sistema

Ítem	Implementación de productos	Índice de interdependencia entre paquetes
1	Producto1	6.1%
2	Producto2	5.8%
3	Producto3	2.5%
4	Producto4	2.1%
5	Producto5	6.3%
6	Producto6	1.9%
7	Producto7	1.5%
8	Producto8	0.5%
9	Producto9	6.5%
10	Producto10	0.9%
		dependencias

Fuente: Elaboración propia

Tabla 40: Cumplimiento de reglas después de la implementación del sistema

Ítem	Implementación de productos	Índice de Interdependencia entre paquetes (0.0 %)
1	Producto1	2.1%
2	Producto2	2.0%
3	Producto3	1.2%
4	Producto4	1.0%
5	Producto5	2.8%
6	Producto6	0.0%
7	Producto7	0.0%
8	Producto8	0.0%
9	Producto9	1.0%
10	Producto10	0.0%
		dependencias

Fuente: Elaboración propia.

Tabla 41: código duplicado antes de la implementación del sistema

Ítem	Implementación de productos	Duplicados (0 % – 10 %)
1	Producto1	17%
2	Producto2	19%
3	Producto3	18%
4	Producto4	11%
5	Producto5	15%
6	Producto6	13%
7	Producto7	10%
8	Producto8	10%
9	Producto9	20%
10	Producto10	12%
		Duplicado

Fuente: Elaboración propia

Tabla 42: Código duplicado después de la implementación del sistema

Ítem	Implementación de productos	Duplicados (0 % – 10 %)
1	Producto1	11%
2	Producto2	10%
3	Producto3	10%
4	Producto4	9%
5	Producto5	12%
6	Producto6	8%
7	Producto7	6%
8	Producto8	8%
9	Producto9	10%
10	Producto10	5%
		Duplicado

Fuente: Elaboración propia.

Tabla 43: Complejidad por método antes de la implementación del sistema

Ítem	Implementación de productos	Complejidad por Método(1 – 5)
1	Producto1	12%
2	Producto2	13%
3	Producto3	7%
4	Producto4	7%
5	Producto5	9%
6	Producto6	10%
7	Producto7	6%
8	Producto8	5%
9	Producto9	20%
10	Producto10	7%
		Flujo

Fuente: Elaboración propia

Tabla 44: Complejidad por método después de la implementación del sistema

Ítem	Implementación de productos	Complejidad por Método(1 – 5)
1	Producto1	6%
2	Producto2	7%
3	Producto3	6%
4	Producto4	5%
5	Producto5	8%
6	Producto6	5%
7	Producto7	5%
8	Producto8	4%
9	Producto9	5%
10	Producto10	4%
		Flujo

Fuente: Elaboración propia.

Tabla 45: LCOM4 antes de la implementación del sistema

Ítem	Implementación de productos	LCOM4 (1.0/clase)
1	Producto1	3.0%
2	Producto2	2.8%
3	Producto3	2.0%
4	Producto4	1.9%
5	Producto5	2.4%
6	Producto6	2.1%
7	Producto7	1.3%
8	Producto8	1.1%
9	Producto9	3.0%
10	Producto10	1.3%
		cmp/conect

Fuente: Elaboración propia

Tabla 46: LCOM4 después de la implementación del sistema

Ítem	Implementación de productos	LCOM4 (1.0/clase)
1	Producto1	1.3%
2	Producto2	1.3%
3	Producto3	1.0%
4	Producto4	1.0%
5	Producto5	1.5%
6	Producto6	1.0%
7	Producto7	1.0%
8	Producto8	1.0%
9	Producto9	1.2%
10	Producto10	1.0%
		cmp/conect

Fuente: Elaboración propia.

Tabla 47: Comentarios del código antes de la implementación del sistema

Ítem	Implementación de productos	Comentarios (10% - 20 %)
1	Producto1	5%
2	Producto2	5%
3	Producto3	7%
4	Producto4	11%
5	Producto5	9%
6	Producto6	10%
7	Producto7	11%
8	Producto8	10%
9	Producto9	8%
10	Producto10	9%

Fuente: Elaboración propia

Tabla 48: comentarios del código después de la implementación del sistema

Ítem	Implementación de productos	Comentarios (10% - 20 %)
1	Producto1	18%
2	Producto2	18%
3	Producto3	17%
4	Producto4	12%
5	Producto5	11%
6	Producto6	15%
7	Producto7	13%
8	Producto8	14%
9	Producto9	15%
10	Producto10	14%

Fuente: Elaboración propia.

Anexo 4: Validación de instrumentos Juicio Experto

UCV UNIVERSIDAD CESAR VALLEJO

sube

INFORME DE VALIDACIÓN DE INSTRUMENTOS A TRAVÉS DE JUICIO DE EXPERTOS

Datos Generales

1.1 Apellidos y nombres del validador:

Mario Verastegui Wilson Ricardo

1.2 Institución donde labora/cargo:

Docente a tiempo completo

1.3 Especialidad del validador:

Ingeniería de sistemas

1.4 Nombre del Instrumento y finalidad de su aplicación:

Ficha de observación: en la presente investigación se usó para registrar la productividad, el costo laboral, la remuneración promedio y la calidad de los productos de software de la empresa Novatronic S.A.C.

Nombre: Fiche de registro de las metricas del proceso de desarrollo de software

1.5 Título de la investigación:

Sistema de encriptación para optimizar el proceso de cesarrollo de software de la empresa Novatronic s.a.c., 2016

1.6 Autor del Instrumento:

Josue Wenceslao Ríos Taipe

UCV UNIVERSIDAD CÉSAR VALLEJO

sube

IV. Certificado de validez de contenido del instrumento

Nº	DIMENSIONES / indicadores	Pertinencia[1]		Relevancia[2]		Claridad[3]		Sugerencias
		Si	No	Si	No	Si	No	
	DIMENSIÓN 1							
1	Productividad	✓		✓		✓		
2								
3								
		Si	No	Si	No	Si	No	
	DIMENSIÓN 2							
1	Calidad del software	✓		✓		✓		
2								
3								
		Si	No	Si	No	Si	No	
	DIMENSIÓN 3							
1								
2								

Observaciones (precisar si hay suficiencia): ____________________

Opinión de aplicabilidad: Aplicable [X] Aplicable después de corregir [ ] No aplicable [ ]

Apellidos y nombres del juez validador. Dr/ Mg: Hario Verastegui Wilson Ricardo DNI: 45801046

Especialidad del validador: Mgtr. Gestión de tecnologías de información

26 de 06 del 2017.

[1]Pertinencia: El indicador corresponde al concepto teórico formulado.
[2]Relevancia: El indicador es apropiado para representar al componente o dimensión específica del constructo
[3]Claridad: Se entiende sin dificultad alguna, es conciso, exacto y directo

Nota: Suficiencia, se dice suficiencia cuando los indicadores planteados son suficientes para medir la dimensión

Firma del Experto Informante.

5

UCV UNIVERSIDAD CÉSAR VALLEJO

sube

INFORME DE VALIDACIÓN DE INSTRUMENTOS A TRAVÉS DE JUICIO DE EXPERTOS

I. **Datos Generales**

1.1 Apellidos y nombres del validador:

Gálvez Tapia Orlean Moisés

1.2 Institución donde labora/cargo:

Docente a tiempo completo

1.3 Especialidad del validador:

Ingeniería de sistemas

1.4 Nombre del instrumento y finalidad de su aplicación:

Ficha de observación: en la presente investigación se usó para registrar la productividad, el costo laboral, la remuneración promedio y la calidad de los productos de software de la empresa Novatronic S.A.C.

Nombre: Fiche de registro de las métricas del proceso de desarrollo de software

1.5 Título de la investigación:

Sistema de encriptación para optimizar el proceso de desarrollo de software de la empresa Novatronic s.a.c., 2016

1.6 Autor del instrumento:

Josue Wenceslao Ríos Taipe

UCV UNIVERSIDAD CÉSAR VALLEJO

sube

IV.Certificado de validez de contenido del instrumento

Nº	DIMENSIONES / indicadores	Pertinencia[1]		Relevancia[2]		Claridad[3]		Sugerencias
	DIMENSIÓN 1	Si	No	Si	No	Si	No	
1	Productividad	✓		✓		✓		
2								
3								
	DIMENSIÓN 2	Si	No	Si	No	Si	No	
1	Calidad del Software	✓		✓		✓		
2								
3								
	DIMENSIÓN 3	Si	No	Si	No	Si	No	
1								
2								

Observaciones (precisar si hay suficiencia): _____

Opinión de aplicabilidad: Aplicable [X] Aplicable después de corregir [ ] No aplicable [ ]

Apellidos y nombres del juez validador. Dr/ Mg: Mg. Gálvez Tapia Orfeus Nonel DNI: 16708332

Especialidad del validador: Mg. en Ingeniería de Sistemas

26 de 06 del 2017

[1]Pertinencia:El indicador corresponde al concepto teórico formulado.
[2]Relevancia: El indicador es apropiado para representar al componente o dimensión específica del constructo
[3]Claridad: Se entiende sin dificultad alguna, es conciso, exacto y directo

Nota: Suficiencia, se dice suficiencia cuando los indicadores planteados son suficientes para medir la dimensión

Firma del Experto Informante.

UCV UNIVERSIDAD CÉSAR VALLEJO

sube

INFORME DE VALIDACIÓN DE INSTRUMENTOS A TRAVÉS DE JUICIO DE EXPERTOS

I. **Datos Generales**

1.1 Apellidos y nombres del validador:

Pérez Rojas Evert Alejos.

1.2 Institución donde labora/cargo:

Docente a tiempo completo

1.3 Especialidad del validador:

Ingeniería de sistemas

1.4 Nombre del instrumento y finalidad de su aplicacion:

Ficha de observación: en la presente investigacion se uso para registrar la productividad, costo laboral, la remuneración promedio y la calidad de los productos de software de la empresa Novatronic S.A.C.

Nombre: Fiche de registro de las métricas del proceso de desarrollo de software

1.5 Titulo de la investigación:

Sistema de encriptación para optimizar el proceso de desarrollo de software de la empre Novatronic s.a.c., 2016

1.6 Autor del Instrumento:

Josue Wenceslao Rios Taipe

IV. Certificado de validez de contenido del instrumento

Nº	DIMENSIONES / Indicadores	Pertinencia[1]		Relevancia[2]		Claridad[3]		Sugerencias
	DIMENSIÓN 1	Si	No	Si	No	Si	No	
1	Productividad	✓		✓		✓		
2								
3								
	DIMENSIÓN 2	Si	No	Si	No	Si	No	
1	Calidad del software	✓		✓		✓		
2								
3								
	DIMENSIÓN 3	Si	No	Si	No	Si	No	
1		✓		✓		✓		
2								

Observaciones (precisar si hay suficiencia): SI HAY SUFICIENCIA

Opinión de aplicabilidad: Aplicable [X] Aplicable después de corregir [ ] No aplicable [ ]

Apellidos y nombres del juez validador. Dr/ Mg: Pérez Rojas Ever Elysl. **DNI:** 43716341

Especialidad del validador: Tecnología de la Información

26 de 06 del 2012

[1]**Pertinencia:** El indicador corresponde al concepto teórico formulado.
[2]**Relevancia:** El indicador es apropiado para representar al componente o dimensión específica del constructo
[3]**Claridad:** Se entiende sin dificultad alguna, es conciso, exacto y directo

Nota: Suficiencia, se dice suficiencia cuando los indicadores planteados son suficientes para medir la dimensión

Firma del Experto Informante.

ESP. 15513

Anexo 5: Modelamiento de la solución del software

1. Plan de Desarrollo

El Proyecto tiene como propósito la implementación de un componente de software para el cifrado y firmado de información, esta aplicación se podrá implementar como un componente de software o como un servicio de una plataforma transaccional, dando como resultado de su implementación mejoras sobre en la producción de software, así como mejora en la rentabilidad de los productos debido al incremento de las características de seguridad del mismo.

La metodología aplicada es de propia de la empresa, y la secuencia metodológica se desarrolla a continuación:

1.1 Vista General del Proyecto

1.1.1 Propósito

Satisfacer las necesidades del área de desarrollo, en relación a la necesidad de un componente de seguridad, que permita proteger la información de los usuarios finales que utilicen un determinado producto de software de una empresa de Lima.

1.1.2 Alcance

El presente estudio está delimitado al área de desarrollo de software y se tomó como muestra al equipo de desarrollo de la empresa, por ser los que contribuyen en una su totalidad al cumplimiento de los objetivos del área y del proyecto.

1.1.3 Objetivos

- Mejorar el tiempo que toma el desarrollo de nuevos productos de software, mediante la reutilización del componente de cifrado y firmado de información.

- Incrementar la rentabilidad de los productos de software, como consecuencia de incrementar las características de seguridad de los mismos.

- Mejorar el nivel de seguridad mediante la protección de la información, por medio del cifrado y firmado de estas.

1.1.4 Suposiciones

- Suponemos que los estadales en los cuales nos basamos para el diseño del software no cambiaran en corto plazo.

- Cuenta con una buena infraestructura tecnológica.

- Los ingresos y egresos económicos del área comercial están con tendencia estable.

- El tipo de cambio del $ dólar se mantiene.

- El desarrollador tiene conocimientos sobre mecanismos de seguridad.

- Las estrategias, objetivos y metas se obtuvieron del plan estratégico de una empresa de Lima

1.1.5 Restricciones

- Se consideran dentro los mecanismos de seguridad sólo el soporte a PCI y EMV, para sistemas transaccionales de pagos.

- No se considera equipo de desarrollos virtuales dentro del proyecto.

- No se considera una arquitectura con tolerancia a los fallos.

- Se han calculado los beneficios del proyecto en los costos de desarrollo de los productos y el incremento de ventas de los mismos.

- El proyecto no considera un sistema de administración de componentes.

- La capacidad y el tiempo de análisis son limitados y lentos dentro de la cultura de la empresa así como el tiempo dedicado a la planeación.

- En el diseño del artefacto se llegó hasta la última etapa de la metodología RUP (transición) pero no se desarrolló manuales de instalación ni material de apoyo al usuario debido que para el desarrollo de este proyecto no aplica.

1.2 Entregables del Proyecto

A continuación se indican y describen cada uno de los artefactos que serán generados y utilizados por el proyecto y que constituyen los entregables. Para un buen seguimiento del desarrollo del producto final (El nuevo componente), se le brinda a la empresa el plan de trabajo del proyecto, que contiene como hitos los entregables del proyecto que se mencionan a continuación.

1.2.1 Plan de Desarrollo del Software

Es el presente documento.

1.2.2 Modelo de Casos de Uso del Negocio

Es un modelo que presenta las distintas funciones de componente en relación al negocio bajo la perspectiva de los actores externos (Adquirentes, sistemas externos, etc.). Permite situar al sistema en el contexto del negocio haciendo énfasis en los objetivos del implementador. Este modelo se representa con un Diagrama de Casos de Uso usando estereotipos específicos para este modelo.

1.2.3 Modelo de Objetos del Negocio

Para la representación de este modelo se utilizan Diagramas de Colaboración (para mostrar factores externos, internos y las entidades (información), un Diagrama de Clases para mostrar gráficamente las entidades del sistema y sus relaciones, y Diagramas de Actividad para mostrar los flujos de trabajo

Este modelo describe los casos de uso de negocio, estableciendo los actores internos, así como los flujos de trabajo (workflows) asociados al caso de uso del mismo.

1.2.4 Glosario

Es un documento que define los principales términos usados en el proyecto. Permite establecer una terminología consensuada.

1.2.5 Modelo de Casos de Uso

El modelo de Casos de Uso presenta las funciones del sistema y los actores que hacen uso de ellas. Se representa mediante Diagramas de Casos de Uso.

1.2.6 Visión

Este documento define la visión del servicio desde la perspectiva del área de desarrollo, especificando las necesidades y características del servicio o producto.

1.2.7 Especificaciones de Casos de Uso

Se realiza una descripción detallada utilizando una plantilla de documento, donde se describirá todos los flujos de negocio, tanto el principal como los alternos, estas incluyen: precondiciones, post-condiciones, flujo de eventos, requisitos funcionales y no funcionales asociados. También, para casos de uso cuyo flujo de eventos sea

complejo podrá adjuntarse una representación gráfica mediante un Diagrama de Actividad.

1.2.8 Interfaces de clases o servicios

Se trata de interfaces que permitirán el uso de la aplicación, ya sea como servicios o como un componente de un producto, cada uno orientado a los estándares para el desarrollo de software o para el uso de servicios.

1.2.9 Modelo de Análisis y Diseño

Este modelo representa mediante diagramas, gráficos, casos de uso la arquitectura, funcionalidad y el diseño del software que se va a implementar, este puede cambiar dependiendo de la etapa en la cual se encuentre el proyecto

1.2.10 Modelo de Datos

Este modelo describe la representación lógica de los datos persistentes. Para expresar este modelo se utiliza un Diagrama de Clases (donde se utiliza un profile UML para Modelado de Datos, para conseguir la representación de tablas, claves, etc.).

1.2.11 Modelo de Implementación

Este modelo es una colección de componentes que incluyen: ficheros ejecutables, ficheros de código fuente, y todo otro tipo de ficheros necesarios para la implantación y despliegue del sistema. (Este modelo es una representación del entregable final del proyecto).

1.2.12 Modelo de Despliegue

Este modelo muestra el despliegue de la configuración de tipos de nodos del sistema, en los cuales se hará el despliegue de los componentes.

1.2.13 Casos de Prueba

Cada prueba es especificada mediante un documento que establece las condiciones de ejecución, las entradas de la prueba, y los resultados esperados. Estos casos de prueba son aplicados como pruebas de regresión en cada iteración. Cada caso de prueba llevará asociado un procedimiento de prueba con las instrucciones para realizarla, y dependiendo del tipo de prueba dicho procedimiento podrá ser automatizable mediante un script de prueba.

1.2.14 Manual de Programación

Este documento específica los pasos a seguir para el uso del componente.

1.2.15 Manual de instalación

Este documento específica los pasos a seguir para desplegar el componente como servicio.

1.3.15 Material de Apoyo al Usuario Final

No aplica para este tipo de solución.

1.3 Organización del Proyecto

Roles y Responsabilidades

A continuación se describen las responsabilidades de cada uno de los puestos en el equipo de desarrollo durante las fases de Inicio y Elaboración, de acuerdo con los roles que desempeñan en RUP.

Tabla 49: Cuadro de Roles y Responsabilidades

Puesto	responsabilidad
Jefe de proyecto	Es el responsable principal del proyecto, cuya función es la gestión, planificación, monitoreo y seguimiento de las actividades del proyecto. Este vela porque los objetivos y el alcance del proyecto se cumplan.
Programador	Es la persona encargada de codificar lo funcionalidad definida en el documento de análisis y diseño del sistema
Analista	Es el responsable de representar mediante casos de uso, diagramas y gráficos la funcionalidad de la aplicación
Arquitecto	Es el responsable de definir las bases sobre la cual se diseñará la aplicación, este define el patrón de diseño y la funcionalidad base de la solución.

Fuente: Elaboración propia

1.4 Gestión del Proceso

1.4.1 Estimaciones del Proyecto

Para el desarrollo del proyecto se necesita recurrir a los siguientes recursos

Tabla 50: Lista de Recursos Humanos para el desarrollo del proyecto.

Nombre	Tiempo de Trabajo	Tasa estándar	Costo	Capacidad
Jefe de Proyectos	240 horas	S/. 35.00	S/. 8,400.00	1
Consultor Marketing	32 horas	S/. 10.00	S/. 320.00	1
Administrador de Redes y Comunicaciones	16 horas	S/. 25.00	S/. 400.00	1
Analista de Soporte	32 horas	S/. 10.00	S/. 320.00	1
Analistas de Calidad	320 horas	S/. 15.00	S/. 9,600.00	2
Analista Funcional	160 horas	S/. 15.00	S/. 2,400.00	1
Analista programador	720 horas	S/. 20.00	S/. 28,800.00	2
Sponsor	80 horas	S/. 50.00	S/. 4,000.00	1
Total	1600 horas	S/. 180.00	S/. 54,240.00	10

Fuente: Elaboración propia.

Tabla 51: Lista de Hardware y Software a utilizar en el desarrollo del proyecto.

Concepto	Descripción	Cantidad	Monto S/.
Hardware	Servidor de Base de Datos	1	S/. 2,500.00
	Servidor para desarrollo	1	S/. 2,500.00
	Servidor para Plataforma	1	S/. 2,500.00
	Estaciones de Trabajo	10	S/. 660.00
	Sub Total Hardware		S/. 8,160.00
Software	Aplicación Web	1	S/. 0.00
	Base de Datos (PostgreSql)	1	S/. 0.00
	Licencias de BD(PostgreSql)	1	S/. 0.00
	Licencias de SO (Linux RedHat)	10	S/. 0.00
	Herramientas de Análisis (Eclipse, java)	1	S/. 0.00
	Antivirus	10	S/. 0.00
	Sub Total Software		S/. 0.00
Capacitación	Desarrollo de Capacitación	-	S/. 200.00
	Materiales	-	S/. 100.00
	Sub Total Capacitación		S/. 300.00
Otros	Útiles de Escritorio	-	S/. 100.00
	Varios	-	S/. 100.00
	Sub Total Otros		S/. 200.00
	Total de Inversión		S/. 8,660.00

Fuente: Elaboración propia.

Tabla 52: Resumen de Inversión Total

Costo Total del Proyecto	**S/.**
Inversión Inicial en Software y Hardware	S/. 8,660.00
Inversión en Recursos	S/. 54,240.00
Total	S/.62,900.00

Fuente: Elaboración propia.

1.4.2 Plan de Proyecto

En esta sección se presenta la organización en fases e iteraciones y el calendario del proyecto.

El desarrollo se llevará a cabo en base a fases con una o más iteraciones en cada una de ellas. La siguiente tabla muestra la distribución de tiempos y el número de iteraciones de cada fase (para las fases de Construcción y Transición es sólo una aproximación muy preliminar).

Tabla 53 : Cuadro de Iteraciones.

ITERACIONES	
1	Interfaces aplicativa como componente o servicio
2	Funciones de cifrado y Firmado
3	Funciones de firma y validación
4	Funciones de carga de llaves y configuración
5	Funciones de administración y uso de algoritmos.

Fuente: Elaboración propia

Tabla 54: Cuadro de Fases

Fase	Numero de iteración	Duración
Inicio	3	04/01 - 06/02
Elaboración	4	20/01 - 25/02
Contrición	4	11/02 - 25/05
transición	1	29/05 15/06

Tabla 55: Cuadro de Hitos (descripción del final de cada fase)

Descripción	Hito
Fase inicio	Esta fase tiene como propósito definir y acordar el alcance del proyecto con los patrocinadores, identificar los riesgos asociados al proyecto, proponer una visión muy general de la arquitectura de software y producir el plan de las fases y el de iteraciones posteriores Como resultado de este punto se generaran todos los entregables necesarios para poder llevar a cabo el desarrollo del sistema, y se entrega el plan de trabajo del proyecto
Fase de Elaboración	En la fase de elaboración se seleccionan los casos de uso que permiten definir la arquitectura base del sistema y que se desarrollaran en esta fase, se realiza la especificación de los casos de uso seleccionados y el primer análisis del dominio del problema, se diseña la solución preliminar. Como resultado de esta fase, se generan los diagramas, casos de uso y toda información necesaria para definir la arquitectura y las funcionalidades del sistema
Fase de Construcción	El propósito de esta fase es completar la funcionalidad del sistema, para ello se deben clarificar los requisitos pendientes, administrar los cambios de acuerdo a las evaluaciones realizados por los usuarios y se realizan las mejoras para el proyecto. En esta fase se desarrollará el software, y como parte de ello se realizarán las iteraciones de las fases anteriores de las mismas.
Fase de Transición	El propósito de esta fase es asegurar que el software esté disponible para los usuarios finales, ajustar los errores y defectos encontrados en las pruebas de aceptación, capacitar a los usuarios y proveer el soporte técnico necesario. Se debe verificar que el producto cumpla con las especificaciones entregadas por las personas involucradas en el proyecto. en esta fase se entregarán los entregables finales del proyecto y se dará el soporte y seguimiento pos implantación

Fuente: Elaboración propia

1.4.3 Seguimiento y Control del Proyecto

El propósito de esta fase es asegurar que cumpla el alcance y los objetivos del proyecto. Para ello se revisa el plan de trabajo y se realiza el monitoreo y seguimiento de cada una de las actividades, para asegurar el cumplimento de ellas, como parte de esto también

se realizar el análisis de los riegos para evitar imprevistos que afecten el proyecto.

En un proyecto que usa la metodología de Rup, los artefactos son generados muy tempranamente, pero van desarrollándose en mayor o menor grado de acuerdo a la fase e iteración del proyecto, esto de acuerdo a lo planificado para cada etapa.

A continuación se presenta un calendario de las tareas del proyecto incluyendo las fases de Inicio, Elaboración, Construcción y Pruebas.

Tabla 56 : Cuadro de Gantt

Nombre de tarea	Duración	Comienzo	Fin
Sistema de encriptación	0 días	lun 07/09/16	lun 07/09/16
Inicio	**89 días**	**lun 07/09/16**	**jue 07/01/16**
Fase I: Iniciación	**32 días**	**lun 07/09/16**	**mar 20/10/16**
Elaboración del Project Charter	**29 días**	**lun 07/09/16**	**jue 15/10/16**
Definición del Alcance	2 días	lun 07/09/16	mar 08/09/16
Levantamiento de Información Inicial	14 días	mar 08/09/16	vie 25/09/16
Reunión Inicial con el área (Comercial)	3 días	mié 09/09/16	vie 11/09/16
Presentación de Presupuesto	5 días	mar 29/09/16	lun 05/10/16
Formación de Equipos (Stakeholders)	3 días	mar 06/10/16	jue 08/10/16
Planteamiento de Presupuesto	4 días	vie 09/10/16	mié 14/10/16
Elaboración de Presentación de Kickoff	1 día	jue 15/10/16	jue 15/10/16
Presentación de Lanzamiento del Proyecto - Kickoff Meeting	**1 día**	**mar 20/10/16**	**mar 20/10/16**
Presentación	1 día	mar 20/10/16	mar 20/10/16
Fase II: Requerimientos	**3 días**	**mié 21/10/16**	**vie 23/10/16**
Requerimientos de Hardware y Software	1 día	mié 21/10/16	mié 21/10/16
Requerimiento de Aplicativo	1 día	jue 22/10/16	jue 22/10/16
Requerimiento de Proceso	1 día	vie 23/10/16	vie 23/10/16
Fase III: Ejecución	**51 días**	**lun 26/10/16**	**lun 04/01/16**
Análisis y Diseño	**8 días**	**lun 26/10/16**	**mié 04/11/16**
Modelos de Casos de uso	1 día	lun 26/10/16	lun 26/10/16
Diagramas de Actividades	1 día	mar 27/10/16	mar 27/10/16
Diagramas de Estado	1 día	mié 28/10/16	mié 28/10/16
Diagramas de Secuencia	1 día	jue 29/10/16	jue 29/10/16
Diagramas de Colaboración	1 día	jue 29/10/16	jue 29/10/16
Diagramas de Clases	1 día	vie 90/10/16	vie 90/10/16
Diagramas Componentes	1 día	lun 02/11/16	lun 02/11/16
Diagramas de Paquetes	1 día	mar 03/11/16	mar 03/11/16
Diagramas de Despliegue	1 día	mié 04/11/16	mié 04/11/16
Programación	**24 días**	**jue 05/11/16**	**mar 08/12/16**
Sistema de Programación	3 días	jue 05/11/16	lun 09/11/16
Script de Programación	21 días	mar 10/11/16	mar 08/12/16
Pruebas	**19 días**	**mié 09/12/16**	**lun 04/01/16**
Pruebas Unitarias	**5 días**	**mié 09/12/16**	**mar 15/12/16**
Pruebas del aplicativo	5 días	mié 09/12/16	mar 15/12/16
Pruebas de validación de funcionalidad del sistema	5 días	mié 09/12/16	mar 15/12/16
Pruebas de sincronización	5 días	mié 09/12/16	mar 15/12/16
Pruebas Usuarias	**14 días**	**mié 16/12/16**	**lun 04/01/16**
Validación de carga de clientes y artículos	2 días	mié 16/12/16	jue 17/12/16
Validación de carga de transporte y pedidos	2 días	mié 16/12/16	jue 17/12/16
Pruebas funcionales	3 días	jue 17/12/16	lun 21/12/16
Aceptación funcional	3 días	mar 22/12/16	jue 24/12/16
Capacitación	2 días	vie 25/12/16	lun 28/12/16
Implementación (Pruebas en producción)	5 días	mar 29/12/16	lun 04/01/16
Fase IV: Cierre	**3 días**	**mar 05/01/16**	**jue 07/01/16**
Acta de Aceptación de entregables por parte del cliente	**3 días**	**mar 05/01/16**	**jue 07/01/16**
Acta de reuniones	1 día	mar 05/01/16	mar 05/01/16
Lecciones Aprendidas	1 día	mié 06/01/16	mié 06/01/16
Acta de cierre	1 día	jue 07/01/16	jue 07/01/16
FIN	0 días	jue 07/01/16	jue 07/01/16

Fuente: Elaboración propia.

2. Documento Visión

2.1 Posicionamiento

2.1.1 Oportunidad de Negocio

En la actualidad las empresas exigen que los sistemas de transacciones tengan mecanismos de seguridad que permitan proteger la información que viaja por estos sistemas, estas también exigen que se cumplan con los estándares tales como PCI y EMV. En tal sentido la implementación del componente de cifrado y firmado de información, permitirá que los productos de la empresa cuenten con mecanismos de seguridad que contemplen los estándares que exige el mercado, esto los volverá productos más competitivos y más atractivos para el mercado y por ende incrementará el número de venta de los mismos.

2.1.2 Sentencia que define el problema

Tabla 57 : Cuadro que define el problema

El problema de	Cada vez que se desarrolló un producto de software transaccional, se contempla las funciones de seguridad, pero estas no cubren, ni cumplen con lo que exige el mercado, porque son desarrolladas de acuerdo a como lo considera el implementador, por ello actualmente se cuenta con productos con características de seguridad nulas u obsoletas, cabe mencionar que el implementar estos mecanismos en cada proyecto, toma mucho tiempo y es muy costoso.
Afecta	Gerencia de operaciones, gerencia de desarrollo de software
El impacto de lo cual es	Se elimina los desarrollos repetitivos de las funciones de seguridad y se incrementara el nivel de seguridad de los productos antiguos y nuevos.
La solución exitosa podría ser	La implementación de unos componentes que tengan toda lógica de seguridad necesaria para implementación de distintos productos, estas mejorarán los tiempos de desarrollo, y los costos de implementación.

Fuente: Elaboración propia

2.1.3 Sentencia que define la posición del producto

Tabla 58: Cuadro que define la posición del producto.

Para	Gerencial Comercial: Gerente de Ventas Vendedores Gerente de desarrollo Jefes de productos Jefes de proyectos
Quienes	Comerciales Controlan y Supervisan los pedidos como parte del proceso de ventas. Desarrollo Implementan y actualizan los productos de software de la empresa, tales como sistemas transaccionales.
El nombre del producto	Componente “Encript-TRX”.
Que	Expone servicios o interfaces que permiten realizar el cifrado y descifrado de todo tipo de meta data, así como la firma y validación de meta data.
No como	La forma en la cual se han venido realizando los desarrollos, ahora tendrán un componente que cuenta con toda la funcionalidad de seguridad que se requiere para generación u actualización de los productos de software transaccional.
Nuestro producto	Permite implementar mecanismos de seguridad de forma rápida y ágil, estos mecanismos protegerán tanto la información del producto, como la información del cliente final, estos también soportarán los estándares de seguridad tales como PCI y EMV.

Fuente: Elaboración propia

Para generar un software que cumpla con lo que requiere el mercado, es necesario identificar a todos los interesados del proyecto, estos participarán en todo el ciclo de vida del proyecto, para ello se generan reuniones de seguimiento y facilitación, entre los internados estarán los clientes, los proveedores, el área de operaciones.

Esta sección muestra los distintos participantes del proyecto, así como sus roles o funciones.

2.1.4 Resumen de Stakeholders

Tabla 59: Resumen Stakeholders.

Cargos	QUE REPRESENTA	ROL
Sponsors	Representante interno de nuestros cliente	Máximo interesado del proyecto, busca satisfacer las necesidades de los clientes
Gerentes	Gerentes y Jefes de áreas, representan las necesidades de la empresa en realización a los productos	Disminuir el tiempo y costo de desarrollo de los productos

Fuente: Elaboración propia

2.1.5 Resumen de Usuarios

Tabla 60: Resumen Usuarios.

Cargos	QUE REPRESENTA	ROL
Directores	Analiza los reportes de los pedidos.	Toma de decisiones.
Comerciales	Representan la fuerza de venta de la empresa	Realiza la gestión de las ventas
desarrolladores	Implementar u actualizar nuevos productos	Desarrollan nuevos productos

Fuente: Elaboración propia

2.1.6 Interfaces de los componentes

- El sistema fue diseñado para ser implementado como un componente que forma parte de un producto.
- El sistema fue diseñado para ser utilizado como un servicio que forme parte de una plataforma transaccional segura.
- Se proveen funcionalidad de encriptado y desencriptado de información, así como firma digital y validación.

2.2 Descripción Global del Producto

2.2.1 Perspectiva del Producto

El producto a desarrollar es un componente de cifrado y firmado de información, este contará con mecanismos de seguridad que permitan cifrar la información bajo distintos tipos de algoritmos y métodos de cifrado, ya sea con claves públicas o privadas.

El componente de seguridad contará con las siguientes funcionalidades:

- Interface de clases para uso del componente.
- Interface de servicios
- Funciones de encriptación y desencriptación
- Funciones de Firma y validación.
- Mecanismo de algoritmos simétricos y asimétricos
- Almacenamiento de claves públicas y privadas

2.2.2 Resumen de Características

Encriptación y desencriptación

Los desarrolladores, adquirentes o sistemas externos que formen parte de una plataforma transaccional, podrán acceder a interfaces que les permitan proteger la información de imágenes, tramas, textos, meta data, mediante funciones de encriptación y desencriptación

Firma y validación

Los desarrolladores, adquirentes o sistemas externos que formen parte de una plataforma transaccional, podrán acceder a interfaces que le permitan proteger la información y asegurar la integridad de imágenes, tramas, textos, meta data, mediante funciones de firma y validación digital

Interfaces de servicio y componente

Los desarrolladores, adquirentes o sistemas externos que formen parte de una plataforma transaccional, podrán utilizar el mecanismo de seguridad consumiendo servicios o interfaces expuestas por el componente de seguridad

2.3 Descripción de Procesos del Área

2.3.1 Gestión de proyectos

El propósito es mejorar el tiempo de implementación de los proyectos, reduciendo el tiempo de análisis y desarrollo del mismo, permitiendo que el jefe de proyecto tenga una visión clara de lo que va a implementar a nivel de la seguridad de la información

2.3.2 Desarrollo de software

El propósito, es disminuir el tiempo de producción de software, así como los costos, mediante la fácil implementación de los mecanismos de seguridad, guiados por la documentación y las interfaces expuestas del nuevo componente.

2.3.3 Análisis y diseño

El propositito es disminuir el tiempo de análisis de la solución integral, debido a que no tendrá que realizar el análisis de los mecanismos de seguridad.

2.3.4 Venta de productos

El propósito es incrementar las ventas, mostrando a nuestros clientes que nuestros productos son seguros y soportan los estándares de seguridad que exigen para las plataformas transaccionales.

2.4 Especificaciones de Requisitos de Software

2.4.1 Especificaciones

2.4.1.1 Especificaciones Funcionales

- La aplicación deberá ser multi intererface
- La aplicación deberá soportar distintos tipos de algoritmos
- La aplicación deberá poder utilizar claves públicas y privadas
- La aplicación deberá exponer servicios para el cifrado y descifrado de información
- La aplicación deberá permitir cifrar cualquier tipo de información, tanto texto como imágenes, etc.
- La aplicación deberá exponer funcionalidades para la firma y validación digital
- La aplicación deberá permitir ejecutar las funciones criptográficas sobre archivos y sobre tramas.
- La aplicación deberá ser parametrizable y configurable.
- La aplicación deberá exponer interfaces para el cifrado por hardware.

2.4.1.2 Especificaciones No Funcionales

- Deberá contar con un log configurable.
- Deberá ser configurable a nivel de servicios
- Tiempos de respuesta menores a 50 milisegundos, con carga.
- Calidad del sistema.

2.4.2 Modelamiento

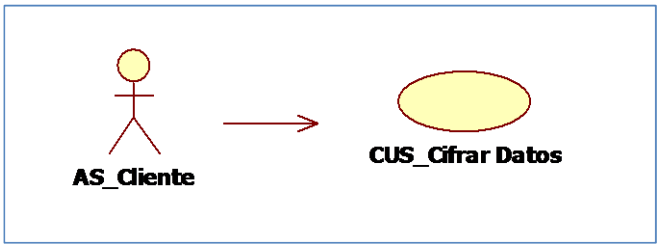
2.4.2.1 Actores de Caso de Uso

Figura 43: Actores de Caso de Uso

Fuente: Elaboración propia

2.4.2.2 Diagrama de Casos de Uso

Figura 44: CUS_Cifrar Mensaje



Fuente: Elaboración propia

2.4.2.3 Especificaciones de Casos de Uso (CUS_Cifrar Mensaje)

1. Actores del Sistema

AS_Cliente

2. Propósito

El caso de uso tiene como propósito cifrar un mensaje que el cliente provea; es decir, cifrar, descifrar, firmar y verificar su firma del mensaje solicitado por el cliente.

3. Breve Descripción

El caso de uso se inicia cuando el cliente solicita cifrar un mensaje. Si el actor solicita el cifrado o descifrado del mensaje, el sistema valida el algoritmo simétrico y lee la llave simétrica. El sistema realiza el cifrado o descifrado del mensaje mediante el algoritmo y llave simétrica. El sistema devuelve el mensaje cifrado o descifrado. Caso contrario, si el actor solicita la firma del mensaje, el sistema valida el algoritmo asimétrico y lee la llave privada. Luego, el sistema realiza la firma del mensaje y devuelve el mensaje cifrado. Para la verificación de la firma, el sistema verifica el mensaje original y el firmado mediante la llave pública.

Flujo de Eventos

3.1. Flujo Básico

3.1.1. El caso de uso se inicia cuando el actor solicita cifrar un mensaje.

3.1.2. El sistema valido el algoritmo, operación y contenido del mensaje

3.1.3. El sistema carga la configuración para el tipo de cifrado.

3.1.4. El sistema ejecuta la operación de acuerdo:

3.1.4.1. Si la operación es Cifrar, ver Sub Flujo “Cifrar”.

3.1.4.2. Si la operación es Descifrar, ver Sub Flujo “Descifrar”.

3.1.4.3. Si la operación es Firmar, ver Sub Flujo “Firmar”.

3.1.4.4. Si la operación es Verificar la Firma, ver Sub Flujo “Verificar Firma”.

3.1.5. El sistema muestra el código de respuesta “00” con el mensaje “Respuesta obtenida” en los registros del sistema.

3.1.6. El sistema retorna el código, mensaje y contenido de la respuesta de la operación y el caso de uso finaliza

3.2. Sub Flujos

3.2.1. Cifrar

3.2.1.1. El sistema valida la configuración del cifrado: ubicación, clave simétrica y algoritmo.

3.2.1.2. El sistema carga la clave simétrica con el algoritmo.

3.2.1.3. El sistema cifra el contenido del mensaje con la llave simétrica.

3.2.1.4. El sistema obtiene el mensaje cifrado y muestra el mensaje: “Mensaje cifrado satisfactoriamente” en los registros del sistema y el sub flujo finaliza.

3.2.2. Descifrar

3.2.2.1. El sistema hace válida la configuración del cifrado: ubicación clave simétrica y algoritmo.

3.2.2.2. El sistema carga la clave simétrica con el algoritmo.

3.2.2.3. El sistema descifra el contenido del mensaje con la clave simétrica.

3.2.2.4. El sistema obtiene el mensaje cifrado y muestra el mensaje “Mensaje descifrado satisfactoriamente” en los registros del sistema y el sub flujo finaliza.

3.2.3. Firmar

3.2.3.1. El sistema valida la configuración de la firma: datos de la clave privada; y algoritmo.

3.2.3.2. El sistema carga la clave privada con la configuración obtenida.

3.2.3.3. El sistema carga el algoritmo de firma con la clave privada.

3.2.3.4. El sistema firma el contenido del mensaje con la clave privada y algoritmo.

3.2.3.5. El sistema obtiene el mensaje firmado y muestra el mensaje “Mensaje firmado satisfactoriamente” en los registros del sistema y el sub flujo finaliza.

3.2.4. Verificar Firma

3.2.4.1. El sistema valida la configuración de la verificación: datos de la clave pública; y algoritmo.

3.2.4.2. El sistema carga la llave pública con la configuración obtenida.

3.2.4.3. El sistema carga el algoritmo de firma con la clave pública.

3.2.4.4. El sistema verifica la firma del mensaje.

3.2.4.5. El sistema muestra el mensaje “Firma del Mensaje verificado satisfactoriamente” en los registros del sistema y el sub flujo finaliza.

3.3. Flujos Alternos

3.3.1. Algoritmo Inválido

3.3.1.1. En el punto 4.1.2 del Flujo Básico, si el algoritmo no está habilitado en el tipo de cifrado, el sistema genera un mensaje de error con código “” y descripción “No se permite este algoritmo para [Tipo Cifrado]”; y se muestra en los registros del sistema y el caso de uso finaliza.

3.3.2. Operación Inválida

3.3.2.1. En el punto 4.1.2 del Flujo Básico, si la operación no está habilitada para el tipo de cifrado, el sistema genera un mensaje de error con código “” y descripción “No se permite esta operación para [Tipo Cifrado]”; y se muestra en los registros del sistema y el caso de uso finaliza.

3.3.3. Contenido del Mensaje no existe

3.3.3.1. En el punto 4.1.2 del Flujo Básico, si no existe contenido del mensaje, el sistema genera un mensaje de error con código “” y descripción “Contenido del mensaje no existe”; y se muestra en los registros del sistema y el caso de uso finaliza.

3.3.4. Configuración Inválida

3.3.4.1. Si no existen datos de la configuración para cada sub flujo, el sistema genera un mensaje de error con código “” y descripción “No se tienen suficientes valores para la operación”; y se muestra en los registros del sistema y el caso de uso finaliza.

3.3.5. No existe Llave Simétrica

3.3.5.1. Al cargar la llave simétrica para el cifrado o descifrado, si no existe la llave simétrica en la ubicación configurada, el sistema genera un mensaje de error con código “” y descripción “Error al obtener clave simétrica”; y se muestra en los registros del sistema y el caso de uso finaliza.

3.3.6. No existe Clave Privada

3.3.6.1. Al cargar la clave simétrica para la firma, si no existe la llave privada con la configuración proporcionada, el sistema genera un mensaje de error con código “” y descripción “Error al obtener llave privada”; y se muestra en los registros del sistema y el caso de uso finaliza.

3.3.7. No existe Clave Pública

3.3.7.1. Al cargar la clave simétrica para la verificación de la firma, si no existe la clave pública con la configuración proporcionada, el sistema genera un mensaje de error con código “” y descripción “Error al obtener clave pública”; y se muestra en los registros del sistema y el caso de uso finaliza.

3.3.8. Error al cifrar mensaje

3.3.8.1. En el punto 4.2.1.3, si ocurre un error al cifrar el mensaje, el sistema genera un mensaje de error con código “” y descripción “Error al cifrar el mensaje”; y se muestra en los registros del sistema y el caso de uso finaliza

3.3.9. Error al descifrar mensaje

3.3.9.1. En el punto 4.2.2.3, si ocurre un error al descifrar el mensaje, el sistema genera un mensaje de error con código “” y descripción “Error al descifrar el mensaje”; y se muestra en los registros del sistema y el caso de uso finaliza.

3.3.10. Error al firmar mensaje

3.3.10.1. En el punto 4.2.3.4, si ocurre un error al cifrar el mensaje, el sistema genera un mensaje de error con código “” y descripción “Error al firmar el mensaje”; y se muestra en los registros del sistema y el caso de uso finaliza.

3.3.11. Error al verificar la firma mensaje

3.3.11.1. En el punto 4.2.4.4, si ocurre un error al descifrar el mensaje, el sistema genera un mensaje de error con código “” y descripción “Error al verificar la firma del mensaje”; y se muestra en los registros del sistema y el caso de uso finaliza.

3.3.12. Error sistema

3.3.12.1. Si ocurre un error en el sistema desconocido, genera una respuesta con código “99” y la descripción del error “Error en el sistema”; y se muestra en los registros del sistema y el caso de uso finaliza.

4. Precondiciones

4.1. Clave Simétrica Disponible

4.1.1. El sistema tiene disponible la clave simétrica en la ubicación configurada.

4.2. Clave Pública Disponible

4.2.1. El sistema tiene disponible la clave pública en la ubicación configurada.

4.3. Clave Privada Disponible

4.3.1. El sistema tiene disponible la clave privada en la ubicación configurada.

5. Postcondiciones

5.1. Mensaje Cifrado

5.1.1. El contenido del mensaje se devuelve cifrado con el código de mensaje “00.

5.2. Mensaje Descifrado

5.2.1. El contenido del mensaje se devuelve descifrado con el código de mensaje “00.

5.3. Mensaje Firmado

5.3.1. El contenido del mensaje se devuelve firmado con el código de mensaje “00.

5.4. Mensaje Verificado

5.4.1. El contenido del mensaje se verificó satisfactoriamente con el código de mensaje “00.

5.5. Puntos de Inclusión

No aplica

6. **Puntos de Extensión**

No aplica

7. **Requerimientos Funcionales asociados**

No aplica

8. **Reglas de Negocio**

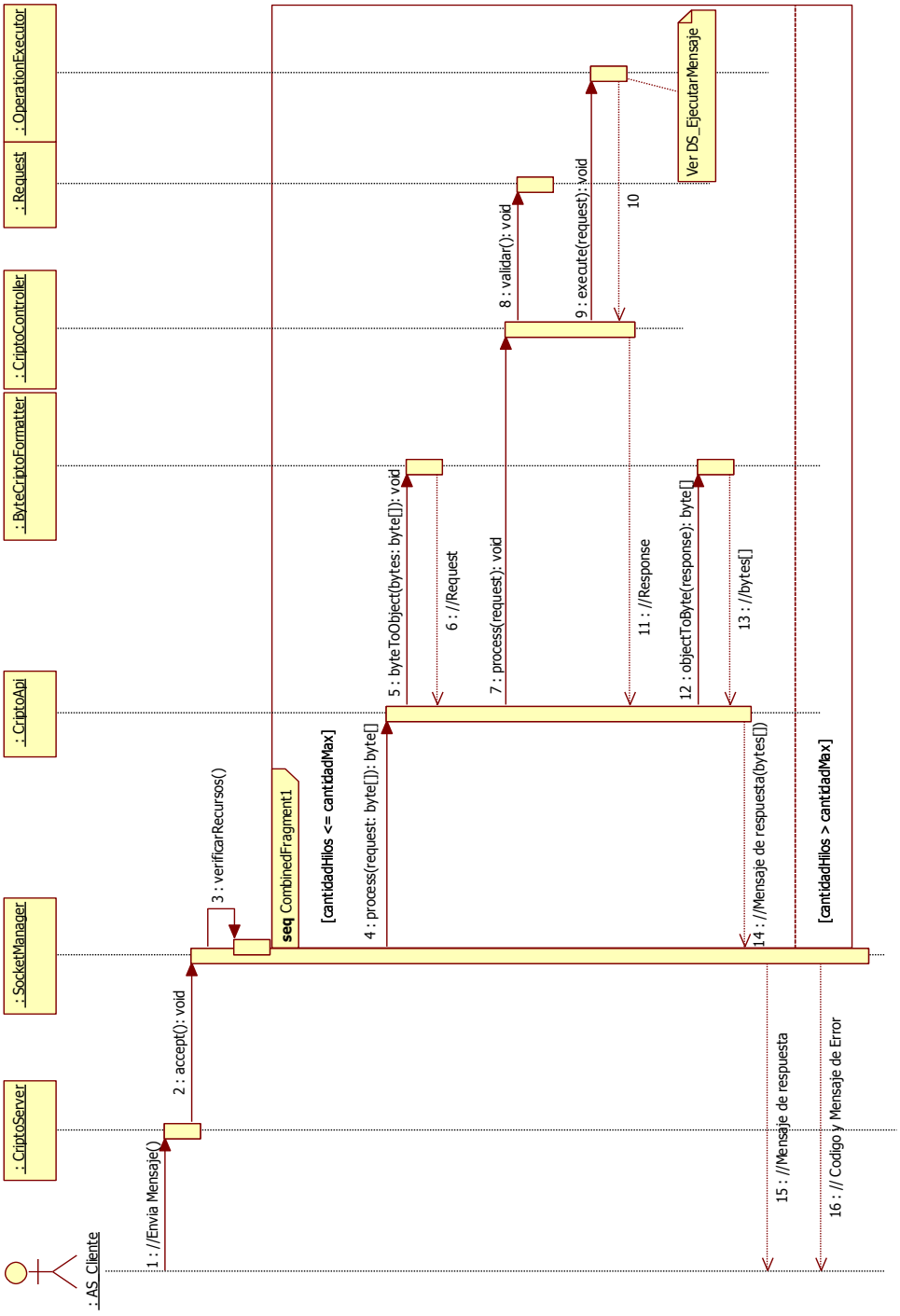
No aplica

9. **Interfaces asociadas**

10. No aplica

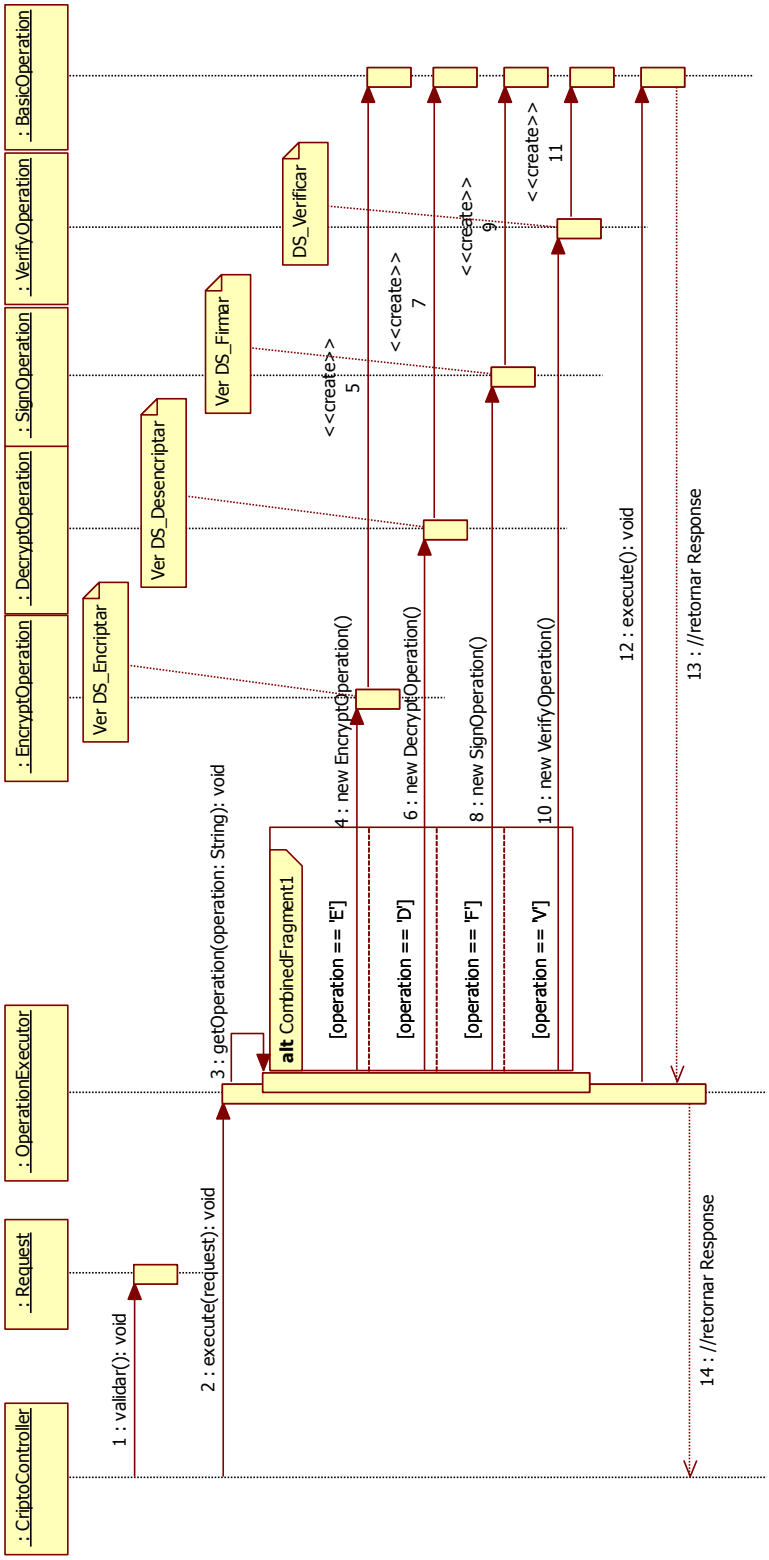
CU Diagrama de Secuencia.

Figura 45: Diagrama de secuencia enviar Mensaje



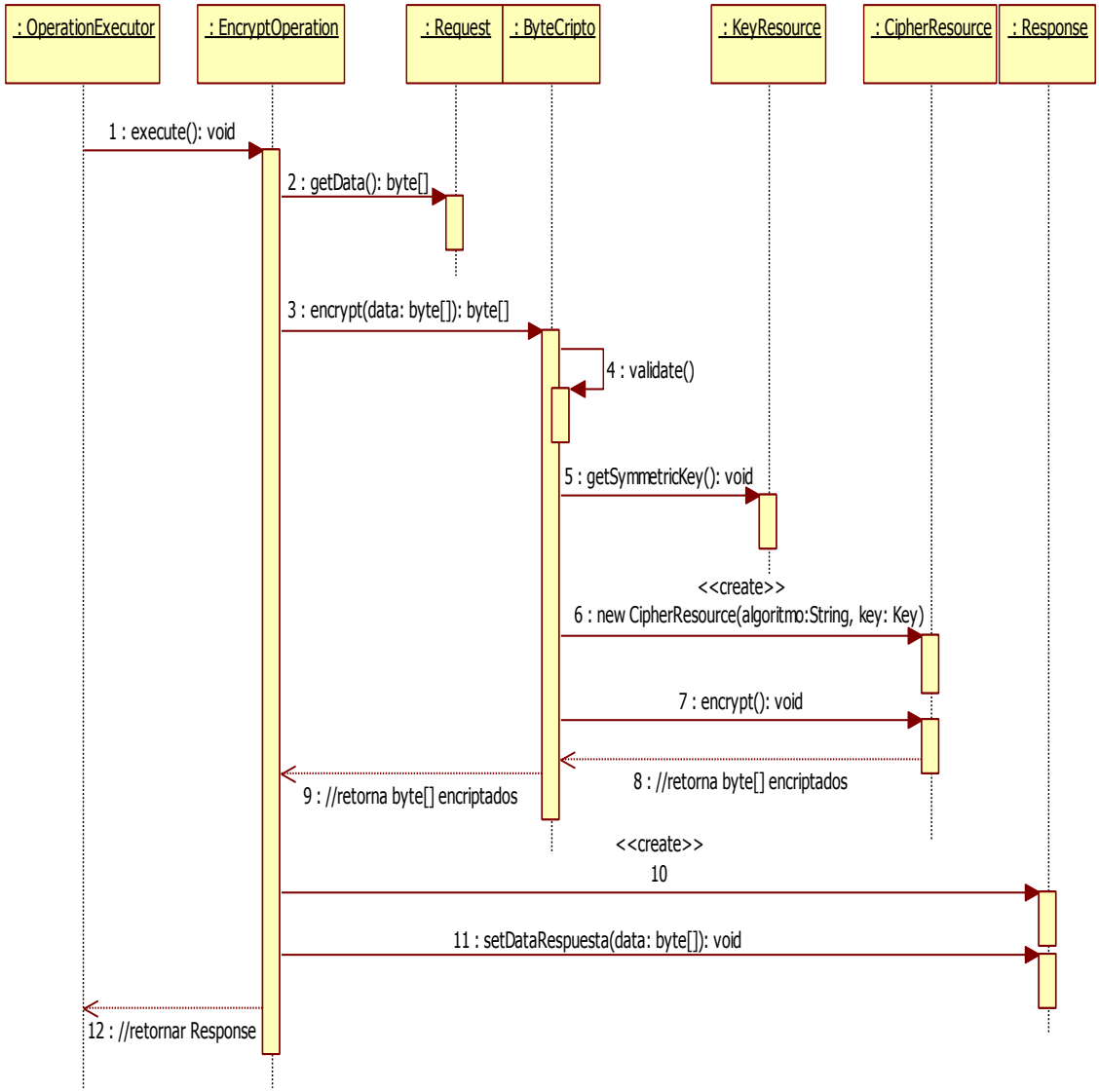
Fuente: Elaboración propia

Figura 46: Diagrama de secuencia de Ejecutar mensaje



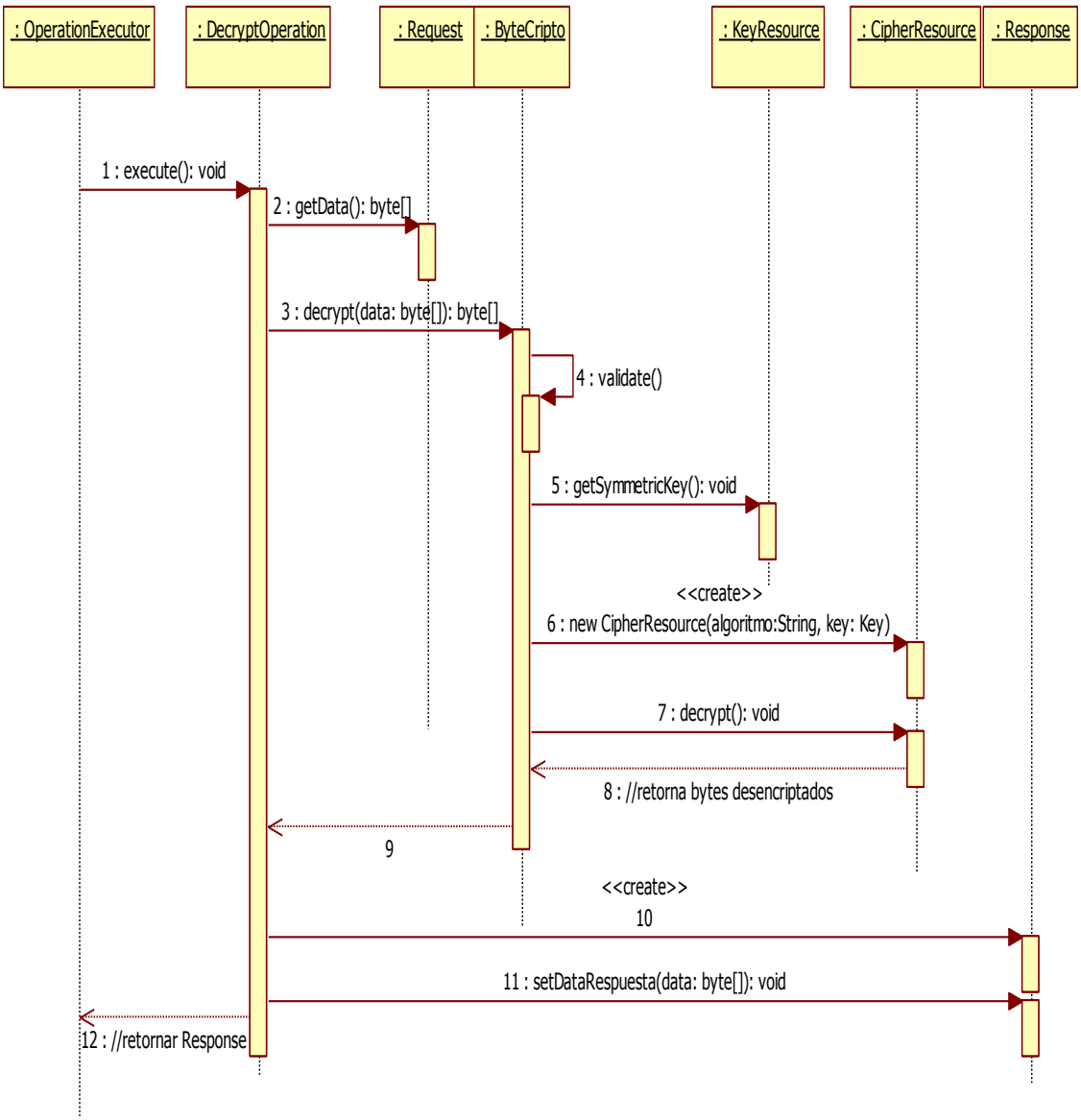
Fuente: Elaboración propia

Figura 47: Diagrama de secuencia de encriptar



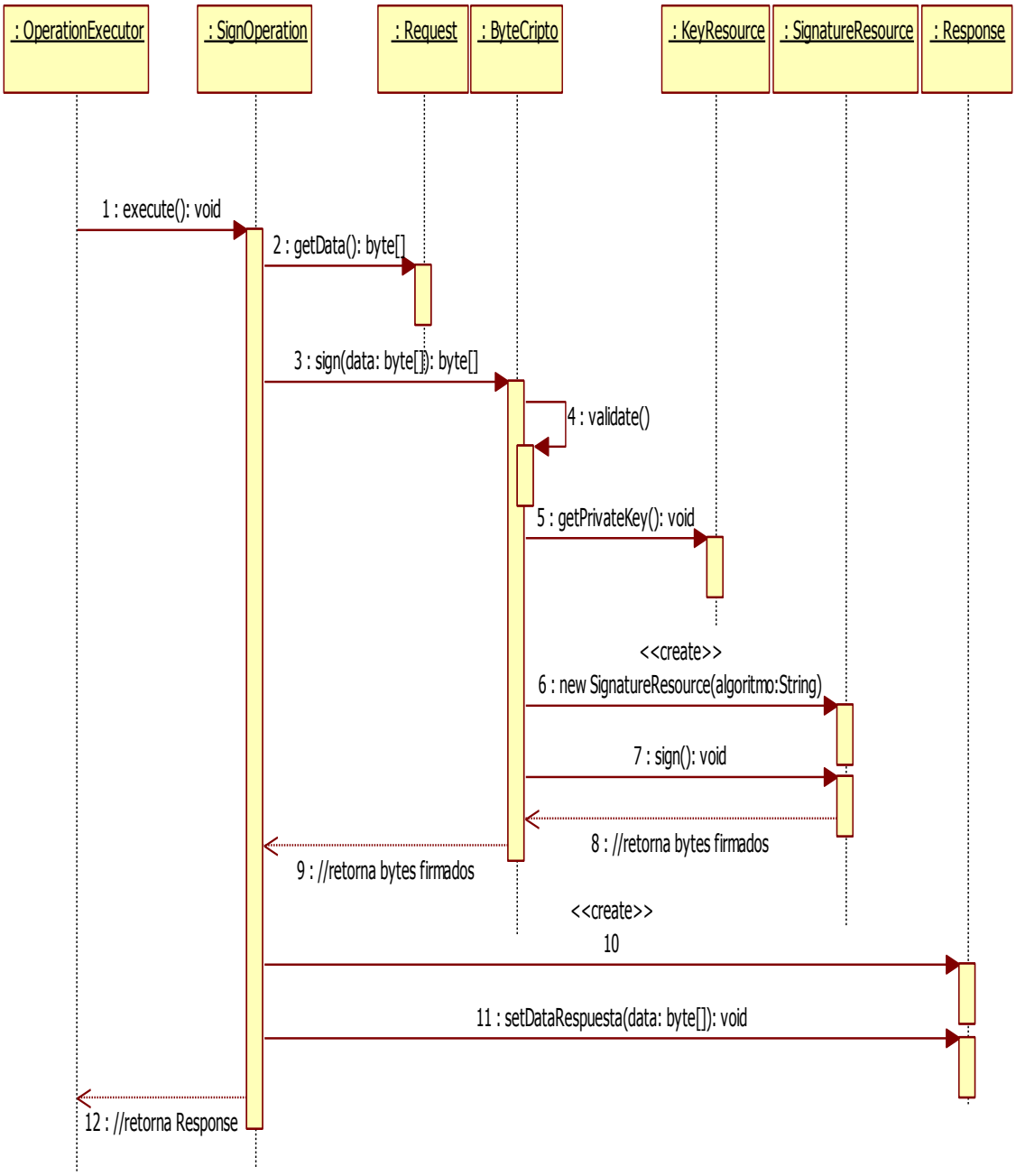
Fuente: Elaboración propia

Figura 48: Diagrama de secuencia de desencriptar



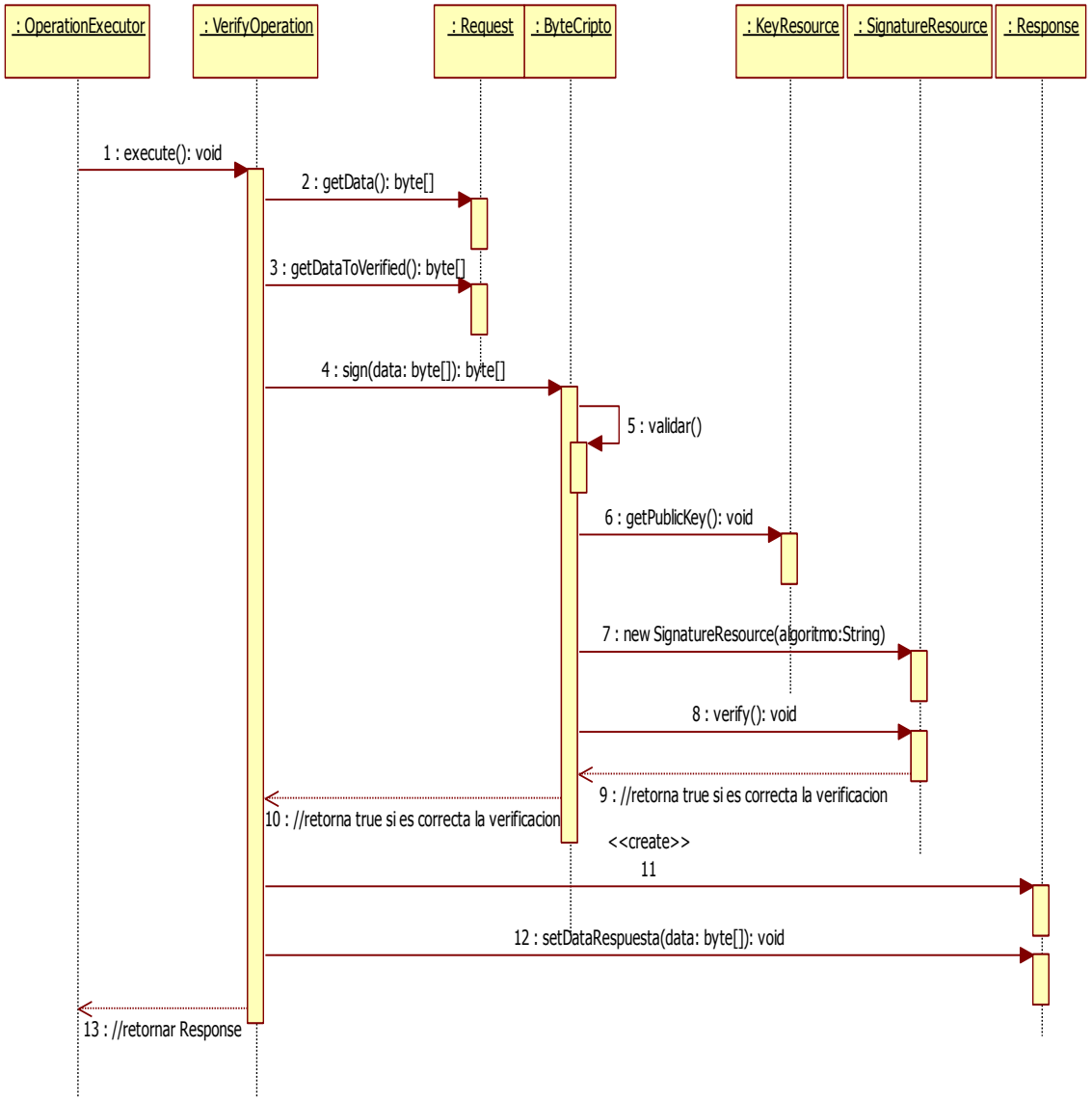
Fuente: Elaboración propia

Figura 49: Diagrama de secuencia de firmar



Fuente: Elaboración propia

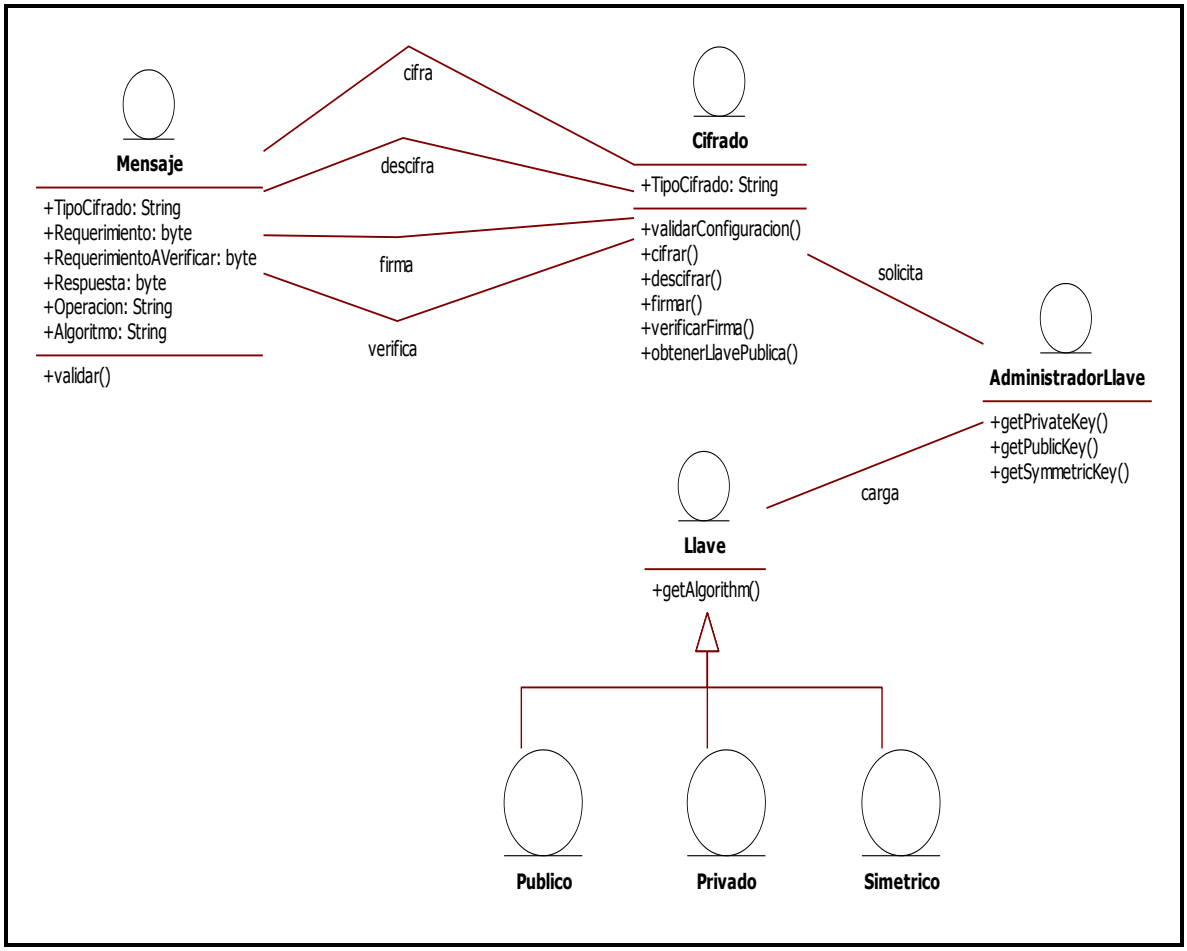
Figura 50: Diagrama de secuencia de verificar



Fuente: Elaboración propia

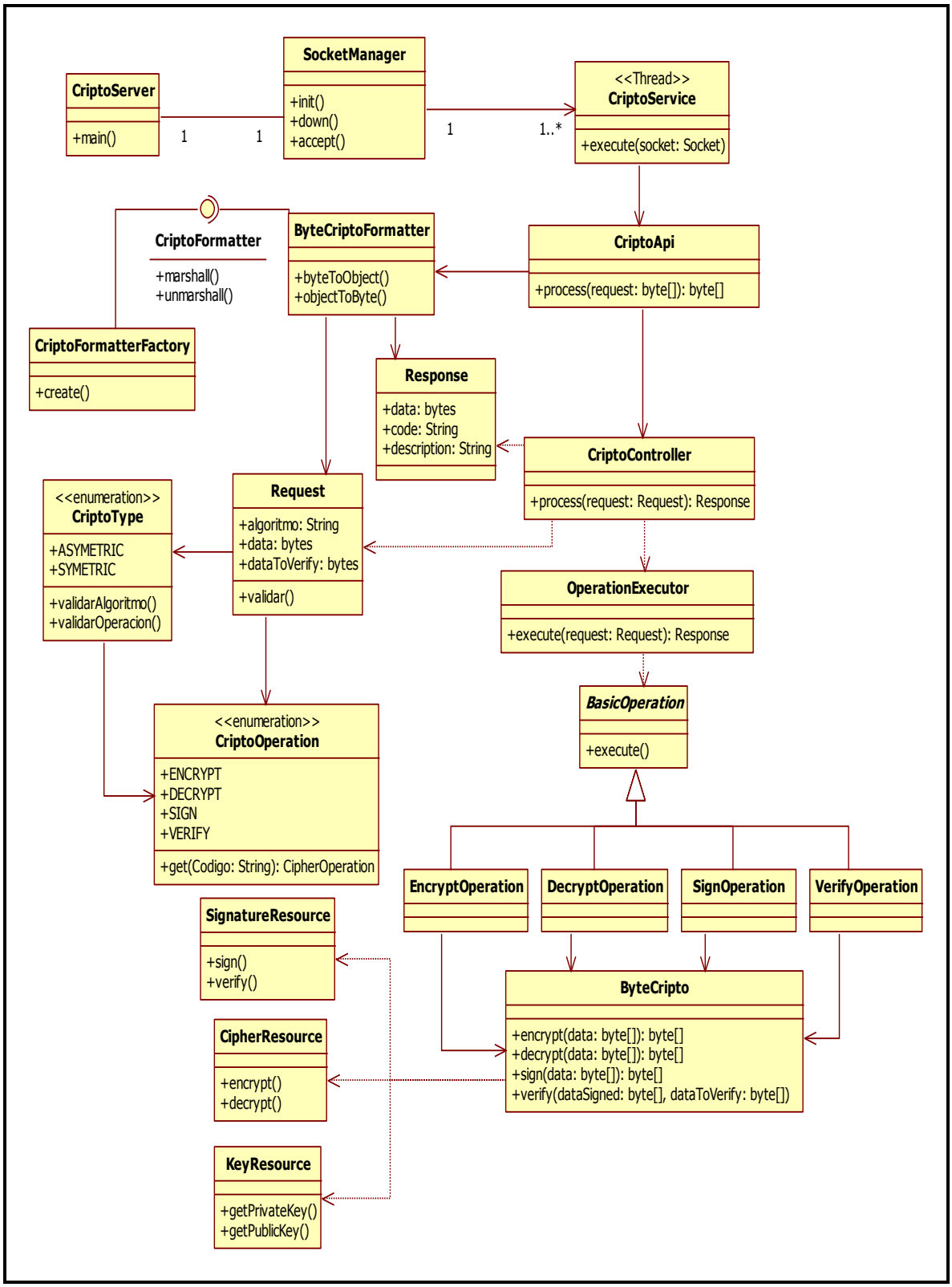
CU Diagrama de Clases.

Figura 51: Diagrama del modelo Conceptual.



Fuente: Elaboración propia

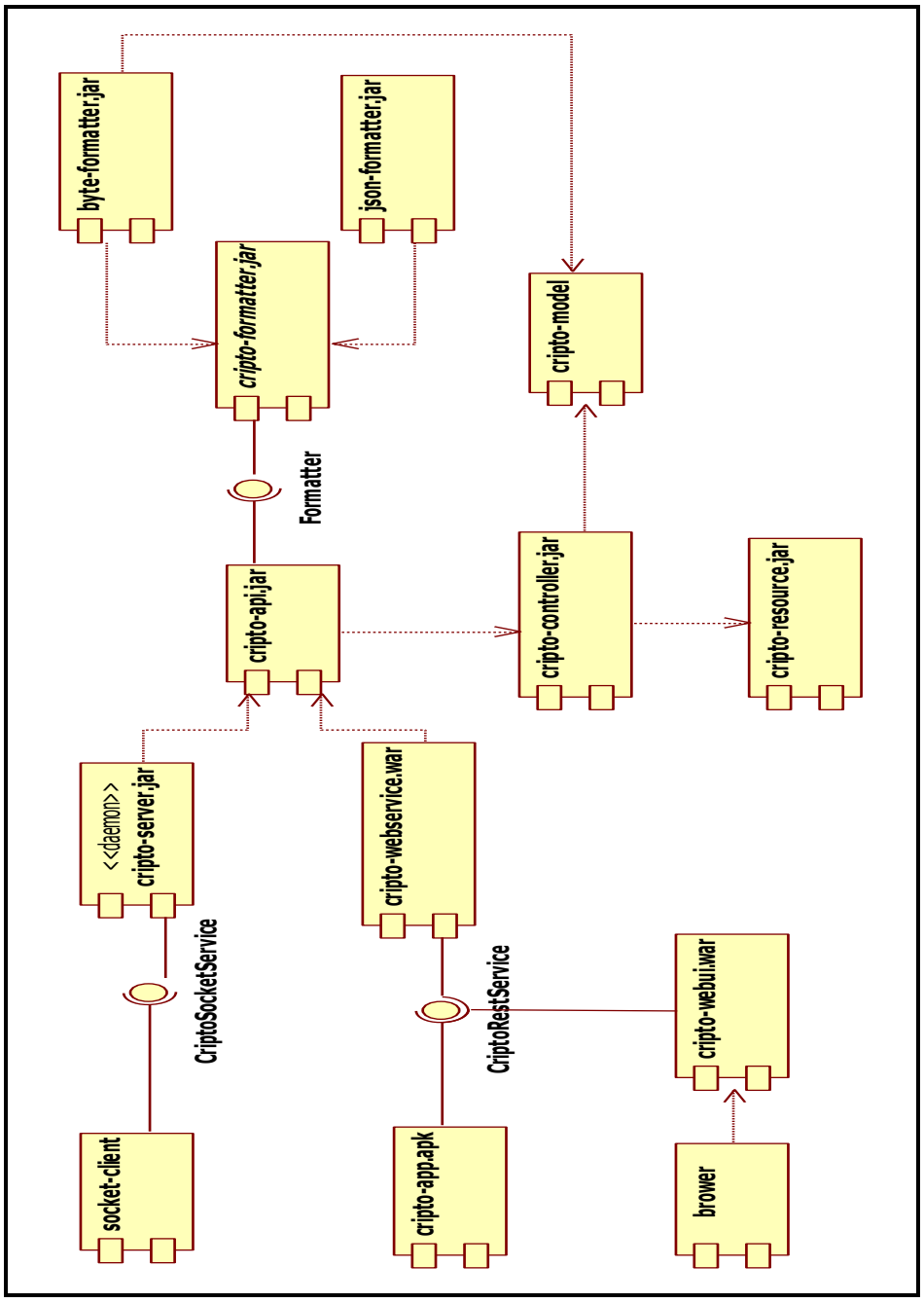
Figura 52: Diagrama del modelo lógico



Fuente: Elaboración propia

Modelo de Análisis

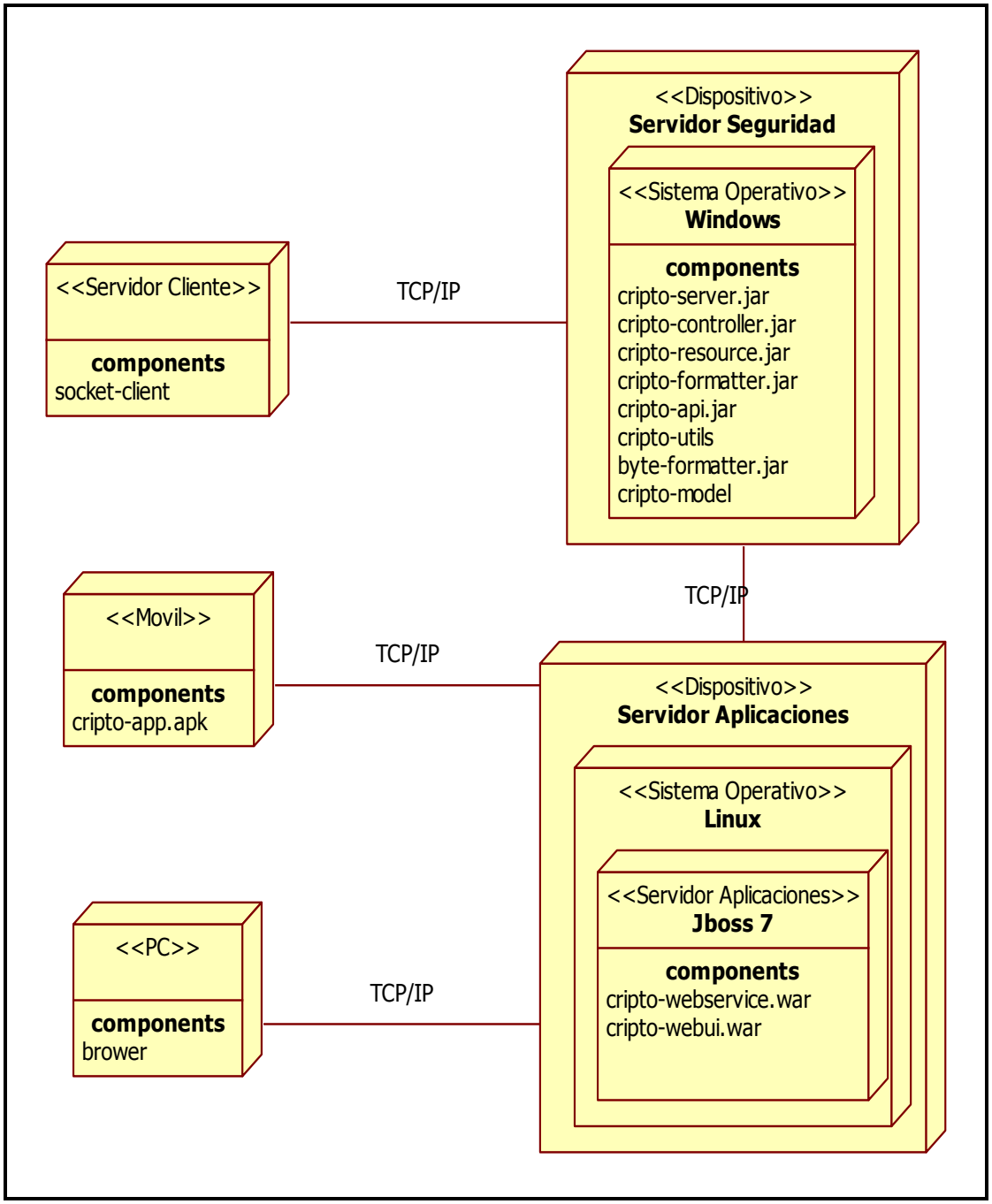
Figura 53: Diagrama de Componente Sistema



Fuente: Elaboración propia

CU Diagrama de Despliegue.

Figura 54: Diagrama de despliegue



Fuente: Elaboración propia

2.5 Especificaciones Suplementarias

Los requerimientos fueron especificados por Juicio experto. Contiene tanto requerimientos funcionales como no funcionales del sistema. Las funcionalidades a considerarse para este artefacto son las siguientes: usabilidad, confiabilidad, desempeño, seguridad, restricciones de diseño, requerimientos de documentación en línea y de sistemas de ayuda, componentes comprados, interfaces, requerimientos de licenciamiento, y aspectos legales, derecho de autor y otros avisos.

2.5.1 Funcionalidad común

Los componentes de encriptación proporcionan a los productos mecanismos de seguridad que les permiten proteger tanto su información, coma la información del cliente final, a nivel de integridad, como confidencialidad.

2.5.2 Utilidad

Las interfaces del sistema están diseñadas para que sean fáciles de implementar en un producto o para que sean fáciles de consumir como servicio.

Las interfaces de configuración de la plataforma, tiende a ser amigable y Estándar. Para lograrlo esto, mantenemos validaciones de los campos de las interfaces, de forma tal que la trazabilidad de los errores o problemas de uso sería fácil de identificar.

La disponibilidad de los componentes, será administrado mediante un repositorio de componentes implementados por la empresa.

Los componentes podrán ser consumidos como servicios dependiendo de la arquitectura de la plataforma transaccional a implementar.

2.5.3 Performance

Los componentes fueron diseñados para soportar una alta carga, soportando concurrencias óptimas de 400 Tps, bajo una configuración base de la plataforma, pero mediante tuneos de esta los se podría soportar 880 TPS. Con tiempo de respuesta de 40 milisegundos

Cabe mencionar que si se utiliza como componente, los tiempos de respuesta que medirán serán del producto.

2.5.4 Restricciones de diseño

En el diseño del software no se han considerado los algoritmos que no están orientados a sistemas financieros, esto debido a que solo nos enfocamos en las necesidades estratégicas de la empresa.

3. Interfaces del Sistema

```
public interface Crypto<S,T> {
 public enum CryptoMode {
 ENCRYPT, DECRYPT, VERIFY, SIGN
 }
 public void config(Properties prop);
 /**
 * Metodo que firma flujo de bytes con las llaves cargadas en el metodo
 * config
 *
 * @param in flujo de bytes para firmar
 * @return flujo de bytes de la firma
 * @throws CryptoException
 */
 public T sign(S in) throws CryptoException;
 /**
 * Metodo que cifra un flujo de bytes con la llave cargada en el metodo
 * config
 *
 * @param in flujo de bytes para cifrar
 * @return flujo de bytes cifrados
 * @throws CryptoException
```

```
 */
 public T encrypt(S in) throws CryptoException;
 /**
 * Metodo que descifra un flujo de bytes con la llave cargada en el metodo
 * config
 *
 * @param in flujo de bytes para descifrar
 * @return flujo de bytes descifrados
 * @throws CryptoException
 */
 public T decrypt(S in) throws CryptoException;
 /**
 * Metodo que verifica la firma contra un flujo de bytes con la llave
 * cargada en el metodo config
 *
 * @param in flujo de bytes de la firma
 * @return valor boolean que indica el resultado de la verificacion
 * @throws CryptoException
 */
 public boolean verify(S in) throws CryptoException;

}
```

Código fuente base de la aplicación.

```
package com.txr.components.ImplFile;

import com.txr.components.crypto.Crypto;
import java.security.Key;
import java.security.PrivateKey;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.txr.components.exceptions.CryptoException;
import com.txr.components.exceptions.ParametersCryptoException;
import java.io.FileInputStream;
import java.security.KeyStore;
import java.security.cert.Certificate;
import java.util.Properties;

public abstract class BasicCipher<S, T> implements Crypto<S, T> {

 private static final Logger log = LoggerFactory.getLogger(BasicCipher.class);

 protected Properties prop;

 public BasicCipher() {
 }
```

```
@Override
public void config(Properties props) throws CryptoException {
this.prop = props;
}

protected void validarDatos(String[] enc_props) {
for (String value : enc_props) {
if (!(prop.containsKey(value) && !prop.get(value).equals(""))) {
throw new ParametersCryptoException("No se tienen suficientes valores para la accion");
}
}
}

protected Key getPrivateKey(String privateKeyFile, String alias, String keystoreLoadPassword,
String privatePassword) {
Key privateKey;
try {
log.debug("Obteniendo clave privada de keystore");
KeyStore ks;
FileInputStream fisKey;

// Consiguiendo la llave Privada antes generada
ks = KeyStore.getInstance(KeyStore.getDefaultType());
fisKey = new FileInputStream(privateKeyFile);
ks.load(fisKey, keystoreLoadPassword.toCharArray());
fisKey.close();

// llave Privada
privateKey = (PrivateKey) ks.getKey(alias, privatePassword.toCharArray());
} catch (Exception ex) {
throw new CryptoException("Error al obtener clave privada", ex);
}
return privateKey;
}

protected Key getPublicKey(String signKeyFile, String signKeyAlias, String
keystoreLoadPassword) {
Key publicKey;
try {
log.debug("Obteniendo clave publica de keystore");
KeyStore ks;
FileInputStream fisKey;
Certificate cert;

ks = KeyStore.getInstance(KeyStore.getDefaultType());
fisKey = new FileInputStream(signKeyFile);
ks.load(fisKey, keystoreLoadPassword.toCharArray());
fisKey.close();

// llave Publica
cert = ks.getCertificate(signKeyAlias);
```

```
 publicKey = cert.getPublicKey();
 } catch (Exception ex) {
 throw new CryptoException("Error al obtener clave publica", ex);
 }

 return publicKey;
 }
}
```

```
package com.txr.components.ImplFile;

import java.io.FileOutputStream;
import java.security.Key;
import java.security.PrivateKey;
import java.security.PublicKey;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.txr.components.exceptions.CryptoException;
import com.txr.components.support.CryptFileSupport;
import com.txr.components.support.KeySupport;

public class ByteCipher extends BasicCipher<byte[], byte[]> {

 private static final Logger log = LoggerFactory.getLogger(ByteCipher.class);

 private static final String[] ENC_PROPS = {"encKeyFile", "encSignKeyFile", "encSignAlg",
 "encAlg", "encSignKeyAlias", "encKeystoreLoadPassword", "encPrivatePassword"};
 private static final String[] DEC_PROPS = {"decKeyFile", "decAlg"};
 private static final String[] VER_PROPS = {"verSignFile", "verSignKeyFile", "verSignKeyAlias",
"verKeystoreLoadPassword", "verSignAlg"};

 public ByteCipher() {
 }



 public byte[] sign(byte[] in) throws CryptoException {

 try {
 log.debug("Iniciando cifrado de archivo segun properties");
 byte[] signOut;
 FileOutputStream fos;
 FileSign signer;
 CryptFile crypt;
 Key privateKey;
 Key symmetricKey;
```

```
//verificar datos
validarDatos(ENC_PROPS);
signer = new FileSign(prop.getProperty("encSignAlg"));

//Leer llave privada de archivo
privateKey = getPrivateKey(prop.getProperty("encSignKeyFile"),
prop.getProperty("encSignKeyAlias"),
prop.getProperty("encKeystoreLoadPassword"),
prop.getProperty("encPrivatePassword"));

//sign
signOut = signer.sign(in, (PrivateKey) privateKey);
return signOut;

} catch (Exception ex) {
throw new CryptoException("Error al intentar cifrar arreglo bytes", ex);
}
}

@Override
public byte[] encrypt(byte[] in) throws CryptoException {
Key symmetricKey;
String encKeyFile;
String algoritm;
byte[] out;

try {
log.debug("Iniciando cifrado de bytes segun arreglo de bytes");
CryptFile crypt;
//leer llave simetrica de archivo
algoritm = prop.getProperty("encAlg");

symmetricKey = KeySupport.getSymmetricKey(prop.getProperty("encKeyFile"), algoritm);

//cifrar con filein, fileout, algoritmo, llave
out = CryptFileSupport.encryptBytes(algoritm, symmetricKey, in);
return out;
} catch (Exception ex) {
throw new CryptoException("Error al intentar cifrar flujo bytes", ex);
}
}

@Override
public byte[] decrypt(byte[] in) throws CryptoException {
String algoritm;
byte[] out;
try {
log.debug("Inciando descifrar archivo segun properties");
Key symmetricKey;
CryptFile crypt;
```

```
//verificar datos
validarDatos(DEC_PROPS);
//leer llave simetrica de archivo
algoritm = prop.getProperty("decAlg");

symmetricKey = KeySupport.getSymmetricKey(prop.getProperty("decKeyFile"), algoritm);

//descifrar file
out = CryptFileSupport.decryptBytes(algoritm, symmetricKey, in);
return out;
} catch (Exception ex) {
throw new CryptoException("Error al intentar descifrar arreglo bytes", ex);
}
}

@Override
public boolean verify(byte[] in) throws CryptoException {
boolean result = false;
try {
log.debug("Iniciando verificar firma de arreglo bytes");

FileSign signer;
Key publicKey;
//verificar datos
validarDatos(VER_PROPS);
//Leer llave publica de archivo
publicKey = getPublicKey(prop.getProperty("verSignKeyFile"),
prop.getProperty("verSignKeyAlias"),
prop.getProperty("verKeystoreLoadPassword"));

signer = new FileSign(prop.getProperty("verSignAlg"));
result = signer.verify(in, (PublicKey) publicKey, prop.getProperty("verSignFile"));
} catch (Exception e) {
throw new CryptoException("Error al intentar verificar firma de archivo", e);
}
return result;
}

}
```

```
package com.txr.components.ImplFile;

import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
```

```
import java.io.OutputStream;
import java.security.GeneralSecurityException;
import java.security.InvalidKeyException;
import java.security.Key;

import javax.crypto.Cipher;
import javax.crypto.CipherInputStream;
import javax.crypto.CipherOutputStream;
import javax.crypto.spec.SecretKeySpec;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class CryptFile {

 private static final Logger log = LoggerFactory.getLogger(CryptFile.class);

 private String algorithm = "AES";
 private Cipher cipher;
 private SecretKeySpec keySpec;

 public CryptFile(String algorithm, Key secKey) throws GeneralSecurityException {
 // instancia cipher
 this.algorithm = algorithm;
 this.keySpec = (SecretKeySpec) secKey;

 cipher = Cipher.getInstance(this.algorithm);
 }

 public byte[] encrypt(byte[] in) throws IOException, InvalidKeyException {
 log.debug("Cifrando arreglo bytes {}", in.length);
 cipher.init(Cipher.ENCRYPT_MODE, keySpec);

 ByteArrayInputStream is = new ByteArrayInputStream(in);

 ByteArrayOutputStream baos = new ByteArrayOutputStream();
 CipherOutputStream os = new CipherOutputStream(baos, cipher);

 copyAndClose(is, os);

 return baos.toByteArray();
 }

 public void encrypt(File in, File out) throws IOException, InvalidKeyException {
 log.debug("Cifrando archivo {}", in.getName());
 cipher.init(Cipher.ENCRYPT_MODE, keySpec);

 FileInputStream is = new FileInputStream(in);
 CipherOutputStream os = new CipherOutputStream(new FileOutputStream(out), cipher);

 copyAndClose(is, os);
```

```
}

public byte[] decrypt(byte[] in) throws IOException, InvalidKeyException {
log.debug("Descifrando arreglo bytes {}", in.length);
cipher.init(Cipher.DECRYPT_MODE, keySpec);

CipherInputStream is = new CipherInputStream(new ByteArrayInputStream(in), cipher);
ByteArrayOutputStream os = new ByteArrayOutputStream();
copyAndClose(is, os);

return os.toByteArray();
}

public void decrypt(File in, File out) throws IOException, InvalidKeyException {
log.debug("Descifrando archivo {}", in.getName());
cipher.init(Cipher.DECRYPT_MODE, keySpec);

CipherInputStream is = new CipherInputStream(new FileInputStream(in), cipher);
FileOutputStream os = new FileOutputStream(out);
copyAndClose(is, os);
}

private void copyAndClose(InputStream is, OutputStream os) throws IOException {
copy(is, os);
is.close();
os.close();

}

private void copy(InputStream is, OutputStream os) throws IOException {
int i;
byte[] b = new byte[1024];
while ((i = is.read(b)) != -1) {
os.write(b, 0, i);
}
}
}
```

```
package com.txr.components.ImplFile;

import java.io.File;
import java.io.FileOutputStream;
import java.security.Key;
import java.security.PrivateKey;
import java.security.PublicKey;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.txr.components.exceptions.CryptoException;
```

```
import com.txr.components.support.KeySupport;

public class FileCipher extends BasicCipher<Object, Boolean> {

 private static final Logger log = LoggerFactory.getLogger(FileCipher.class);

 private String[] ENC_PROPS = {"encFileIn", "encFileOut", "encSignFileOut", "encKeyFile",
"encSignKeyFile", "encSignAlg",
 "encAlg", "encSignKeyAlias", "encKeystoreLoadPassword", "encPrivatePassword"};
 private String[] DEC_PROPS = {"decFileIn", "decFileOut", "decKeyFile", "decAlg"};
 private String[] VER_PROPS = {"verFileIn", "verSignFile", "verSignKeyFile", "verSignKeyAlias",
"verKeystoreLoadPassword", "verSignAlg"};

 public FileCipher() {
 }

 @Override
 public boolean verify(Object in) throws CryptoException {
 return verify();
 }

 @Override
 public Boolean sign(Object in) throws CryptoException {
 sign();
 return Boolean.TRUE;
 }

 @Override
 public Boolean encrypt(Object in) throws CryptoException {
 encrypt();
 return Boolean.TRUE;
 }

 @Override
 public Boolean decrypt(Object in) throws CryptoException {
 decrypt();
 return Boolean.TRUE;
 }

 public boolean verify() {
 boolean result = false;
 try {
 log.debug("Iniciando verificar firma de archivo");

 FileSign signer;
 Key publicKey;
 //verificar datos
 validarDatos(VER_PROPS);
 //Leer llave publica de archivo
 publicKey = getPublicKey(prop.getProperty("verSignKeyFile"),
 prop.getProperty("verSignKeyAlias"),
```

```
 prop.getProperty("verKeystoreLoadPassword"));

 signer = new FileSign(prop.getProperty("verSignAlg"));
 result = signer.verify(prop.getProperty("verFileIn"), (PublicKey) publicKey,
prop.getProperty("verSignFile"));
 } catch (Exception e) {
 throw new CryptoException("Error al intentar verificar firma de archivo", e);
 }
 return result;
 }

 public void decrypt() {
 String algoritm;
 try {
 log.debug("Inciando descifrar archivo segun properties");
 Key symmetricKey;
 CryptFile crypt;

 //verificar datos
 validarDatos(DEC_PROPS);

 //leer llave simetrica de archivo
 algoritm = prop.getProperty("decAlg");
 symmetricKey = KeySupport.getSymmetricKey(prop.getProperty("decKeyFile"),
algoritm);//getSymmetricKey(prop.getProperty("decKeyFile"));

 //descifrar file
 crypt = new CryptFile(algoritm, symmetricKey);
 crypt.decrypt(new File(prop.getProperty("decFileIn")), new
File(prop.getProperty("decFileOut")));
 } catch (Exception ex) {
 throw new CryptoException("Error al intentar descifrar archivo", ex);
 }
 }

 public void sign() {

 try {
 log.debug("Iniciando cifrado de archivo segun properties");
 byte[] signOut;
 FileOutputStream fos;
 FileSign signer;
 CryptFile crypt;
 Key privateKey;
 Key symmetricKey;

 //verificar datos
 validarDatos(ENC_PROPS);

 signer = new FileSign(prop.getProperty("encSignAlg"));
```

```
//Leer llave privada de archivo
privateKey = getPrivateKey(prop.getProperty("encSignKeyFile"),
prop.getProperty("encSignKeyAlias"),
prop.getProperty("encKeystoreLoadPassword"),
prop.getProperty("encPrivatePassword"));

//sign File
signOut = signer.sign(prop.getProperty("encFileIn"), (PrivateKey) privateKey);
fos = new FileOutputStream(prop.getProperty("encSignFileOut"));
fos.write(signOut);
fos.flush();
fos.close();

} catch (Exception ex) {
throw new CryptoException("Error al intentar cifrar archivo", ex);
}

}

public void encrypt() {
String algoritm;
try {
log.debug("Iniciando cifrado de archivo segun properties");
byte[] signOut;
FileOutputStream fos;
//FileSign signer;
CryptFile crypt;
Key privateKey;
Key symmetricKey;

//verificar datos
validarDatos(ENC_PROPS);

//signer = new FileSign(prop.getProperty("encSignAlg"));
//Leer llave privada de archivo
privateKey = getPrivateKey(prop.getProperty("encSignKeyFile"),
prop.getProperty("encSignKeyAlias"),
prop.getProperty("encKeystoreLoadPassword"),
prop.getProperty("encPrivatePassword"));

//leer llave simetrica de archivo
algoritm = prop.getProperty("encAlg");
symmetricKey = KeySupport.getSymmetricKey(prop.getProperty("encKeyFile"),
algoritm);//getSymmetricKey(prop.getProperty("encKeyFile"));

//cifrar con filein, fileout, algoritmo, llave
crypt = new CryptFile(algoritm, symmetricKey);
crypt.encrypt(new File(prop.getProperty("encFileIn")), new
File(prop.getProperty("encFileOut")));

} catch (Exception ex) {
```

```
 throw new CryptoException("Error al intentar cifrar archivo", ex);
 }

 }

}
```

```
package com.txr.components.ImplFile;

import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.security.PrivateKey;
import java.security.PublicKey;
import java.security.Signature;
import java.security.SignatureException;
import org.apache.log4j.Logger;
public class FileSign {
 private static final Logger log = Logger.getLogger(FileSign.class);
 private String algorithm = "SHA1withRSA";
 public FileSign(String algorithm) {
 this.algorithm = algorithm;
 }

 public String convertToHexa(byte[] bytes) {
 StringBuilder sb = new StringBuilder("");
 for (byte b : bytes) {
 sb.append(String.format("%02X", b));
 }
 return sb.toString();
 }

 public byte[] sign(byte[] data, PrivateKey prvKey) throws NoSuchAlgorithmException,
InvalidKeyException, IOException, SignatureException {
 log.debug("Creando firma a partir de arreglo de bytes");
 ByteArrayInputStream fis = new ByteArrayInputStream(data);
 return sign(fis, prvKey);
 }

 public byte[] sign(String datafile, PrivateKey prvKey) throws NoSuchAlgorithmException,
InvalidKeyException, IOException, SignatureException {
 log.debug("Creando firma a partir de archivo");
 FileInputStream fis = new FileInputStream(datafile);
 return sign(fis, prvKey);
 }
```

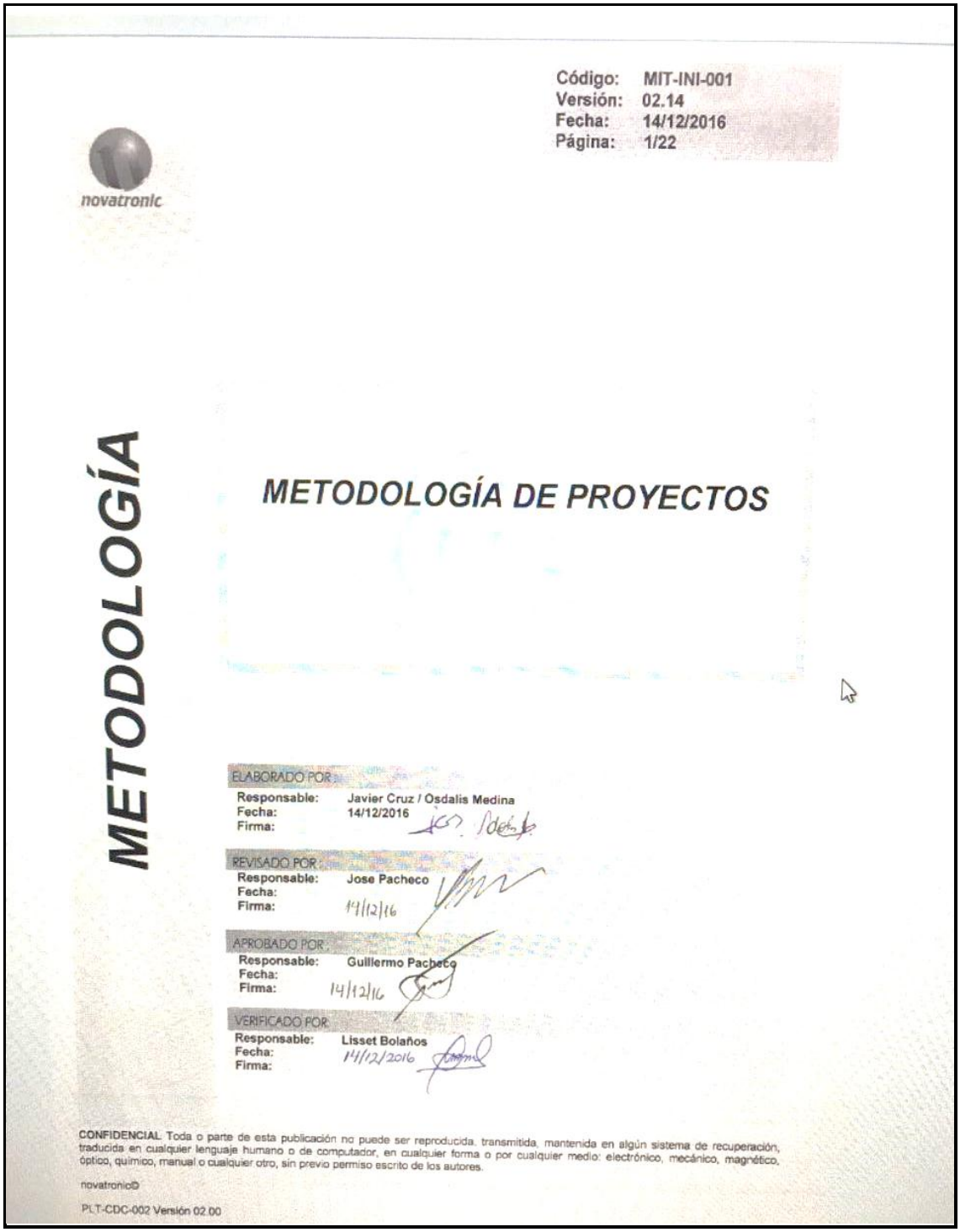
```
 public byte[] sign(InputStream fis, PrivateKey prvKey) throws NoSuchAlgorithmException,
InvalidKeyException, IOException, SignatureException {
 Signature sig = Signature.getInstance(algorithm);
 sig.initSign(prvKey);
 byte[] dataBytes = new byte[1024];
 int nread = fis.read(dataBytes);
 while (nread > 0) {
 sig.update(dataBytes, 0, nread);
 nread = fis.read(dataBytes);
 }
 fis.close();
 return sig.sign();
 }

 public boolean verify(byte[] data, PublicKey pubKey, String sigbytes) throws
NoSuchAlgorithmException, InvalidKeyException, IOException, SignatureException {
 ByteArrayInputStream fis = new ByteArrayInputStream(data);
 return verify(fis, pubKey, sigbytes);
 }

 public boolean verify(String datafile, PublicKey pubKey, String sigbytes) throws
NoSuchAlgorithmException, InvalidKeyException, IOException, SignatureException {
 FileInputStream fis = new FileInputStream(datafile);
 return verify(fis, pubKey, sigbytes);
 }

 public boolean verify(InputStream fis, PublicKey pubKey, String sigbytes) throws
NoSuchAlgorithmException, InvalidKeyException, IOException, SignatureException {
 log.debug("Verificando firma");
 Signature sig = Signature.getInstance(algorithm);
 sig.initVerify(pubKey);
 byte[] dataBytes = new byte[1024];
 int nread = fis.read(dataBytes);
 while (nread > 0) {
 sig.update(dataBytes, 0, nread);
 nread = fis.read(dataBytes);
 };
 fis.close()
 File sigFile = new File(sigbytes);
 FileInputStream fisign = new FileInputStream(sigFile);
 byte[] sigBytes = new byte[(int) sigFile.length()];
 fisign.read(sigBytes);
 fisign.close();
 return sig.verify(sigBytes);
 }
}
```

Anexo 6: Metodología de gestión de proyectos



Código: MIT-INI-001
Versión: 02.14
Fecha: 14/12/2016
Página: 1/22

novatronic

METODOLOGÍA

METODOLOGÍA DE PROYECTOS

ELABORADO POR:
Responsable: Javier Cruz / Osdalis Medina
Fecha: 14/12/2016
Firma:

REVISADO POR:
Responsable: Jose Pacheco
Fecha:
Firma: 14/12/16

APROBADO POR:
Responsable: Guillermo Pacheco
Fecha:
Firma: 14/12/16

VERIFICADO POR:
Responsable: Lisset Bolaños
Fecha: 14/12/2016
Firma:

PLT-CDC-002 Versión 02.00