UNIVERSIDAD NACIONAL DEL ALTIPLANO - PUNO

FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

Sistema de gestión documental para la Oficina de Cooperación Nacional e Internacional de la Universidad Nacional del Altiplano Puno – 2017

TESIS

PRESENTADA POR:

Bach. YÉSICA MAGALY RAMÍREZ ESTRELLA

Bach. JOSEPH JOSAFAT RAMOS ARPASI

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ESTADÍSTICO E INFORMÁTICO

PUNO – PERÚ

2017

UNIVERSIDAD NACIONAL DEL ALTIPLANO - PUNO

FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA
ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

Sistema de gestión documental para la Oficina de Cooperación Nacional e Internacional de la Universidad Nacional del Altiplano Puno - 2017

TESIS

PRESENTADA POR:

Bach. Yésica Magaly RAMÍREZ ESTRELLA
Bach. Joseph Josafat RAMOS ARPASI

Para optar el Título Profesional de:

INGENIERO ESTADÍSTICO E INFORMÁTICO

APROBADA POR:

PRESIDENTE DEL JURADO : Dr. Juan R. Paredes Quispe

PRIMER MIEMBRO : M.Sc. Ernesto Nayer Tumi Figueroa

SEGUNDO MIEMBRO : M.Sc. Nestor Tipula Quispe

DIRECTOR DE LA TESIS : Dr. Vladimiro Ibañez Quispe

ASESOR DE LA TESIS : Mg. Leonid Alemán Gonzales

ÁREA: Informática
TEMA: Sistemas de información
FECHA DE SUSTENTACIÓN: 09/11/17

DEDICATORIAS

A Dios por acompañarme en cada paso de mi vida, por brindarme la oportunidad de obtener otra meta personal, y darme salud, sabiduría y entendimiento para lograr esta meta.

Con todo cariño y amor para mis padres Liceria y Juan, que han velado por mi bienestar y mi educación para que yo pueda lograr mis sueños y metas, depositando su entera confianza en cada reto que se me presentaba sin dudar un solo momento, por motivarme y darme la mano cuando la necesitaba, es por ellos que soy la persona que soy ahora, a ustedes por siempre mi agradecimiento.

A mis hermanos Noelia, Rina, Rosmery, Juan y Henry que en este andar por la vida sus palabras y consejos me ayudan a ser mejor día a día.

Yésica Magaly Ramírez Estrella

Al ser más supremo del universo, por concederme vida y sabiduría.

Con respeto y admiración a mis padres: Rosa y Francisco por su invalorable amor y sacrificio, y su constante apoyo incondicional en mi formación humana y profesional.

A mis hermanos por el aliento constante en mi formación profesional.

Joseph Josafat Ramos Arpasi

AGRADECIMIENTOS

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

A los catedráticos de la Escuela Profesional de Ingeniería Estadística e Informática por brindarnos sus conocimientos en nuestra formación Profesional, a la Oficina de Cooperación Nacional e Internacional por permitirnos desarrollar el trabajo de investigación, a las personas que nos apoyaron en la elaboración de esta tesis el Dr. Vladimiro Ibáñez Quispe y el M.Sc. Leonid Aleman Gonzales.

A nuestros compañeros de la Facultad de Ingeniería Estadística e Informática, y en especial a nuestros amigos de Escuelab Puno por sus preciados consejos, gratos momentos y vuestro generoso apoyo. Siempre estaremos en deuda con todos.

A nuestros compañeros de trabajo en Taller con los cuales compartimos conocimientos y experiencias, lo que constituye un aliento y ánimo para la realización del presente trabajo.

ÍNDICE

ÍNDICE DE CUADROS

ÍNDICE DE FIGURAS

RESUMEN

La investigación se desarrolló en la Oficina de Cooperación Nacional e Internacional de la Universidad Nacional del Altiplano - Puno. El objetivo principal fue evaluar la gestión documental de la oficina a través de un sistema, que les permitió mejorar las deficiencias en su gestión de documentos a través de la automatización de procesos y actividades como: el acceso y búsqueda de convenios; la publicación y difusión de convocatorias, programas de movilidad, becas y otros; que se realizan en dicha oficina. Para el desarrollo del sistema se consideró un modelo que combina la metodología ágil Extreme Programming, el framework de administración de proyectos Scrum y los principios de la filosofía Lean Software Development. Además, para alcanzar el objetivo principal de la investigación, se analizó la satisfacción de los usuarios respecto a la facilidad del sistema y la utilidad del sistema por el personal de la oficina desde el punto de vista organizacional y funcional. Los resultados de la evaluación determinaron que el nivel de satisfacción de los usuarios fue alto, alcanzando una media de 28.1 puntos, y de igual manera el nivel de utilidad del sistema también fue alto, obteniendo una media de 28 puntos, en una escala de 7 a 35 puntos para ambos casos. En conclusión, se confirmó que la implantación del sistema mejoró favorablemente la gestión documental de la Oficina de Cooperación Nacional e Internacional.

Palabras clave: Investigación, Gestión documental, Extreme Programming, Scrum, Lean Software Development, Sistema implantado.

ABSTRACT

The research was development in the National and International Cooperation Office at the National University of the Altiplano - Puno. The main objective was to evaluate the documentary management of the office through a system, which would allow them to improve the deficiencies in their document management through the automation of processes and activities such as: access and search for agreements; the publication and dissemination of announcements, mobility programs and scholarships; and others that realize in said office. For the development of the system it considered a model that combines the agile Extreme Programming methodology, the management projects framework Scrum and Lean principles software development philosophy. In addition, in order to achieve the main objective of the research, the satisfaction of the users regarding the ease of the system and the usefulness of the system by the staff of the office from the organizational and functional point of view was analyzed. The results of the evaluation determined that the level of the users' satisfaction was high, reaching an average of 28.1 points, and similarly the level of the utility of the system was also high, obtaining an average of 28 points on a scale of 7 to 35 points for both cases. In conclusion, it was confirmed that the implementation of the system favorably improved the document management of the National and International Cooperation Office.

Key words: Investigation, Document Management, Extreme Programming, Scrum, Lean Software Development, Efficacy, Implanted system.

INTRODUCIÓN

La agilidad en los procesos de negocio y el incremento de la productividad son preocupaciones fundamentales para cualquier organización pública, privada y sin fines de lucro. En un ambiente donde el volumen de documentos crece cada vez más, la gestión de documentos necesita de un tiempo significativo. Además, si los objetivos principales de la organización dependen directamente de ello, el impacto en la eficiencia organizacional es aún mayor.

La Oficina de Cooperación Nacional e Internacional es un órgano de apoyo a las funciones que cumple Rectorado, y tiene como objetivos mejorar la gestión en los procesos de internacionalización, orientar y apoyar las acciones destinadas a la utilización de los convenios; difundir información sobre convocatorias, becas y ofertas académicas. Además, asesorar a los estudiantes y docentes postulantes a los programas de movilidad. En dicha oficina se identificó deficiencias en la gestión documental respecto a la recuperación de información, acceso de documentos, generación de reportes, publicación y difusión de información oportuna.

Como solución a la problemática planteada se decidió desarrollar un Sistema de Gestión Documental que automatice los procesos repetitivos, permita una distribución más eficiente y control sobre la información, archivos y registros, proporcionando una búsqueda y recuperación de documentos más rápido; autonomía en la difusión, como también una mejora general en el proceso de trabajo y la eficiencia organizacional.

Además, para determinar de qué manera la implantación del sistema mejoraría favorablemente la gestión documental se definió como objetivo general de la investigación evaluarla a través de un sistema, analizando los principales procesos y actividades de gestión documental de la oficina utilizando como base el modelo propuesto por Franco (2007); también evaluando la satisfacción de los usuarios respecto a la facilidad de uso del sistema y la utilidad del sistema desde el punto de vista organizacional y funcional por parte del personal de la oficina, basado en el cuestionario propuesto por Andrade (2001).

CAPÍTULO I

PLAN DE INVESTIGACIÓN

1.1. EL PROBLEMA

1.1.1. Descripción del problema

De acuerdo con el Reglamento de Organizaciones y Funciones de la Universidad Nacional del Altiplano la Oficina de Cooperación Nacional e Internacional, tiene como misión la internacionalización de la Universidad Nacional del Altiplano - Puno.

A la oficina se le encomienda la gestión de las diversas relaciones con el ámbito exterior conectándose con las instituciones, entidades y organismos internacionales, favoreciendo la movilidad y la cooperación internacional, completando la formación de estudiantes y profesores, desarrollando y potenciando sus relaciones internacionales.

La Oficina de Cooperación Nacional e Internacional de la Universidad Nacional del Altiplano, es la dependencia encargada de orientar, proponer, gestionar, y canalizar los recursos de cooperación científico tecnológica, nacional e internacional, establecer y mantener relaciones de la Universidad con entidades públicas y privadas; y está conformada por las siguientes unidades:

- Unidad de Convenios Nacionales e Internacionales.

- Unidad de Movilidad de Estudiantes, Docentes, Investigadores y Personal Administrativo.
- Unidad de Becas Nacionales e Internacionales.

La unidad de Convenios Nacionales e Internacionales, lleva el registro de convenios suscritos a través de hojas de cálculo y directorios físicos; esto genera que las solicitudes de información de convenios suscritos hechas por los decanos de las facultades, directores de estudio de las escuelas profesionales, entidades públicas y/o privadas no sean atendidos oportunamente, porque aparte que se tiene un promedio de más de 700 convenios suscritos, su búsqueda es manual.

Es importante resaltar que al no contar con un registro estandarizado y/o base de datos de convenios que permita su organización y actualización continua; la búsqueda es más dificultosa porque estos vienen aumentando con el pasar del tiempo.

Por otra parte, la unidad de Becas Nacionales e Internacionales y la unidad de Movilidad no cuentan con una plataforma que le permita la publicación de información de becas, programas de movilidad de estudiantes, docentes e investigadores; impidiendo su difusión y socialización oportuna, del mismo modo ocurre con las convocatorias de los programas de movilidad del Consejo de Rectores por la Integración de la Subregión Centro Oeste de Sudamérica (CRISCOS), Red Interuniversitaria del Sur del Perú (REDISUR) y Alianza del Pacífico; esto se evidencia en la baja cantidad de postulantes. El único medio de difusión con el que cuentan es el sitio web de la universidad; pero muchas veces no cumple con las expectativas que

se requiere por la burocracia existente en la Oficina de Tecnología e Información (OTI).

Y finalmente se debe que tener en cuenta que la Oficina de Cooperación Nacional e Internacional es el reflejo de la universidad que se muestra al exterior.

Motivo por el cual se necesita un sistema de gestión documental, que permita sistematizar los diferentes procesos que se realizan en la Oficina de Cooperación Nacional e Internacional, el cual será desarrollado utilizando principios de la filosofía Lean Software Development, el framework de administración de gestión de proyectos Scrum, la metodología ágil XP, con tecnologías y herramientas de última generación como Drupal 8, React, AngularJS, MongoDB, Choko, Node.js, Git, SemaphoreCI, Docker y Trello.

1.1.2. Formulación del problema

¿De qué manera la implantación de un sistema mejorará favorablemente en la gestión documental de la Oficina de Cooperación Nacional e Internacional de la Universidad Nacional del Altiplano - Puno?

1.2. OBJETIVOS

1.2.1. Objetivo General

Evaluar la gestión documental de la Oficina de Cooperación Nacional e Internacional de la Universidad Nacional del Altiplano a través de un sistema, Puno - 2017.

1.2.2. Objetivos Específicos

- Analizar los principales procesos y actividades de las unidades de la Oficina de Cooperación Nacional e Internacional.
- Analizar la satisfacción de los usuarios respecto al sistema implantado.
- Analizar la utilidad del sistema implantado, desde la perspectiva del personal de la oficina.

1.3. HIPÓTESIS

La implantación del sistema mejorará favorablemente en la gestión documental de la Oficina de Cooperación Nacional e Internacional de la Universidad Nacional del Altiplano – Puno.

1.4. JUSTIFICACIÓN DE LA INVESTIGACIÓN

Con el fin de mejorar la gestión documental se pretende con el presente trabajo desarrollar un Sistema de Gestión Documental, el cual será importante porque reducirá en gran medida los problemas que se presentan actualmente; automatizando procesos repetitivos.

El sistema tendrá registrados gran parte de los convenios suscritos; en consecuencia, la recuperación de documentos y las búsquedas serán en menor tiempo. Además, el responsable de la Unidad de Convenios podrá organizar mejor los convenios existentes, registrar nuevos convenios y generar informes detallados en menos tiempo. También tendrá autonomía en la publicación de los mismos.

Además, el responsable de la Unidad de Becas Nacionales e Internacionales y el responsable de la Unidad de Movilidad podrán hacer la difusión y publicación de información referente a convocatorias, becas, programas de movilidad de estudiantes y docentes con total autonomía; para que así los interesados tengan la información de manera oportuna.

Asimismo, los estudiantes, egresados, docentes y comunidad universitaria tendrán acceso a la plataforma de la Oficina de Cooperación Nacional e Internacional vía internet, donde encontrarán información acerca de convenios suscritos, publicación de becas, convocatorias, entre otras informaciones que sean de su completo interés.

1.5. LIMITACIONES DE LA INVESTIGACIÓN

El presente proyecto se enfocará en:

- Unidad de Convenios Nacionales e Internacionales
  - Sistematización de convenios suscritos.
- Unidad de Movilidad de Estudiantes, Docentes, Investigadores y Personal Administrativo.
  - Sistematización de la publicación y difusión de información referente a programas de movilidad.
- Unidad de Becas Nacionales e Internacionales.
  - Sistematización de la publicación y difusión de información referente a becas.

CAPÍTULO II

MARCO TEÓRICO

2.1. ANTECEDENTES DE LA INVESTIGACIÓN

ANDRADE (2001) define que un Sistema de Información (SI) se considera eficaz, a medida que se percibe como un elemento que contribuye al alcance de los objetivos organizacionales. De ahí la importancia de la visión de las reales necesidades de informatización debe ser compartida por todos y que, desde ahí sean definidos objetivos claros con transparencia y sobre todo con coherencia.

FRANCO (2007) propone un modelo de gestión de proyectos basado en las metodologías ágiles de desarrollo de software y en los principios de producción Lean. Y concluye que el modelo propuesto combinó dos metodologías que integran la corriente de las Metodologías Ágiles de desarrollo de software (Extreme Programming y Scrum). Esa combinación proporcionó un mayor detalle y una definición más clara y objetiva de las actividades relacionadas al desarrollo de software y a la administración de proyectos, facilitando su implementación para la realización del estudio de caso.

MARTÍN (2010) evaluó la exposición del estado del arte actual de la aplicación de los principios de la *Filosofía Lean* a la *Ingeniería del Software*. Y concluye que la literatura revisada pone de manifiesto el creciente interés en los últimos años por extrapolar las técnicas y prácticas de la filosofía Lean a la Ingeniería del

Software. Además, que la aplicabilidad de las técnicas y prácticas Lean en esta disciplina se pone de manifiesto y los resultados obtenidos en las experiencias son notablemente positivos.

QUISPE (2014) implementó un Sistema Informático para la *gestión administrativa* de la Coordinación de Investigación de la Facultad de Ingeniería Estadística e Informática de la UNA – Puno 2014, quien concluyó que la implementación del sistema SIGACI permite realizar los procesos descritos en menor tiempo.

2.2. BASE TEÓRICA

2.2.1. Objetivos y Funciones de la Oficina de Cooperación Nacional e Internacional

La Oficina de Cooperación Nacional e Internacional es la encargada de proponer, gestionar y canalizar los recursos de cooperación científico – tecnológica, nacional e internacional, de establecer y mantener relaciones colaborativas de la Universidad con entidades públicas y privadas. Está bajo la jefatura de un Profesor Principal Ordinario, designado por Consejo Universitario de una terna propuesta por el Rector para un periodo de tres años, con ratificaciones anuales. Son funciones del Jefe de la Oficina de Cooperación Nacional e Internacional:

- Planear, proponer y gestionar los convenios de cooperación técnica nacional e internacional, en concordancia con el Plan Estratégico de Desarrollo Universitario.
- Proponer la normatividad pertinente para el desarrollo de los convenios de cooperación nacional e internacional.

- Realizar el seguimiento y control del desarrollo y la ejecución de los convenios suscritos por la Universidad.
- Velar por el estricto cumplimiento de los términos de los convenios y otros suscritos por la institución.
- Informar trimestralmente al Rectorado sobre el cumplimiento de sus actividades.
- Mantener actualizado la documentación y formatos de solicitudes de becas otorgadas por entidades nacionales y extranjeras.

2.2.2. Sistema de información

Según Laudon (2004), un sistema de información es un organismo que recolecta, procesa, almacena y distribuye información. Son indispensables para ayudar a los gerentes a mantener ordenada su compañía, a analizar todo lo que por ella pasa y a crear nuevos productos que coloquen en un buen lugar a la organización. Esta definición es una de las únicas que manifiesta la exigencia de que un sistema de información tenga componentes, aunque no especifica cuáles deban ser, posiblemente porque intenta englobar todas las posibles variantes de este concepto. El objetivo primordial de un sistema de información es apoyar la toma de decisiones y controlar todo lo que en ella ocurre. Es importante señalar que existen dos tipos de sistema de información, los formales y los informales; los primeros utilizan como medio para llevarse a cabo estructuras sólidas como ordenadores, los segundos son más artesanales y usan medios más antiguos como el papel y el lápiz.

Un Sistema de Información realiza cuatro actividades básicas:

- Entrada de información: proceso en el cual el sistema toma los datos que requiere.
- Almacenamiento de información: puede hacerse por computadora o archivos físicos para conservar la información.
- Procesamiento de la información: permite la transformación de los datos fuente en información que puede ser utilizada para la toma de decisiones
- Salida de información: es la capacidad del sistema para producir la información procesada o sacar los datos de entrada al exterior.

Los usuarios de los sistemas de información tienen diferente grado de participación dentro de un sistema y son el elemento principal que lo integra, así se puede definir usuarios primarios quienes alimentan el sistema, usuarios indirectos que se benefician de los resultados pero que no interactúan con el sistema, usuarios gerenciales y directivos quienes tienen responsabilidad administrativa y de toma de decisiones con base a la información que produce el sistema.

2.2.3. Gestión

Se denomina gestión al correcto manejo de los recursos de los que dispone una determinada organización, por ejemplo, empresas, organismos públicos, organismos no gubernamentales y otros. El término gestión puede abarcar una larga lista de actividades, pero siempre se enfoca en la utilización eficiente de estos recursos, en la medida en que debe maximizarse sus rendimientos. (Cantú, 2006)

Este concepto se utiliza para hablar de proyectos o en general de cualquier tipo de actividad que requiera procesos de planificación, desarrollo, implementación y control. (Cantú, 2006)

Según Gauchi Risso (2012) el término Gestión se denomina preliminarmente al desarrollo de cualquier tipo de formalización teórica; resulta imprescindible definir el término gestión, puesto que todos los conceptos implicados en este trabajo hacen referencia a la gestión en sus diferentes prácticas y/o ámbitos de aplicación. El diccionario de la Real Academia Española (RAE) define gestión como el: «conjunto de trámites que se llevan a cabo para resolver un asunto»; como segunda opción, se refiere a la: «dirección, administración de una empresa, negocio, entre otros.» Ambas definiciones hacen referencia a la acción y al efecto de administrar. A la luz de la definición, el término gestión parecería encontrarse equiparado al de administración, sin embargo y tal como se concibe actualmente la gestión dentro de la perspectiva de las prácticas organizativas, ésta conforma un conjunto de actividades de decisión que tienen lugar dentro de una organización y es aplicada como un conjunto de procedimientos de adecuación de recursos de cualquier índole a aquellos fines para los cuales han sido obtenidos los recursos.

Por tanto, gestionar es hacer que las decisiones se ejecuten, tramitar asuntos con vistas a la obtención de unos resultados, involucra el conjunto de proposiciones teóricas que explican el uso de unas reglas, procedimientos y modos operativos para llevar a cabo con eficacia las actividades económicas que permiten lograr los objetivos de una

organización. Con relación a la dirección, la gestión, es mucho más restringida, puesto que la dirección abarca un proceso más general por el cual se conduce a una organización hacia sus objetivos, mediante la aplicación de los factores disponibles, y desarrollando las funciones de: planificación, organización, gestión y control; principios directivos que sirven para toda clase de instituciones.

2.2.4. Gestión Documental (GD)

Antiguamente conocido también como “archiveros”. Roberge (2006) define la Gestión Documental como: «el conjunto de operaciones y técnicas relativas a la concepción, al desarrollo, a la implantación y a la evaluación de los sistemas administrativos necesarios, desde la creación de los documentos hasta su destrucción o su transferencia a los archivos». Sin embargo, la GD actualmente comprende algo más que una readecuación y renovación de la tarea de los archiveros, puede considerarse como un proceso vital para toda organización debido a la magnitud que alcanza la producción de documentos, que permiten analizar y controlar sistemáticamente cómo la información registrada se crea, recibe, mantiene o utiliza.

Una característica de la GD es su transversalidad en toda una organización, poniendo énfasis en la vinculación con todas las actividades y otorgando una alta prioridad al valor primario del documento, es decir, a la utilidad de la documentación para la acción del quehacer organizacional; sin lugar a dudas esta concepción varió sensiblemente el desempeño del archivero ampliando sus tareas desde el momento en que se da origen al documento.

Si bien a menudo se asocia la GD con la implementación de un programa informático, no necesariamente debe ser así, en algunas ocasiones una solución de GD puede pasar por fijar unas pautas corporativas.

La integración de soluciones tecnológicas puede ser un elemento imprescindible, porque ya no se puede depender exclusivamente de las transferencias físicas o de las hojas de remisión para el control de la documentación, en la mayoría de los casos resulta necesaria la implantación de un sistema que ofrezca los datos necesarios sobre los documentos, desde que nacen, hasta que se guardan permanentemente o destruyen.

2.2.5. Sistema de gestión documental (SGD)

Según Gauchi Risso (2012), los Sistemas de Gestión Documental agrupan operaciones y técnicas de la gestión administrativa general con el objeto de coordinar y controlar todas aquellas funciones y actividades específicas que afectan la creación, recepción, ubicación, acceso y preservación de los documentos, protegiendo sus características estructurales y contextuales con el objeto de garantizar su autenticidad e integridad a lo largo del tiempo; la integración de los procesos y controles documentales en los procesos de trabajo debe ser el objetivo principal de cualquier modelo de gestión de documentos, porque permite: la disminución del tiempo de localización y recuperación de los documentos, la disminución, del espacio físico de almacenamiento, aumenta la seguridad mediante una política de back-up correcta y acrecienta la rapidez en la atención al cliente, incluso

actualmente existen soluciones open source que brindan funcionalidades similares a las soluciones comerciales.

Figura 1: Sistema de Gestión Documental

2.2.6. Metodologías ágiles

El desarrollo ágil del software es un marco de trabajo conceptual que fue promovido en febrero de 2001 después de una reunión en los Estados Unidos donde nace el término “Ágil” aplicado al desarrollo de software con nuevos enfoques metodológicos.

De esta reunión fueron miembros 17 expertos en ingeniería de software, los cuales definieron los valores y principios básicos que debe tener un sistema que se pretenda desarrollar implementando una metodología ágil y conservando la buena calidad de los productos; En esta reunión se creó *The Agile Alliance*, una organización sin fines de lucro, dedicada a promover los conceptos relacionados con el desarrollo ágil de software y ayudar a las organizaciones para que adopten dichos conceptos. El punto

de partida fue el manifiesto ágil, un documento que resume la filosofía ágil. (Zapata, 2012)

2.2.7. Manifiesto ágil

Según Zapata (2012), en el manifiesto se presentan cuatro valores a tener en cuenta en el desarrollo de software con buenas prácticas.

- **Los individuos e interacciones del equipo de desarrollo por encima de los procesos y las herramientas.** El recurso humano es el principal factor de éxito de un proyecto de software. Es más importante construir un buen equipo que construir el entorno. Muchas veces se comete el error de construir primero el entorno y esperar que el equipo se adapte automáticamente, pero es mejor crear el equipo y que éste configure su propio entorno de desarrollo en base a sus necesidades.
- **Desarrollar software que funciona por encima de conseguir una buena documentación.** La regla a seguir es no producir documentos a menos que sean necesarios de forma inmediata para tomar una decisión importante. Estos documentos deben ser cortos y centrarse en lo fundamental.
- **La colaboración del cliente por encima de la negociación de un contrato.** Se propone que exista una interacción constante entre el cliente y el equipo de desarrollo, desde el comienzo hasta la culminación del proyecto. Esta colaboración entre ambos será la que marque la marcha del proyecto y asegure su éxito.
- **Responder a los cambios por encima de seguir estrictamente un plan.** La habilidad de responder a los cambios que puedan surgir a lo

largo del proyecto (cambios en los requisitos, en la tecnología, en el equipo, otros.) determina también el éxito o fracaso del mismo. Por lo tanto, la planificación no debe ser estricta sino flexible y abierta.

Los valores anteriores inspiran los doce principios del manifiesto. Son características que diferencian un proceso ágil de uno tradicional. Los dos primeros principios son generales y resumen gran parte del espíritu ágil. El resto tienen que ver con el proceso a seguir y con el equipo de desarrollo, en cuanto a metas a seguir y organización del mismo. Los principios son:

- Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
- Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
- Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
- Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
- Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiables la ejecución del trabajo.
- El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
- El software funcionando es la medida principal de progreso.

- Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
- La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
- La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
- Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
- A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

2.2.8. Programación Extrema (XP)

Bautista (2009) define XP como una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

- *¿Qué es Programación Extrema o XP?*
  - Metodología liviana de desarrollo de software.

- Conjunto de prácticas y reglas empleadas para desarrollar software.
- Basada en diferentes ideas acerca de cómo enfrentar ambientes muy cambiantes.
- Originada en el proyecto C3 para Chrysler.
- En vez de planificar, analizar y diseñar para el futuro distante, hacer todo esto un poco cada vez, a través de todo el proceso de desarrollo.

- Objetivos

  - Establecer las mejores prácticas de Ingeniería de Software en el desarrollo de proyectos.
  - Mejorar la productividad de los proyectos.
  - Garantizar la Calidad del Software desarrollando, haciendo que este supere las expectativas del cliente.

- Contexto XP

  - Cliente bien definido.
  - Los requisitos pueden (y van a) cambiar.
  - Grupo pequeño y muy integrado (máximo 12 personas).
  - Equipo con formación elevada y capacidad de aprender.

- Características XP

  - Metodología basada en prueba y error.
  - Fundamentada en Valores y Prácticas.

- Expresada en forma de 12 Prácticas–Conjunto completo–Se soportan unas a otras–Son conocidas desde hace tiempo. La novedad es juntarlas.

- Valores XP
  - Simplicidad: XP propone el principio de hacer la cosa más simple que pueda funcionar, en relación al proceso y la codificación. Es mejor hacer hoy algo simple, que hacerlo complicado y probablemente nunca usarlo mañana.
  - Comunicación: Algunos problemas en los proyectos tienen origen en que alguien no dijo algo importante en algún momento. XP hace casi imposible la falta de comunicación.
  - Realimentación: Retroalimentación concreta y frecuente del cliente, del equipo y de los usuarios finales da una mayor oportunidad de dirigir el esfuerzo eficientemente.
  - Coraje: El coraje (valor) existe en el contexto de los otros 3 valores. (si funciona...mejóralo)

- El estilo XP
  - Está orientada hacia quien produce y usa el software.
  - Reduce el costo del cambio en todas las etapas del ciclo de vida del sistema.
  - Combina las que han demostrado ser las mejores prácticas para desarrollar software, y las lleva al extremo.

Prácticas Básicas de la Programación Extrema

Para que todo esto funcione, la programación extrema se basa en **doce prácticas básicas** que deben seguirse al pie de la letra. (www.xprogramming.com/xpmag/whatisxp.htm)

- **Equipo completo:** Forman parte del equipo todas las personas que tienen algo que ver con el proyecto, incluido el cliente y el responsable del proyecto.
- **Planificación:** Se hacen las historias de usuario y se planifica en qué orden se van a hacer y las mini-versiones. La planificación se revisa continuamente.
- **Test del cliente:** El cliente, con la ayuda de los desarrolladores, propone sus propias pruebas para validar las mini-versiones.
- **Versiones pequeñas:** Las mini-versiones deben ser lo suficientemente pequeñas como para poder hacer una cada semana. Deben ser versiones que ofrezcan algo útil al usuario final y no trozos de código que no pueda ver funcionando.
- **Diseño simple:** Hacer siempre lo mínimo imprescindible de la forma más sencilla posible. Mantener siempre sencillo el código.
- **Pareja de programadores:** Los programadores trabajan por parejas (dos delante del mismo ordenador) y se intercambian las parejas con frecuencia (un cambio diario).
- **Desarrollo guiado por las pruebas automáticas:** Se deben realizar programas de prueba automática y deben ejecutarse con mucha frecuencia. Cuantas más pruebas se hagan, mejor.

- **Integración continua:** Deben tenerse siempre un ejecutable del proyecto que funcione y en cuanto se tenga una nueva pequeña funcionalidad, debe recompilarse y probarse. Es un error mantener una versión congelada dos meses mientras se hacen mejoras y luego integrarlas todas de golpe. Cuando falle algo, no se sabe qué es lo que falla de todo lo que hemos metido.
- **El código es de todos:** Cualquiera puede y debe tocar y conocer cualquier parte del código. Para eso se hacen las pruebas automáticas.
- **Normas de codificación:** Debe haber un estilo común de codificación (no importa cuál), de forma que parezca que ha sido realizado por una única persona.
- **Metáforas:** Hay que buscar unas frases o nombres que definan cómo funcionan las distintas partes del programa, de forma que sólo con los nombres se pueda uno hacer una idea de qué es lo que hace cada parte del programa. Un ejemplo claro es el “recolector de basura” de java. Ayuda a que todos los programadores (y el cliente) sepan de qué estamos hablando y que no haya mal entendidos.
- **Ritmo sostenible:** Se debe trabajar a un ritmo que se pueda mantener indefinidamente; esto quiere decir que no debe haber días muertos en que no se sabe qué hacer y que no se deben hacer un exceso de horas otros días. Al tener claro semana a semana lo que debe hacerse, hay que trabajar duro en ello para conseguir el objetivo cercano de terminar una historia de usuario o mini-versión.

Las prácticas de XP son agrupadas en cuatro grupos de acuerdo con su finalidad, empezando del medio hasta el borde de la elipse: Codificación,

Equipo, Proceso y Producto. La presenta una ilustración gráfica de agrupación de las prácticas XP, descritas anteriormente.

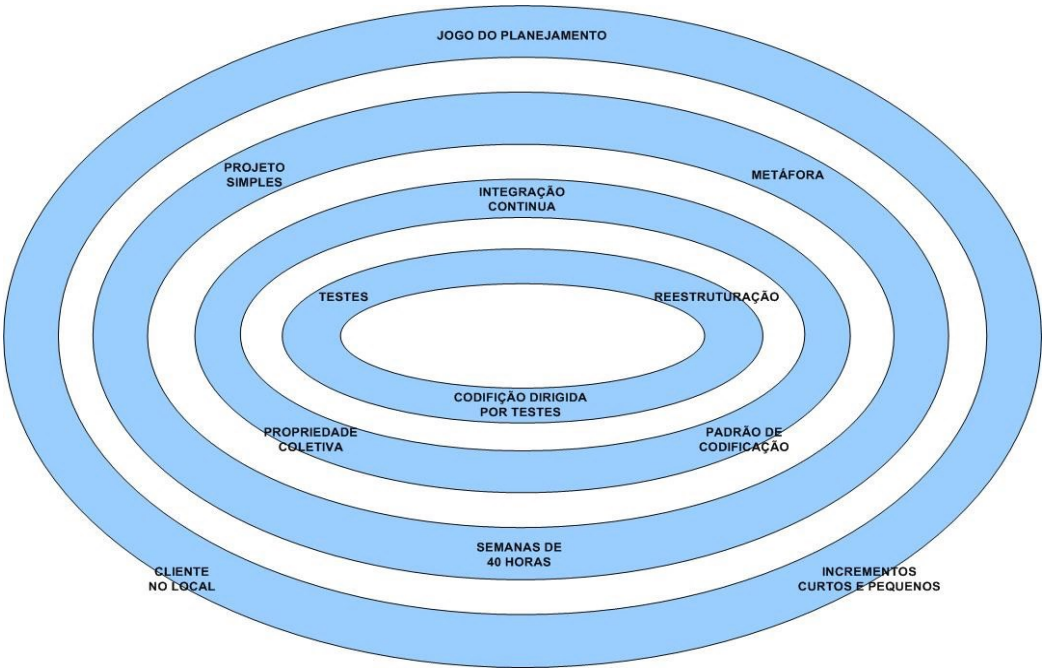

Figura 2: Argumento de las prácticas de XP

Figura 3: Ejemplo de un proyecto XP

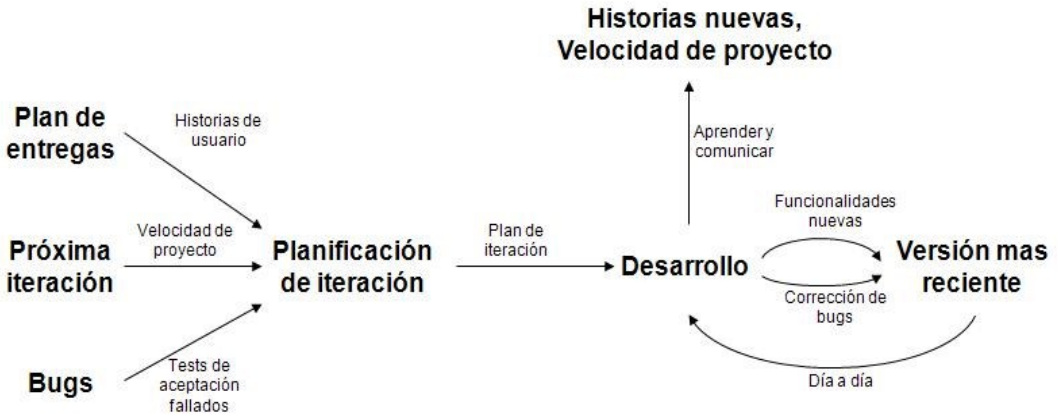

Figura 4: Ejemplo de Iteración XP

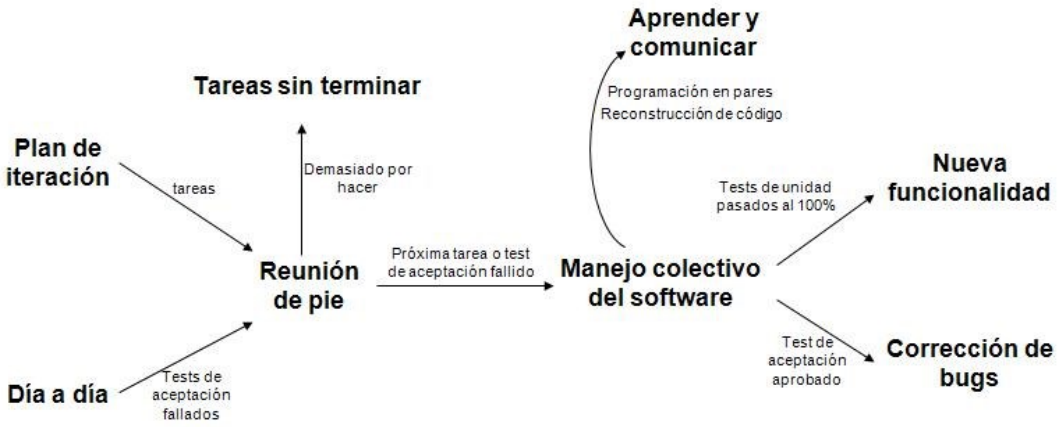

Figura 5: Ejemplo de desarrollo XP

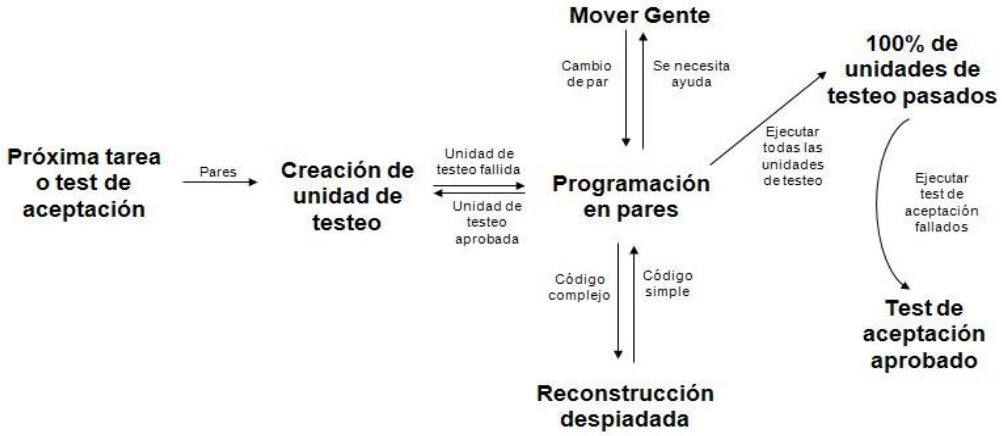

Figura 6: Manejo colectivo en XP

Ventajas y Desventajas de Programación Extrema

- Ventajas:
  - Programación organizada.
  - Menor taza de errores.
  - Satisfacción del programador.

- Desventajas:
  - Es recomendable emplearlo solo en proyectos a corto plazo.
  - Altas comisiones en caso de fallar.

- Conclusiones
  - Apostolado de metodologías exitosas.
  - Aporte de la experiencia práctica a los modelos teóricos.
  - Enfoque de conjunto de prácticas como rompecabezas.
  - Tecnología en expansión.
  - Importancia de revisitar las metodologías desde la experiencia práctica.

2.2.9. Scrum

Schwaber (2011) define Scrum como un marco de trabajo estructurado para dar soporte al desarrollo de productos complejos. Scrum consiste en los equipos Scrum y en sus roles, eventos, artefactos y reglas asociadas.

En 1995, Sutherland y Schwaber presentaron conjuntamente un artículo describiendo la metodología Scrum Sutherland (1995), de las metodologías ágiles, es la más utilizada, según una encuesta publicada por VersionOne realizada a 4048 profesionales del Software, la misma, revela que el 54%

de los encuestados, utiliza Scrum como metodología para la gestión de proyectos de desarrollo de Software.

Scrum se fundamenta en la teoría empírica de control de procesos, o empirismo. El empirismo asegura que el conocimiento procede de la experiencia y de tomar decisiones basándose en lo que se conoce. Scrum emplea una aproximación iterativa e incremental para optimizar la predictibilidad y controlar el riesgo.

Scrum permite la creación de equipos auto-organizados impulsando la colocalización de todos los miembros del equipo, y la comunicación verbal entre todos los miembros y disciplinas involucrados en el proyecto.

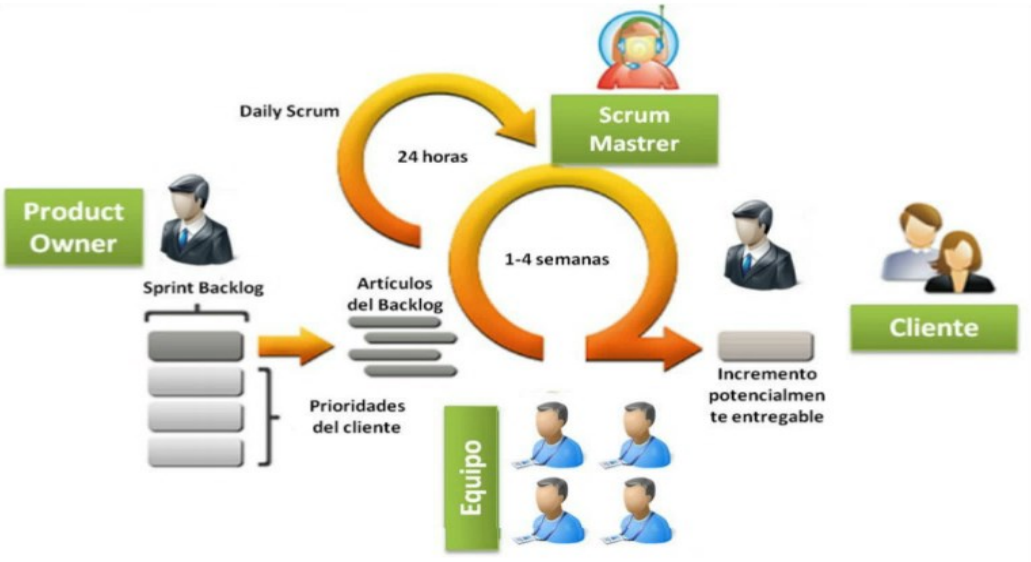

Figura 7: Ejemplo del flujo de trabajo y elementos de Scrum

En la Figura 7 se muestra un ejemplo del flujo de trabajo de Scrum y algunos de sus elementos, los cuales se describen a continuación:

Equipo Scrum

Existen seis roles en el Scrum que poseen tareas y propósitos diferentes durante el proceso y sus prácticas: Scrum master, responsable del producto, Equipo Scrum, cliente, usuario y gerente. A continuación, estos roles son descritos de acuerdo con las siguientes definiciones. (Schwaber y Beedle, 2001).

- **Responsable del producto (Product Owner):** Es oficialmente responsable por el proyecto, administración, control y por hacer visible la lista de funcionalidades del producto. Él es seleccionado por el Scrum master, clientes y gerente. Él también es responsable por tomar las decisiones finales referente a las tareas necesarias para transformar la lista de funcionalidades en el producto final, participando en la estimación del esfuerzo del desarrollo necesario y es responsable de las informaciones referente a la lista de funcionalidades utilizada por el Equipo *Scrum*.
- **Scrum Master (o Facilitador):** Es el responsable para garantizar que el proyecto esté siendo conducido de acuerdo con las prácticas, valores y reglas definidas en el *Scrum* y que el progreso del proyecto esté de acuerdo con lo deseado por los *Clientes*. El *Scrum Master* interactúa tanto con el *Equipo Scrum*, como con los *Clientes* y el *Gerente* durante el proyecto. Él también es responsable por remover y alterar cualquier obstáculo a lo largo del proyecto, para garantizar que el equipo trabaje de forma más productivamente posible.
- **Equipo Scrum (Scrum Team):** Es el equipo del proyecto que posee autoridad de decidir sobre las acciones necesarias y de organizarse

para poder alcanzar los objetivos pre-establecidos. El Equipo Scrum es incluida, por ejemplo, en la estimación del esfuerzo, en la creación y revisión de la lista de funcionalidad del producto, sugiriendo obstáculos que necesitan ser removidos del proyecto.

- **Cliente (Client):** Participa de las tareas relacionadas a la definición de la lista de funcionalidades del software que está siendo desarrollado o mejorado, elaborando los requisitos y restricciones del producto final deseado.
- **Gerente (Management)**: Es el encargado para tomar las decisiones finales, utilizando las informaciones visuales disponibilizadas gráficamente por los padrones y convenciones a seguir en el proyecto. Él también es responsable por acordar, junto a los clientes, los objetivos y requisitos del proyecto.

Eventos

- **Daily Scrum:** Es una reunión restringida a un bloque de tiempo de 15 minutos, para que el equipo de Desarrollo sincronice sus actividades y cree un plan para las siguientes 24 horas. Esto se lleva a cabo inspeccionando el trabajo avanzado desde el último Scrum Diario y haciendo una predicción acerca del trabajo que podría ser completado antes del siguiente.

  Es mantenido a la misma hora y en el mismo lugar todos los días, para reducir la complejidad. Durante la reunión, cada miembro del Equipo de Desarrollo explica: ¿Qué se ha conseguido desde la última reunión?,

¿Qué se hará antes de la próxima reunión? Y ¿Qué obstáculos se encuentran en el camino?

- **Sprint:** El corazón de Scrum es el Sprint, un bloque de tiempo (time-box) de un mes o menos durante el cual se crea un incremento de producto "Hecho", utilizable y potencialmente entregable. La duración de los Sprints es consistente a lo largo del esfuerzo de desarrollo. Cada nuevo Sprint comienza inmediatamente después de la finalización del Sprint previo. Los Sprints contienen y consisten en la Reunión de Planificación del Sprint (Sprint Planning Meeting), los Scrums Diarios (Daily Scrums), el trabajo de desarrollo, la Revisión del Sprint (Sprint Review), y la Retrospectiva del Sprint (Sprint Retrospective). Durante el Sprint:
  - No se realizan cambios que afectarían al Objetivo del Sprint (Sprint Goal).
  - La composición del Equipo de Desarrollo se mantiene constante.
  - Los objetivos de calidad no disminuyen;
  - El alcance puede ser clarificado y renegociado entre el Dueño de Producto y el Equipo de Desarrollo a medida que se va aprendiendo más.
- **Reunión de Planificación del Sprint (Sprint Planning Meeting):** El trabajo a realizar durante el Sprint es planificado en la Reunión de Planificación del Sprint. Este plan es creado mediante el trabajo colaborativo del Equipo Scrum al completo. La Reunión de Planificación del Sprint está restringida a una duración de ocho horas para un Sprint de un mes. Para Sprints más cortos, el evento es proporcionalmente

más corto. Por ejemplo, los Sprints de dos semanas tienen una Reunión de Planificación de Sprint de cuatro horas.

- **Revisión de Sprint (Sprint Review):** Al final del Sprint se lleva a cabo una Revisión de Sprint, para inspeccionar el Incremento y adaptar la Pila de Producto si fuese necesario. Durante la Revisión de Sprint, el Equipo Scrum y los interesados colaboran acerca de lo que se ha hecho durante el Sprint. Basándose en eso, y en cualquier cambio a la Pila de Producto hecho durante el Sprint, los asistentes colaboran para determinar las siguientes cosas que podrían hacerse. Se trata de una reunión informal, y la presentación del Incremento tiene como objetivo facilitar la retroalimentación de información y fomentar la colaboración.
- **Retrospectiva del Sprint (Sprint Retrospective):** La Retrospectiva de Sprint es una oportunidad para el equipo Scrum de inspeccionarse a sí mismo, y crear un plan de mejoras que sean abordadas durante el siguiente Sprint. La Retrospectiva de Sprint tiene lugar después de la Revisión de Sprint y antes de la siguiente Reunión de Planificación del Sprint. Se trata de una reunión restringida a un bloque de tiempo de tres horas para Sprints de un mes. Para Sprints más cortos se reserva un tiempo proporcionalmente menor.

Artefactos

- **Product backlog:** Es una lista ordenada de todo lo que podría ser necesario en el producto, y es la única fuente de requerimientos para cualquier cambio a realizarse en el producto. El Dueño de Producto

(Product Owner) es el responsable del product backlog, incluyendo su contenido, disponibilidad y ordenación.

- **Sprint backlog:** Es el conjunto de elementos de la product backlog seleccionados para el Sprint, más un plan para entregar el Incremento de producto y conseguir el Objetivo del Sprint. El Sprint backlog es una predicción hecha por el Equipo de Desarrollo acerca de qué funcionalidad formará parte del próximo Incremento y del trabajo necesario para entregar esa funcionalidad.

2.2.10. Lean Development y Lean Software Development

Según Franco (2007) Lean Development fue iniciado por Bob Charette y se inspira en el éxito del proceso industrial Lean Manufacturing, bien conocido en la producción automotriz y en manufactura desde la década de 1980. Este proceso tiene como precepto la eliminación de residuos a través de la mejora constante, haciendo que el producto fluya a instancias del cliente para hacerlo lo más perfecto posible.

Los procesos a la manera americana corrían con sus máquinas al 100% de capacidad y mantenían inventarios gigantescos de productos y suministros; Toyota, en contra de la intuición, resultaba más eficiente manteniendo suministros en planta para un solo día, y produciendo sólo lo necesario para cubrir las órdenes pendientes. Esto es lo que se llama Just in Time Production (JIT). Con JIT se evita además que el inventario degrade o se torne obsoleto, o empiece a actuar como un freno para el cambio.

Otros aspectos esenciales de Lean Manufacturing son la relación participativa con el empleado y el trato que le brinda la compañía, así como una especificación de principios, disciplinas y métodos iterativos, adaptativos, auto-organizativos e interdependientes en un patrón de ciclos de corta duración que tiene algo más que un aire de familia con el patrón de procesos de los Métodos Ágiles. Existe unanimidad de intereses, consistencia de discurso y complementariedad entre las comunidades Lean de manufactura y desarrollo de software (Reynoso, 2004).

Principios

Aunque Charette fue quien desarrollo el método, no hay mucha bibliografía escrita por él en la actualidad, por lo que se van a desarrollar los principios de Lean según describen Tom y Mary Poppendieck (2003, 2006) en su libro, tomado de la traducción realizada en (De Seta, 2009).

1. **Eliminar el Desperdicio**

   - *Brindar un liderazgo técnico y de mercado*: La organización puede ser exitosa si produce productos innovadores y tecnológicamente avanzados, pero es importante comprender lo que valoran los clientes y conocer la tecnología que se está usando.

   - *Crear solamente cosas de valor:* Hay que ser cuidadoso con todos los procesos que se siguen. Por ejemplo, debe asegurarse que todos los procesos son útiles y están enfocados en crear valor.

- *Escribir menos código:* Mientras más código se tenga, más pruebas se van a necesitar, por lo que se necesitará más trabajo. Si se escriben pruebas para funcionalidad que no se necesitan, está perdiendo el tiempo.

  Los principales tipos de desperdicios son:

  a. Funcionalidad que se desarrolla y no se utiliza en producción: Esto genera código extra, el cual tuvo que seguirse, compilarse y testearse, el mismo incrementa la complejidad y es un posible punto de fallo.

  b. La documentación: Consume recursos, demoras, se pierde y se vuelve obsoleta. Si se utiliza, que sea solo la necesaria.

  c. Asignar una misma persona a múltiples proyectos: Cada vez que el desarrollador de software tiene que cambiar entre tareas, pierde un tiempo significativo para recordar de que se trataba el proyecto.

     Pertenecer a diferentes equipos de trabajo causa más interrupciones que beneficios.

  d. Las Esperas: Uno de los grandes desperdicios, en cuanto a tiempo, es esperar a que sucedan cosas: se espera al comienzo del proyecto, en la definición de requerimientos, en el armado de la excesiva documentación, en la codificación, en revisión y aprobación, en testeo, etc.

e. Puesta en marcha: Cuando un desarrollador tiene una pregunta, ¿Cuánto cuesta encontrar la respuesta? Los requerimientos van a los analistas, de los analistas a los diseñadores, de aquí a los desarrolladores y luego al testeo, en cada pasaje es probable que se pierda algo, ya que un porcentaje de conocimiento tácito queda con el creador del documento y no se transmite.

f. Errores en el Software: El porcentaje de desperdicio causado por un error se puede medir como el impacto del mismo por el tiempo sin ser detectado. Hay que encontrarlos en cuanto ocurran, corregirlos, testearlos y actualizar la versión en producción.

2. Crear Conocimiento

- *Crear equipos de diseño y construcción:* El líder del equipo de desarrollo tiene que escuchar a los miembros y hacerles preguntas inteligentes que los incite a buscar respuestas y volver lo más pronto posible con los problemas que surgen, o con las soluciones inventadas.

- *Mantener una cultura de mejora continua:* Crear un ambiente en donde las personas estén mejorando continuamente en lo que trabajan, deben saber que no son y no deben ser perfectas, y que siempre tienen algún área que pueden mejorar.

- *Enseñar métodos de resolución de problemas:* Los equipos de desarrollo deberían comportarse como pequeños centros de

investigación, estableciendo hipótesis y realizando varios experimentos rápidos para verificar su validez.

3. Construir con Calidad

- *Código Prueba y Error usando Test Driven Development:* Escribir especificaciones ejecutables en lugar de requerimientos.

- *Automatizar:* Automatizar las pruebas, la construcción, las instalaciones, y cualquier cosa que sea rutinaria. Hay que automatizar de una manera inteligente, de forma que las personas puedan mejorar el proceso y cambiar cualquier cosa que quieran sin preocuparse por si el cambio hace que las cosas dejen de funcionar.

- *Refactorizar:* Eliminar la duplicación. Cada vez que aparezca la oportunidad, realizar la refactorización del código, de las pruebas y de la documentación para minimizar la complejidad.

4. Postergar el Compromiso

- *Agendar las decisiones irreversibles hasta el último momento responsable:* Debe saber hacia dónde quiere ir pero no conoce el camino del todo, lo va descubriendo día a día, lo más importante es mantener la dirección correcta.

- *Romper con las dependencias:* Los componentes deben estar lo más desacoplados posibles para que puedan implementarse en cualquier orden

- *Mantener opciones:* Desarrollar múltiples soluciones para todas las decisiones críticas y ver cuales funcionan mejor.

5. Optimizar el Total

- *Enfocarse en el flujo completo de valor:* Enfocarse en ganar la carrera completa (que es el software). No hay que gastar esfuerzo en optimizar ineficiencias locales, sino en ver el todo y optimizar a la organización en su totalidad.

- *Entregar un producto completo:* Los equipos necesitan tener buenos líderes, y también buenos ingenieros, vendedores, especialistas de marketing, secretarias, entre otros. Todos juntos pueden entregar un gran producto final a los clientes.

6. Entregar Rápido

- *Trabajar en bloques pequeños:* Reducir el tamaño del proyecto, acortar los ciclos de entrega, estabilizar el ambiente de trabajo, repetir lo bueno y erradicar las prácticas que crean obstáculos.

- *Limitar el trabajo a la capacidad:* Limitar la cola de tareas al mínimo (una o dos iteraciones por delante es suficiente), no hay que tener miedo al quitar elementos de la cola, rechazar cualquier trabajo hasta que se haya vaciado un lugar en la cola.

- *Enfocarse en el tiempo del ciclo, no en la utilización*: Agregar tareas pequeñas a la cola que no puedan atascar al proceso por un tiempo

largo, reducir el tiempo del ciclo y tener pocas cosas para procesar en la cola.

7. Respetar a las Personas

- *Capacitar a los líderes de equipo:* Darles a los líderes de equipo entrenamiento, guías y espacio libre para implementar el pensamiento Lean en su ambiente.
- *Mover la responsabilidad y la toma de decisiones al nivel más bajo posible*: Dejar que las personas piensen y decidan por su cuenta, ellos saben mejor que nadie cómo implementar algoritmos difíciles y aplicar tecnologías de última generación.
- *Fomentar orgullo por el trabajo:* Fomentar la pasión y la participación del equipo hacia lo que hacen y cómo lo hacen.

La aplicación de los principios Lean en el desarrollo de software permite mejorar los procesos y así obtener mejores resultados. Aumenta la calidad del producto, posibilitando bajar los costos y acortar los tiempos de desarrollo.

Lo importante es poder entender y adoptar la esencia de los principios Lean. Es claro que su aplicación puede ser difícil en algunas compañías, ya que requiere un cambio importante en la cultura y en los hábitos organizacionales. Pero las mejoras que se pueden lograr son importantísimas, no solo en el producto final sino también en su evolución, en la participación, compromiso y satisfacción de las personas involucradas.

Desperdicios en el Desarrollo de Software

Según Shingo (1981), uno de los ingenieros de sistemas de Producción de Toyota (TPS), identificó siete tipos de desperdicios en la fábrica. Tomando como base los desperdicios identificados en Toyota Poppendieck (2003), propusieron el mapeo ilustrado en el Cuadro 1, para el desarrollo de software.

Cuadro 1: Mapeo del desperdicio de la fábrica en el desarrollo de software

Fábrica	Desarrollo del Software
Stock	Trabajo parcialmente finalizado.
Procesamiento extra	Proceso extra.
Exceso de producción	Funcionalidades extras.
Transporte	Cambio de tareas.
Espera	Espera.
Movimiento	Movimiento.
Defectos	Defectos.

Fuente: Franco (2007)

2.2.11. Drupal

Drupal es un sistema de gestión de contenidos (Custom Management System) que se utiliza para crear sitios web dinámicos con gran variedad de funcionalidades. Es un software libre, escrito en PHP, que cuenta con una amplia y activa comunidad de usuarios y desarrolladores que colaboran conjuntamente en su mejora y ampliación. (Drupal, 2017)

Hoy en día casi cualquier proveedor de alojamiento (hosting) dispone de las características mínimas requeridas por Drupal para su instalación y correcto funcionamiento, aunque es recomendable consultar al proveedor previamente. (Gil, 2011)

Esta amplia variedad de funcionalidades es posible gracias a que se trata de un sistema modular con una arquitectura muy consistente, que permite que los módulos creados por cualquier desarrollador puedan interactuar con el núcleo del sistema y con los módulos creados por otros miembros de la comunidad. (Romero, 2014)

Con Drupal es posible implementar una gran variedad de sitios web: un blog personal o profesional, un portal corporativo, una tienda virtual, una red social, y otros.

2.2.12. Choko

Framework de Aplicaciones Web para Node.js. Con Choko se puede desarrollar complejas aplicaciones web individuales en cuestión de minutos, sin necesidad de ningún tipo de conocimiento técnico.

Choko viene con una acumulación en el Sistema de Gestión de Contenidos y Framework, por lo que puede gestionar su solicitud y el contenido relacionado con él en un lugar central, con APIs flexibles y potentes. (Choko, 2016)

2.2.13. Docker

Docker es un framework de administración de containers que permite crear ambientes de desarrollo comunes para los equipos de desarrollo e infraestructura, de manera que el deploy de las aplicaciones se pueda hacer con mayor facilidad y de forma automatizada.

Las herramientas que Docker permiten tener agilidad, control y portabilidad en la administración de ambientes, siendo posible simular ambientes con varios servicios (MySQL, PHP, Solr, NodeJS, MariaDB, MongoDB y otros) donde cada servicio es representado por un *container*.

Docker utiliza las funcionalidades de aislamiento de recursos, del Kernel de Linux, como *cgroups* y *namespaces* para que los containers sean ejecutados aisladamente en el Sistema Operativo (SO). Eso significa que cada container es ejecutado como un proceso por separado, y consume los recursos nativos del hardware. El proceso es virtualizado, pero el procesamiento no. (Taller Blog, 2016)

Imagen y Contenedor

Una imagen se refiere a una lista de capas, que son apiladas una encima de otra y forman la base del contenedor. Una imagen es inmutable, pero fácilmente extendida.

El container, por otro lado, es una instancia en el tiempo de ejecución de una imagen. Cuando un nuevo container es creado, una nueva capa de escritura es creada en la parte superior de las capas adyacentes. Todas las

alteraciones hechas en el container en ejecución son hechas en la misma capa de escritura, como se muestra en la Figura 8.

Una ventaja de los containers Docker es la portabilidad, a diferencia de otras herramientas de contenerización como LXC (Linux containers), pues permite utilizar la misma imagen en diferentes distribuciones Linux con configuraciones distintas de hardware; sin alterar las imágenes.

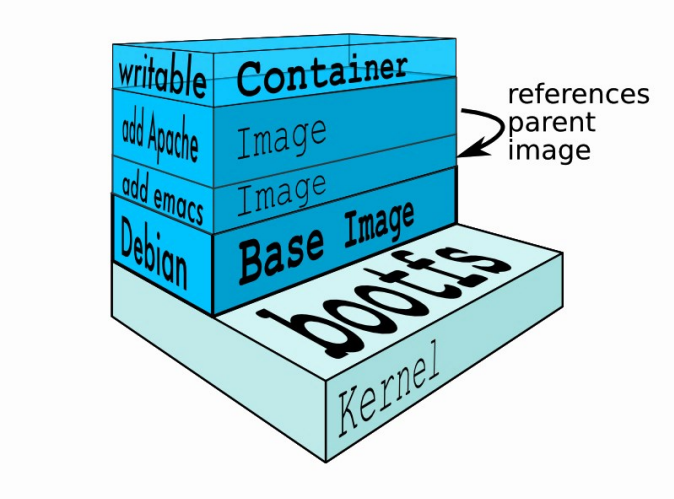

Figura 8: Containers de Docker

Docker Hub

Es un repositorio de imágenes Docker y debido a la popularidad de Docker esta plataforma contiene muchas imágenes oficiales de herramientas como: MongoDB, Redis, Nginx, WordPress, Solr, entre otros, listos para ser utilizados. (Taller Blog, 2016)

2.2.14. Trello

Trello es un gestor de tareas que permite el trabajo de forma colaborativa mediante tableros (board) compuestos de columnas (llamadas listas) que representan distintos estados. Se basa en el método Kanban para la

gestión de proyectos, con tarjetas que viajan por diferentes listas en función de su estado. Así, se puede tener una lista de cosas por hacer (o pendientes), que se están haciendo (o en proceso) o hechas (terminadas).

2.2.15. Git

Según Lopez (2015), Git es un Sistema de control de vesiones (System Control Versiones o SCV) distribuido y diseñado para la gestión eficiente de flujos de trabajo distribuido no lineales. Git fue diseñado y desarrollado inicialmente por Linus Torvalds en 2005 para el desarrollo del kernel de Linux. La licencia de Git es libre y hay distribuciones oficiales para los sistemas operativos Mac OS X, Windows, Linux y Solaris. La distribución de Git incluye herramientas de línea de comando y de escritorio. Además, hay disponibles herramientas proporcionadas por terceros que permiten una mayor integración con el escritorio o con entornos de desarrollo.

La experiencia de Linus Torvalds en la gestión de la integración de las diferentes aportaciones en un proyecto distribuido de la magnitud del kernel de Linux determinó las siguientes decisiones de implementación, que se detallan en la Figura 9:

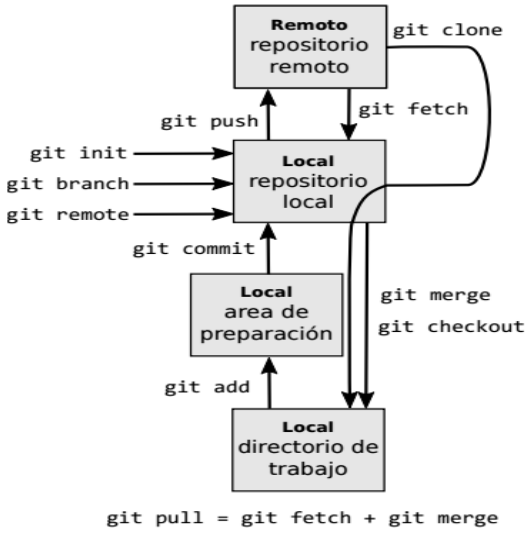

Figura 9: **Principales comandos de Git**

Versiones no incrementales

Git almacena cada cambio como una instantánea de todos los archivos del proyecto. Para ser eficiente, si el archivo no ha sido modificado, sólo se almacena un enlace al archivo idéntico previamente almacenado. Los SCV anteriores a Git habitualmente almacenaban solo una versión base y las modificaciones hechas en cada cambio por archivo.

Autenticación criptográfica de la historia

El identificador de un cambio se computa utilizando un algoritmo criptográfico que utiliza como entrada el cambio y la historia completa de cambios. Esto permite que cualquier cambio de la información durante la transmisión o en sistema de archivos sea detectado por Git.

Trabajo fuera de línea

Por ser un sistema distribuido, cada repositorio de Git es un repositorio completo capaz de funcionar sin acceso a la red o al resto de los

repositorios distribuidos gracias a que contiene una copia local de la historia completa del desarrollo del proyecto. Los cambios en la historia pueden copiarse de un repositorio a otro como nuevas ramas de desarrollo y se pueden copiar de la misma manera que una rama de desarrollo local.

2.2.16. Bitbucket

Bitbucket es un servicio de alojamiento basado en web que es propiedad de Atlassian, utilizado para código fuente y el desarrollo de proyectos que utilizan Mercurial (desde su lanzamiento) o Git (desde octubre de 2011) sistema de control de versiones. Bitbucket está escrito en Python utilizando el framework web de Django; ofrece planes comerciales y cuentas gratuitas. Ofrece cuentas gratuitas con un número ilimitado de repositorios privados (que pueden tener hasta cinco usuarios en el caso de cuentas gratuitas) a partir de septiembre de 2010; se integra con otros softwares de Atlassian como JIra, HipChat, Confluence y Bamboo.

Es similar a GitHub, que usa principalmente Git. Bitbucket tradicionalmente se ha diseñado para ayudar a los desarrolladores especialmente desde que fue adquirido por Atlassian en 2010; tiene 3 modelos de implementación: Cloud, Bitbucket Server y Data Center. (Bitbucket, 2016)

2.2.17. MongoDB

Es una base de datos no relacional (NoSQL) de código abierto que guarda los datos en documentos tipo JSON (JavaScript Object Notation) pero en forma binaria (BSON) para hacer la integración de una manera más rápida. Se pueden ejecutar operaciones en JavaScript en su consola en vez de

consultas SQL. Además, tiene una gran integración con Node.js con el driver propio y con Mongoose. Debido a su flexibilidad es muy escalable y ayuda al desarrollo ágil de proyectos web. (MongoDB, 2016)

2.2.18. Node.js

Node.js es un entorno de programación en la capa del servidor basado en el lenguaje de programación JavaScript, con I/O de datos en una arquitectura orientada a eventos y basado en el motor JavaScript V8. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como servidores web.

Al contrario que la mayoría del código JavaScript, no se ejecuta en un navegador, sino en el lado del servidor. Node.js implementa algunas especificaciones de CommonJS. (Llorens, 2014)

Es la tecnología que se ha utilizado para la creación de la página web expuesta en un apartado anterior y que se apoya en el motor de JavaScript V8 para hacer posible la ejecución de programas hechos en JavaScript en un ámbito independiente del navegador. La mayor ventaja que presenta Node.js es la de no provocar bloqueos. Es decir, si durante la ejecución de un programa hay partes que necesitan un cierto tiempo para generar la respuesta, Node.js no detiene la ejecución del programa esperando que esa parte acabe, sino que continúa procesando las siguientes instrucciones que necesitaban hacer uso de los resultados generados por dicho proceso. Por todo ello su funcionamiento es muy ágil. (Sangeap, 2015)

2.2.19. JavaScript

Es un lenguaje de programación interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y C.

Al igual que Java, JavaScript es un lenguaje orientado a objetos propiamente dicho, ya que dispone de Herencia, si bien esta se realiza siguiendo el paradigma de programación basada en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipo) y extendiendo su funcionalidad. Todos los navegadores modernos interpretan el código JavaScript integrado dentro de las páginas web.

2.2.20. AngularJS

AngularJS es un marco estructural para aplicaciones web dinámicas. Le permite utilizar HTML como lenguaje de plantillas y le permite extender la sintaxis del HTML para expresar los componentes de su aplicación clara y sucinta. El enlace de datos de AngularJS y la inyección de dependencia eliminan gran parte del código que actualmente tiene que escribir. Y todo sucede en el navegador, lo que lo convierte en un socio ideal con cualquier tecnología de servidor. AngularJS toma otro enfoque. Se trata de minimizar la falta de concordancia entre el documento HTML y cuáles son las necesidades de una aplicación mediante la creación de nuevos bloques HTML. AngularJS enseña la nueva sintaxis navegador a través de una construcción que llamamos directivas. (Izquierdo, 2015)

Los objetivos de diseño:

- Disociar la manipulación del DOM de la lógica de la aplicación. Esto mejora la capacidad de prueba del código.
- Considerar a las pruebas de la aplicación como iguales en importancia a la escritura de la aplicación. La dificultad de las pruebas se ve reducida drásticamente por la forma en que el código está estructurado.
- Disociar el lado del cliente de una aplicación del lado del servidor. Esto permite que el trabajo de desarrollo avance en paralelo, y permite la reutilización de ambos lados.
- Guiar a los desarrolladores a través de todo el camino de la construcción de una aplicación: desde el diseño de la interfaz de usuario, a través de la escritura de la lógica del negocio, hasta las pruebas.

2.2.21. Test Automatizados

De todas las prácticas de ingeniería que comprende LEAN la utilización de los test automatizados es la más importante. Los test automatizados son bloques de código, llamados códigos de test que ejecutan el código de producción y pueden ser organizados en constructores, se puede ejecutar millares de test en la aplicación en pocos minutos. Los test pueden ser categorizados como:

- **Test Unitarios:** Testean cada tramo de código del sistema.
- **Test Integrados:** Testean clases o módulos de forma integrada.
- **Test Funcionales:** Testean funciones específicas del sistema de punta a punta.

2.3. DEFINICIÓN DE TÉRMINOS BÁSICOS

2.3.1. Gestión de documentos

Área de gestión responsable de un control eficaz y sistemático de la creación, la recepción, el mantenimiento, el uso y la disposición de los documentos, incluidos los procesos para incorporar y mantener, en forma de documentos, la información y prueba de las actividades y operaciones de la organización. (Bustelo, 2011)

2.3.2. Documentación

Conjunto de documentos que describen operaciones, instrucciones, decisiones, normas, y procedimientos organizativos referidos a una determinada función, proceso o transacción. (Bustelo, 2011)

2.3.3. Sistema de gestión

Conjunto de elementos interrelacionados o que interactúan en una organización con el fin de establecer políticas y objetivos, y los procesos para alcanzarlos. (Bustelo, 2011)

2.3.4. CMS

Según Gil (2011) Content Management System o Sistema de Gestión de Contenidos, es un software que permite crear una estructura base para la creación y administración de contenidos, principalmente de páginas web.

2.3.5. Framework

La palabra “framework” (infraestructura, armazón, marco) define, en términos generales, un conjunto estandarizado de conceptos, prácticas y

criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

En el desarrollo de software, un framework, es una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación.

2.3.6. Arquitectura isomorfa

La idea general es crear sistemas que están hechos de diferentes tecnologías y subsistemas, pero para el usuario final parece consistente y uniforme, como si fuera un sistema único, en otras palabras, hacer que todas las aplicaciones se ejecuten en un navegador web, pero que todas las funciones utilicen la misma interfaz de usuario. (Mcginnis, 2017)

2.3.7. Pitch elevador

Técnica que consiste en resumir el objetivo del proyecto identificando: público objetivo, necesidad, nombre del producto, beneficios, competidores y diferencias competitivas. La técnica tiene este nombre porque debe ser posible presentar un contenido en un corto período, que corresponde al transporte de un ascensor (Casas, 2013).

2.3.8. User Story Mapping

Es una herramienta que permite generar una representación visual del sistema completo. Ofrece una vista general de todas las funcionalidades que lo componen de punta a punta. (Patton, 2016)

2.3.9. SemaphoreCI

Host de integración continua (continuous integration) y desarrollo de servicios para proyectos de código privado y abierto.

2.4. OPERACIONALIZACIÓN DE VARIABLES

Cuadro 2: Operacionalización de variables

Variable	Indicadores	Escala
Satisfacción de usuario	Navegación por el sistema Claridad de los mensajes Claridad de las páginas y listados Cantidad de información Adecuación de la presentación (responsivo) Frecuencia de actualización de las informaciones Tiempo de respuesta del sistema (velocidad de carga)	Deficiente Mala Regular Bueno Muy bueno
Utilidad del sistema	Mejoría en el flujo de trabajo Mejor utilización del tiempo Adquisición de nuevos conocimientos Autonomía en relación a otros Reducción en el tiempo de ejecución de las tareas Disponibilidad del sistema (siempre on-line) Mejoría en el ambiente de trabajo	Deficiente Mala Regular Bueno Muy bueno

Fuente: Los ejecutores del proyecto

CAPÍTULO III

MATERIALES Y MÉTODOS

3.1. POBLACIÓN

La población de estudio estuvo conformada por el personal administrativo y técnico que labora en la Oficina de Cooperación Nacional e Internacional y los usuarios que fueron atendidos en las unidades durante el II semestre académico 2017.

3.2. DISEÑO DE MUESTRA

El diseño de muestra que se utilizó es el muestreo por conveniencia, el cual es una técnica de muestreo no probabilístico donde los sujetos son seleccionados dada la conveniente accesibilidad y proximidad de los sujetos para el investigador.

Para el análisis de utilidad del sistema el tamaño de muestra fue de 4 personas, personal administrativo y técnico que labora en la OCNI.

Por otra parte, para el análisis de la satisfacción de los usuarios, se tomó una muestra de 60 usuarios. Cabe resaltar que el tamaño de muestra fijado, fue en proporción al número de usuarios atendidos en las unidades de la OCNI en el I semestre académico 2017.

Criterios de selección

Se encuestó a aquellos usuarios que fueron atendidos en las unidades de la OCNI considerando los criterios de inclusión, exclusión y eliminación. Estos criterios sólo se aplicaron a los usuarios y no al personal que labora en la OCNI.

Criterios de Inclusión

- Unidad de Convenios Nacionales e Internacionales.
  - Usuarios que solicitaron información de Convenios.
  - Usuarios que desearon participar.
- Unidad de Movilidad de Estudiantes, Docentes, Investigadores y Personal Administrativo.
  - Usuarios que postularon a programas de movilidad.
  - Usuarios que desearon participar.
- Unidad de Becas Nacionales e Internacionales.
  - Usuarios que solicitaron información de becas.
  - Usuarios que desearon participar.

Criterios de Exclusión

Usuario que no deseó participar en el estudio

Criterios de Eliminación

Usuario que respondió en forma incompleta a las preguntas de las encuestas.

3.3. MÉTODO DE RECOLECCIÓN DE LA INFORMACIÓN

El método de recolección de información fue el método de la encuesta haciendo uso de la técnica de la entrevista personal a través de dos fichas de evaluación. Encuesta de utilidad del sistema: Dirigida al personal de la OCNI.

Encuesta de satisfacción de usuario: Dirigida a los usuarios de las unidades de la OCNI.

3.4. MÉTODO DE ANÁLISIS DE DATOS

El método de análisis que se utilizó se basó en el cuestionario propuesto por Andrade et al. (2001), para la evaluación de eficacia de un sistema de información a través del nivel de satisfacción del usuario y el nivel de la utilidad de un sistema; fue necesario realizar adaptaciones, excluyendo algunos ítems y modificando otros debido a la traducción del portugués, con el fin de enfocarnos en evaluar la satisfacción del usuario y la utilidad del sistema de forma individual.

Los cuestionarios adaptados tienen 7 preguntas cada uno. La escala de medición cada pregunta se detalla en el Cuadro 3.

Cuadro 3: Escala de medición

Escala de medición	Puntuación
Deficiente	1
Mala	2
Regular	3
Bueno	4
Muy bueno	5

Fuente: Los ejecutores, adaptación de Franco 2001

La escala de medición categorizada de las variables Satisfacción del usuario y Utilidad del sistema están en los cuadros 4 y 5 respectivamente.

Cuadro 4: Escala de medición categorizada de la variable Satisfacción de Usuario

Nivel de satisfacción	Puntuación
Muy bajo	7 – 12
Bajo	13 – 18
Medio	19- 23
Alto	24 – 29
Muy alto	30 – 35

Fuente: Los ejecutores, adaptación de Franco 2001

Cuadro 5: Escala de medición categorizada de la variable Utilidad del Sistema

Nivel de utilidad	Puntuación
Muy baja	7 - 12
Baja	13 - 18
Media	19- 23
Alta	24- 29
Muy alta	30 – 35

Fuente: Los ejecutores, adaptación de Franco 2001

3.5. METODOLOGÍA DE DESARROLLO DE SOFTWARE

El modelo que se consideró para el desarrollo del sistema fue basado en la combinación del framework de administración de proyectos ágiles SCRUM, la metodología ágil XP de desarrollo de software y los principios de la filosofía Lean Software Development propuesto por Franco (2007), el cual se describe a continuación.

La Figura 10 presenta un mapeo de la utilización de Scrum para realizar la administración de proyectos (región amarilla), XP para realizar el desarrollo de

software (región azul), y el pensamiento esbelto proporcionando principios y valores que influencian a ambas metodologías (región verde).

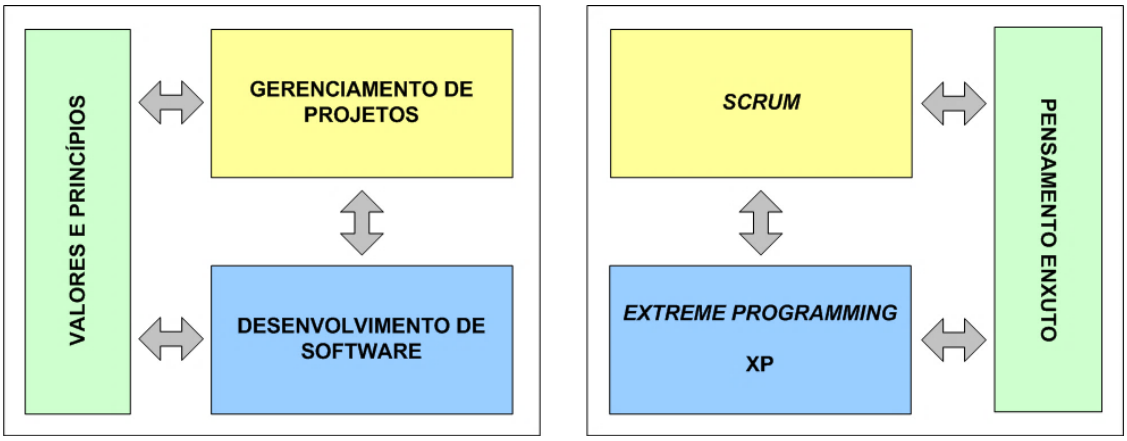

Figura 10: Mapeamiento de las metodologías seleccionadas, sus interacciones y las áreas de aplicación Franco (2007)

La interacción entre las dos metodologías (Scrum y XP) ocurre en los trabajos realizados diariamente dentro de los sprints. A través de la reunión de planeamiento del sprint (Scrum), son definidos los requisitos que serán trabajados en la próxima iteración (backlog del sprint). Iniciado el sprint, cada ítem del backlog será trabajado diariamente por las prácticas y herramientas seleccionadas de XP, para transformar los requisitos seleccionados en incrementos del producto.

La Figura 11, ilustra la interacción descrita, donde las prácticas y herramientas de XP son aplicadas en los trabajos diarios (círculo azul) y las definidas por Scrum, son utilizadas para realizar la administración del desarrollo de software (círculo anaranjado).

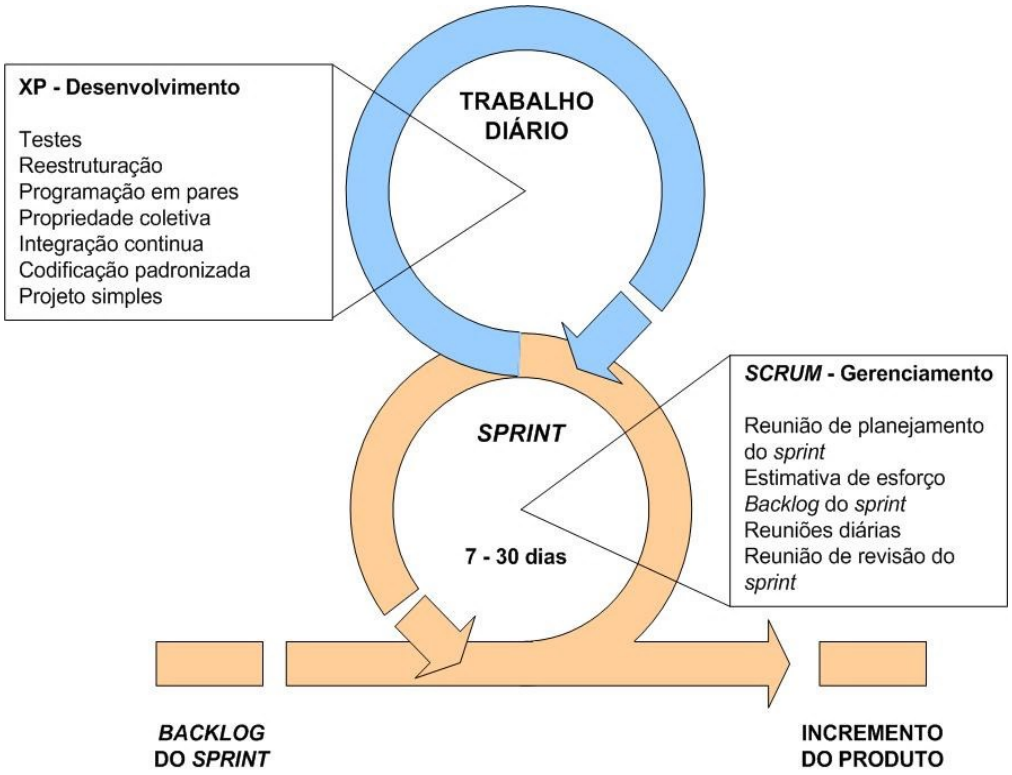

Figura 11: Iteración entre XP y Scrum en el modelo propuesto Franco (2007)

3.5.1. Roles y Responsabilidades

En el modelo propuesto, los roles utilizados son heredados de Scrum, con alteraciones de nomenclatura para generalizar la propuesta. Esta opción fue hecha para preservar la nomenclatura de la metodología que compone la mayor parte del modelo propuesto.

A partir de la definición inicial de Scrum, fueron hechas adaptaciones para incorporar las responsabilidades de desarrollo de software definidas por XP.

- **Entrenador:** Este rol heredó la nomenclatura de XP y las responsabilidades de Scrum. Él es el responsable para garantizar que el proyecto esté siendo conducido de acuerdo con las prácticas,

herramientas, valores y reglas del modelo propuesto y que el progreso del proyecto esté de acuerdo con el deseo de los clientes. El entrenador interactúa con el equipo, como los clientes y el gerente. Él también es responsable por eliminar y alterar cualquier obstáculo identificado, garantizando que el equipo trabaje de la forma más productiva posible.

- **Líder del proyecto:** Éste rol heredó las responsabilidades del rol responsable por el producto, definido por Scrum. En el modelo propuesto, sufrió a penas la alteración de la nomenclatura, para generalizar su significado para proyectos bajo demanda y desarrollo de productos.
- **Equipo:** En el modelo propuesto, este rol asume las responsabilidades de Scrum, o sea, él tiene autoridad para decidir sobre las acciones necesarias y de organizar para poder alcanzar los objetivos de cada sprint. Está involucrada en la estimación del esfuerzo, en la creación y en la revisión de los ítems backlog y por sugerir los obstáculos que deben ser eliminados. Éste rol también heredó las responsabilidades de los roles Testador y Programador de XP, dónde ayudan al cliente a escribir las pruebas funcionales, por el mantenimiento de los mismos y por ejecutar periódicamente las pruebas. El equipo también es responsable por proyectar, codificar y mantener el programa lo más sencillo y breve posible.
- **Cliente:** Rol y responsabilidades heredadas de Scrum. El modelo propuesto en el sufrió alteración.
- **Gerente:** Rol y responsabilidades heredadas de Scrum, El modelo propuesto en el sufrió cambio.

3.5.2. Proceso

La Figura 12, ilustra el modelo propuesto, presentando las prácticas y herramientas seleccionadas de Scrum y XP, identificando sus entradas y salidas, conjuntamente con sus interrelaciones y las fases del proceso de desarrollo que son aplicadas (Concebir, explorar y adaptar y cerrar).

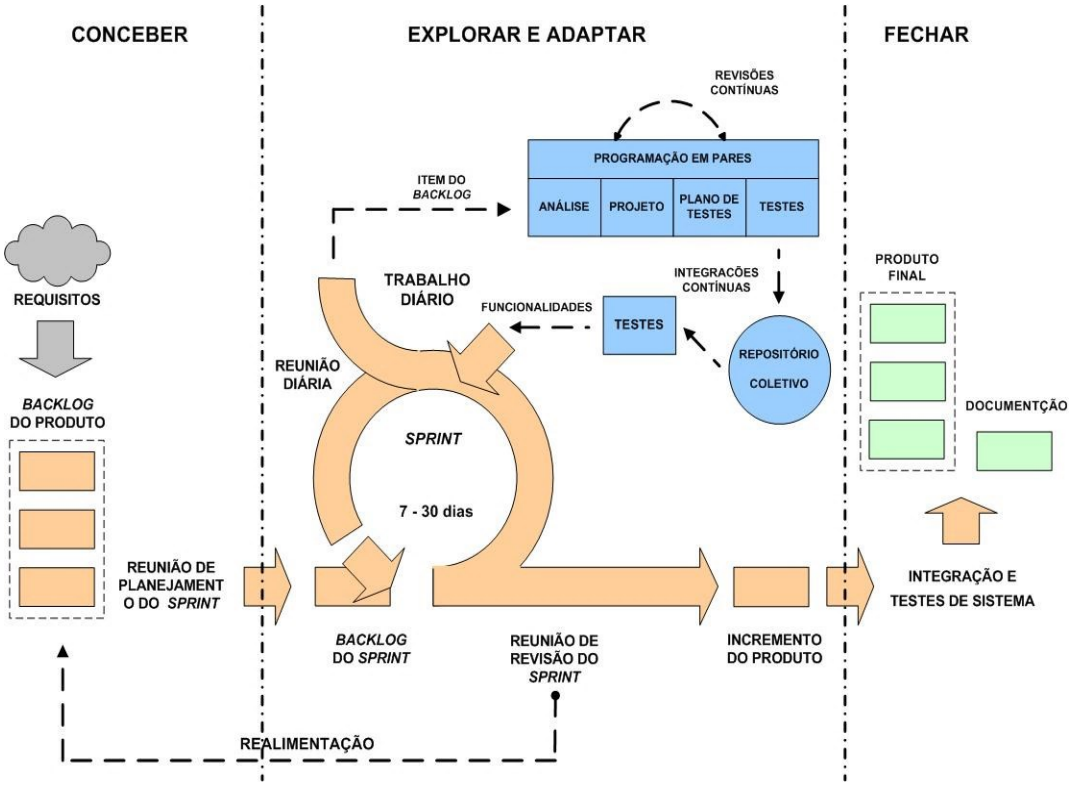

Figura 12: Representación gráfica del modelo propuesto por Franco (2007)

El modelo propuesto toma como base el proceso de administración de proyectos especificado por Scrum, representado por la región en anaranjado en la Figura 12, alterando la nomenclatura de las fases para quitar posibles ambigüedades de traducción de inglés para portugués.

La nomenclatura utilizada fue inspirada en el trabajo publicado por Highsmith (2004). El cuadro 6 presenta en la columna de la izquierda el

nombre original definido en Scrum, en cuanto que la columna de la derecha, presenta el nombre de la fase correspondiente en el modelo propuesto.

Cuadro 6: Mapeo de la nomenclatura de las fases

Scrum	Modelo Propuesto
Antes del juego	Concebir
Desarrollo	Explorar y adaptar
Después del juego	Cerrar

Fuente: Franco (2007)

El período sugerido para cada iteración (sprint) es de 7 a 30 días, manteniendo el período sugerido por Scrum. Esta duración debe ser definida, analizando la necesidad de cada cliente y proyecto.

Cada iteración está cubierta por las fases de proyectos definidos, detallando los requisitos del siguiente sprint, y explorar y adaptar, donde los ítems de backlog del sprint son transformados en incrementos de producto y el aprendizaje adquirido es retroalimentado al proceso después de la realización de las reuniones de revisión del sprint. Seguidamente, son descritas cada una de las fases que componen el modelo propuesto, proporcionando una visión breve de trabajo realizado en cada una de ellas, conjuntamente con los artefactos producidos y apuntando qué principios de desarrollo esbelto de software son aplicables.

Concebir

Esta fase va más allá de la implantación del proyecto. Iniciando la connotación de elaboración de presupuestos y cronogramas detallados.

Estos artefactos, a pesar de ser necesarios, necesitan fluir de una visión bien articulada, discutida y acordada por todos los involucrados. La definición de visión puede ser expresada a través de respuestas a cuatro preguntas:

- ¿Cuál es la visión del producto?
- ¿Cuál es el propósito del proyecto y sus restricciones?
- ¿Quién será incluido en la comunidad del proyecto?
- ¿Cómo se organizará el equipo para entregar el producto?

Para algunos proyectos, esta fase puede ser realizada en pocos días, para otros, particularmente aquellos que pasaron por algún tipo de estudio de viabilidad, puede llevar más tiempo. Este aumento de tiempo necesariamente representa un aumento de esfuerzo, este tiempo puede ser necesario para difundir la visión del producto entre la comunidad del proyecto.

En esta fase también está elaborada la lista de **backlog del producto**, transformando la visión creada en requisitos. Después de elaborar esta lista, es creado un proyecto de alto nivel, contemplando una **arquitectura preliminar** de software, en una reunión y conducida para elaborar planes preliminares de los contenidos del producto de cada sprint. El diseño del producto el que se hace una vez en el inicio del proyecto. Antes del inicio de cada sprint, cuando es reiniciado el ciclo *Concebir*, *Explorar y Adaptar*, el equipo se reúne con los clientes para planear la siguiente iteración. En esta reunión la visión y los requisitos del producto son revisados y una visión particular para el siguiente sprint es creada (que corresponde a un

subconjunto de visión global del producto). Esta revisión sirve para modificar la visión o para reiterar al equipo del propósito del empeño de cada uno. Además, ésta revisión permite adicionar, alterar y quitar ítems que componen la lista de backlog del producto.

Durante la reunión de planeamiento del sprint es definido el ámbito a ser trabajado en el siguiente Sprint, seleccionando qué ítems del backlog del producto serán incorporados a la lista de backlog del sprint, que permanecerá estable a lo largo de todo el sprint.

Antes de iniciar el desarrollo de las funcionalidades, debe ser creado un plan iterativo basado en entregas de funcionalidades. Este tipo de planeamiento fuerza al equipo y a los clientes a comprender el producto y enfocarse en la generación de valor y en el flujo de entregas de funcionalidades, en lugar de controlar el cumplimiento de los planes iniciales.

Los principios de desarrollo esbelto de software aplicable en esta fase son:

Visualizar todo: Antes del inicio del desarrollo del producto es concebida la visión que guiará los trabajos futuros (por ejemplo, a través de un documento de visión, declaración de elevador, diagrama de flujos, etc.), la concepción del producto es hecha de la definición del equipo de proyecto, de las herramientas y otros recursos, la evaluación de riesgos y necesidades del entrenamiento.

Construir integridad: la elaboración y priorización de la lista de backlog del producto y del sprint es centrado en el valor generado para los clientes,

una vez que ellos también participan en la creación de estos artefactos. Esta participación garantiza que los incrementos producidos tendrán utilidad y atenderán las necesidades obteniendo la integridad conceptual del producto.

Tomar decisiones lo más tarde posible: las decisiones son postergadas a través de la segmentación de la lista de backlog del producto. Las decisiones referentes al campo de trabajo a ser realizado en la siguiente iteración son hechas iterativamente en las reuniones de planeamiento del sprint.

El Cuadro 7 presenta los artefactos producidos durante la fase Concebir, conjuntamente con una descripción breve.

Cuadro 7: Artefactos producidos en la fase Concebir

Artefactos producidos	Descripción
Visión del producto	Elaboración en el inicio del proyecto y revisado durante el planeamiento de cada sprint.
Backlog del producto	Elaborado en el inicio del proyecto y revisado durante el planeamiento de cada sprint, pudiendo adicionar, alterar o remover ítems de la lista.
Arquitectura	Una arquitectura preliminar del producto es definida en el inicio del proyecto, y es constantemente adaptada y mejorada para atender las necesidades emergentes.
Backlog del sprint	Producido durante la reunión de planeamiento del sprint y es estable durante toda la iteración.
Visión del sprint	A partir de la visión del producto y del backlog del sprint, es definido una visión específica para el siguiente sprint.

Fuente: Franco (2007)

Explorar y Adaptar

La exploración es la forma que el Equipo ejecuta los proyectos. En lugar de seguir planes prescritos, lo ejecuta o proyecta a través de una serie de experimentos planificados, entregas iterativas e incrementales de funcionalidad, buscando crear una formulación concreta de la visión del producto (Thomke, 2003).

La exploración es realizada por equipos competentes, auto-disciplinados y dirigidos por gerentes con experiencia que crean un ambiente auto-organizado. Los equipos trabajan de forma semi-autónoma, empeñadas en atender los planes iterativos que ellas misma ayudaron a construir, administrando su propia carga de trabajo, colaborando para crear nuevas ideas y utilizando conocimientos técnicos específicos.

Monitorear y adaptar, actividades tradicionalmente llamadas de monitorear y controlar, son parte de cualquier buen enfoque de administración de proyectos. A pesar de que el enfoque ágil utiliza algunas prácticas de administración de proyectos tradicionales, ella utiliza algunas prácticas diferenciadas. Por ejemplo, en lugar de identificar desvíos con relación al planeamiento inicial y elaborar informes de excepción (a pesar de que sean necesarias en algunas situaciones), el enfoque ágil busca explorar los escenarios emergentes.

Iteraciones frecuentes, que entregan funcionalidades incrementalmente, permiten que los equipos de proyectos realicen ajustes basados en resultados verificables, en lugar de artefactos de documentación. Este

escenario puede generar situaciones de malestar para algunos gerentes y también clientes, que no desean tratar constantemente con situaciones de compromiso.

El desarrollo del producto es hecho a través de las prácticas y herramientas seleccionadas de XP, que son utilizadas en los trabajos diarios. Diferente del modelo cascada, las etapas del proyecto, análisis, codificación y pruebas no son caracterizadas formalmente y no exigen un cierre formal. Las funcionalidades producidas diariamente son agrupadas, y al final del sprint, son incorporadas al incremento del producto que será presentado a los Clientes.

Al final de cada sprint, los resultados obtenidos son evaluados en la reunión de revisión del sprint de acuerdo con las diversas perspectivas involucradas en la creación del producto: técnica, percepción del Cliente, desempeño del equipo y del proceso. Los resultados de estas revisiones son utilizados para alimentar la reunión de planeamiento del siguiente sprint, realizado en la fase Concebir.

Con las informaciones levantadas en las reuniones de revisión del sprint, que ocurre al fin de cada iteración, los Clientes y el equipo del proyecto pueden mejorar el planeamiento y el producto, incorporando el aprendizaje obtenido a lo largo de la ejecución del proyecto.

En el Cuadro 8 se presenta los artefactos producidos durante la fase Explorar y Adaptar, conjuntamente con una descripción breve.

Cuadro 8: Artefactos producidos en la fase Explorar y Adaptar

Artefactos producidos	Descripción
Funcionalidad	Para cada ítem del backlog del sprint son realizadas las actividades de análisis, diseño, codificación y pruebas para que sean transformados en funcionalidades.
Incremento del producto	Cuando todos los ítems del backlog del sprint sean transformados en funcionalidades, un incremento del producto funcional para ser entregado a los Clientes.

Fuente: Franco (2007)

Los principios del desarrollo esbelto de Software aplicables en esta fase son:

- **Amplificar el aprendizaje:** el conocimiento es adquirido incrementalmente, conforme la elaboración y la entrega de funcionalidades del producto a los clientes. Al final de cada iteración es realizada una reunión de revisión del sprint, donde el equipo discute sobre el trabajo realizado, los problemas identificados y las alteraciones que deben ser hechos para mejorar el proceso y el producto en construcción.
- **Hacer entregas lo más rápido posible:** el trabajo es dividido en iteraciones, reduciendo el intervalo entre el levantamiento de los requisitos y el contacto de los Clientes con partes funcionales del producto. El ajuste de este intervalo es hecho a través de la duración de los sprints, sugerido entre 7 y 30 días.
- **Construir integridad:** La integridad estructural del producto es obtenida a través de la experiencia de los involucrados en el proyecto, de la

utilización de la práctica de diseños simples (traída de XP), que incrementalmente son detallados y ampliados para emerger la arquitectura del producto.

Cierre

Similar a la fase Pós-Game de Scrum, esta fase se inicia cuando es acordado que el producto está listo para su utilización y no existen más ítems en la lista del backlog del producto a ser implementados. Es en este momento que son realizadas las actividades de integración, pruebas de sistema y documentación.

El objetivo principal de esta fase y de los cierres parciales realizados al final de cada sprint es generar e incorporar conocimiento para el trabajo de la siguiente iteración o transferirlo para el equipo del siguiente proyecto.

El Cuadro 9 presenta los artefactos producidos durante la fase Cierre, conjuntamente con una descripción breve.

Cuadro 9: Artefactos producidos en la fase Cierre (cerrar)

Artefactos producidos	Descripción
Producto final	Contiene todos los ítems de la lista de backlog del producto y representa la visión definida en la fase Concebir.
Documentación	Los documentos que son producidos en esta fase son definidos por los Clientes, ellos definen las necesidades y que tipo de documentación agrega valor al proyecto.

Fuente: Franco (2007)

3.5.3. Prácticas

Fueron seleccionadas las prácticas y herramientas de Scrum, relacionadas a la administración de proyectos y las prácticas y herramientas de XP que están enfocadas al desarrollo del software y a la garantía de la calidad del producto final, para componer el conjunto de prácticas y herramientas del modelo propuesto. A partir de la clasificación presentada en la Figura 2 en el marco teórico, donde las prácticas de los XP fueron agrupadas en cuatro círculos: Codificación, Equipo, Proceso y Producto, las prácticas seleccionadas para componer el modelo propuesto, fueron clasificadas y el resultado es presentado en el Cuadro 10.

Cuadro 10: Relación de las prácticas seleccionadas como los grupos de clasificación

Grupo.	Metodología	Prácticas
Codificación	XP	Pruebas unitarias Reestructuración Programación en pares
Equipo	XP	Integración continua Propiedad colectiva Padrón de codificación
Proceso	XP	Diseño simple
	Scrum	Reunión diaria Reunión de planeamiento del Sprint Reunión de revisión de Sprint
Producto	Scrum	Backlog del product Backlog del Sprint

Fuente: Los ejecutores

La Figura 13 presenta en detalle las prácticas seleccionadas de la fase de iteraciones para el lanzamiento de XP y la interrelación entre ellas. El enlace entre las prácticas de XP y de Scrum ocurre en el ámbito del trabajo diario realizado en cada sprint, o sea, en el día a día el equipo utiliza las prácticas de XP para transformar los ítems del backlog en incrementos del producto. La codificación en pares es realizada en paralelo con las actividades de análisis, diseño, plano de pruebas y codificación de las pruebas, existiendo una transición formal entre ellas.

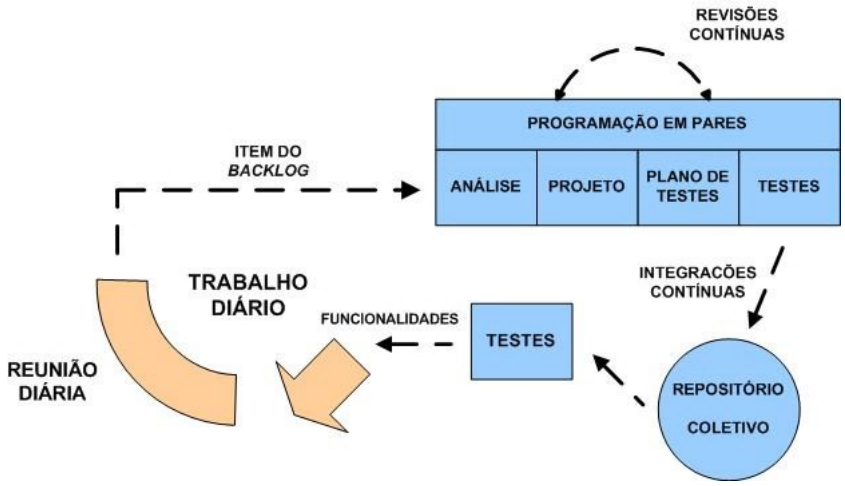

Figura 13: Detalle de las practicas seleccionadas de XP. (Franco, 2007)

El primer paso para implementar el enfoque esbelto, es aprender a identificar las pérdidas envueltas en el proceso. Éstas pérdidas, descritas anteriormente en el Cuadro del marco teórico, Desperdicios en el Desarrollo de Software, son clasificadas como cualquier actividad que no agrega valor directamente.

El Cuadro 11 presenta una descripción de cómo las prácticas y herramientas seleccionadas para componer el modelo propuesto,

minimizan la ocurrencia de las pérdidas en el desarrollo del software identificadas en el Cuadro 1 del marco teórico.

Cuadro 11: Descripciones de cómo las pérdidas del desarrollo del software son tratadas en el modelo propuesto

Pérdidas	Descripción de las actividades
Trabajo parcialmente finalizado	Los requisitos son detallados apenas para la iteración actual. Al final de cada iteración son entregados incrementos funcionales del producto.
Proceso extra	Codificar directo de las especificaciones contenidas en los ítems del backlog, aclarando dudas eventuales, verbalmente, directo en los clientes.
Funcionalidades extras	Son implementados apenas los requisitos de cada sprint.
Cambio de tareas	Todos trabajan en un mismo espacio físico y cada uno trabaja en una funcionalidad específica del producto (ítem de backlog del producto).
Espera	Las entregas son divididas en pequeños incrementos realizados a través de las iteraciones. Por ejemplo, la codificación en el que necesita esperar la especificación completa y detallada de los requisitos para iniciar.
Movimiento	Todos los involucrados en el proyecto, trabajan en un mismo espacio físico. Esto facilita el acceso a la información y reduce el tiempo necesario para aclarar dudas e incertidumbres.
Defectos	El desarrollo es dirigido a pruebas, antes de iniciar la codificación de las funcionalidades es implementando la prueba unitaria. Son realizados continuamente pruebas para garantizar que los incrementos realizados en el repositorio son consistentes y correctos, éstas pruebas son denominadas pruebas de integración.

Fuente: Los ejecutores

El Cuadro 12 presenta resumidamente los elementos que componen el modelo propuesto, indicando las fases, las actividades, las funciones de responsabilidad y los artefactos producidos.

Cuadro 12: Descripciones de cómo las pérdidas del desarrollo del software son tratadas en el modelo propuesto

Fase	Actividad	Responsable	Artefacto producido
Concebir	Elaborar la visión compartida del producto del proyecto.	Cliente, Líder de proyecto y Equipo	Visión del producto
	Elaborar y revisar el backlog del producto.	Cliente, Líder de proyecto y Equipo	Backlog del producto
	Diseño de alto nivel de la arquitectura.	Líder de proyecto y equipo	Arquitectura
	Reunión de planeamiento del sprint.	Equipo, Cliente y Líder de proyecto	Backlog del sprint
	Reunión de planeamiento del sprint.	Equipo, Cliente y Líder de proyecto	Visión del sprint
Explorar y Adaptar	Análisis, diseño, codificación en pares y pruebas.	Equipo y Líder de proyecto	Funcionalidades
	Integrar continuamente.	Equipo y Líder de proyecto	Incremento del producto
	Reunión de revisión del sprint.	Equipo, Líder de proyecto y Cliente	Incremento del producto
Cierre	Integrar sistema.	Líder de proyecto y Equipo	Producto final
	Pruebas de sistema.	Líder de proyecto y Equipo	Producto final
	Identificar los documentos necesarios.	Cliente, Líder de proyecto y Equipo	Documentación
	Elaborar los documentos necesarios	Cliente, Líder de proyecto y Equipo	Documentación

Fuente: Franco, 2007

CAPÍTULO IV

RESULTADOS Y DISCUSIÓN

4.1. EL EQUIPO DE PROYECTO

El equipo que participó en este proyecto fue integrado por cinco personas el cual se detalla en el Cuadro 13.

Cuadro 13: Descripción de los roles y responsables del proyecto

Roles	Responsables
Gerente	Jefe de la oficina.
Líder del proyecto	Encargado de la unidad de los programas de movilidad.
Entrenador	Uno de los desarrolladores de la investigación.
Equipo	Desarrolladores de la investigación.
Cliente	Personal de la oficina.

Fuente: Los ejecutores (basado en el modelo presentado por Franco, 2007).

En un principio el trabajo fue realizado en el local de la oficina, pero debido a las restricciones de la línea de internet dentro del edificio administrativo de la universidad, el Equipo tuvo que optar por trabajar de manera remota utilizando herramientas de videoconferencia (Hangouts, Appear, Skype), para no afectar el desarrollo del proyecto. Asimismo, se tiene que resaltar que las reuniones con el Gerente y el Líder del proyecto tuvieron lugar en las instalaciones de la OCNI.

Esta consideración fue importante para definir el contexto donde se realizó el proyecto y reducir el ámbito de las discusiones sobre complejidades referentes a la comunicación de equipos mayores.

4.2. EJECUCIÓN DEL PROCESO

A pesar de contar con el apoyo del personal de la oficina en la realización de la investigación, en un inicio hubo resistencia por parte del responsable de la oficina porque significó un cambio drástico en relación a las metodologías clásicas, costumbres arraigadas al proceso de desarrollo de software, la dinámica del trabajo, plazos de entrega y costo de la infraestructura en el ámbito local; de tal manera que se necesitó un cambio de mentalidad.

El proyecto tuvo que ser concluido en un año. El modelo propuesto, para preservar las características del Scrum, sugería una duración entre 7 a 30 días, y con el fin de hacer el proceso adaptable a las incertezas y responder a los requisitos emergentes, los sprints fueron definidos en periodos de dos semanas con una carga horaria de 4 horas al día (10 días hábiles). Con la duración definida de cada iteración, el proyecto fue estimado para realizar en 24 sprints. En este periodo fueron realizados **24 reuniones** de sprint (en el inicio de cada nueva iteración), donde podían ser removidos y/o incorporados nuevos ítems al backlog del producto.

4.2.1. Concebir

El producto se inició con una reunión, incluyendo al Equipo de *Scrum*, el Cliente (personal de la oficina) y el líder del proyecto (encargado de la unidad de movilidad). En esta reunión, se creó una visión definiendo lo que

llegaría a ser el producto deseado, a través de la declaración de la *prueba del ascensor*.

Orientado para toda la comunidad universitaria que necesita información referente a becas, programas de movilidad, convenios, convocatorias, testimonios de becarios, entre otros. La implantación del sistema de gestión es una solución compuesta por un conjunto de aplicaciones que permiten automatizar los procesos y actividades de gestión documental. Al contrario de las soluciones actuales que son burocráticas y mecánicas, nuestra opción presenta una mayor flexibilidad, facilidad y alcance en la publicación, acceso y difusión.

Aún en la reunión inicial, conjuntamente con la declaración de la prueba de ascensor, fue también redactado el artefacto denominado *Backlog* del producto. El equipo, junto con el líder de proyecto, tradujeron los requisitos y las restricciones proporcionados por los clientes sobre plazos y riesgos asociados a cada ítem del *backlog*. El artefacto en su etapa inicial fue producido utilizando plantillas de papel donde los responsables de cada unidad describieron los problemas que tenían. En la siguiente se detalla los ítems del backlog del producto.

- **O-001:** *Yo como responsable del área de convenios quiero tener la administración de la página web de la OCNI, para publicar continuamente información referida a los convenios suscritos por la UNA Puno*.
- **O-002:** *Yo como responsable del área de movilidad necesito que se implemente un programa para que los postulantes a los programas de*

movilidad estudiantil: CRISCOS, CRISUR y ALIANZA DEL PACÍFICO; puedan acceder a la información, documentos y/o formularios de las convocatorias, necesarios para participar. A fin de facilitar la inscripción de los interesados.

- **O-003:** *Yo como Responsable de convenios quiero organizar y actualizar el registro de convenios para tener actualizado los convenios suscritos a nivel local, nacional e internacional.*
- ***O-004:*** *Yo como responsable de convenios quiero poder generar informes detallados de los convenios solicitados por las Facultades y entidades públicas o privadas, para su publicación en la página web para poder informar oportunamente sobre los convenios suscritos por la UNA-Puno.*
- **O-005:** *Yo como responsable del área de movilidad necesito tener acceso a la página web de la Oficina de Cooperación Nacional e Internacional, para subir información de Becas, de movilidad estudiantil, docentes, investigadores. Asimismo, información de interés de la oficina para el público usuario. A fin de que la información tenga una mayor difusión y socialización al público objetivo, la comunidad Universitaria y en General.*

Finalizada la reunión, con los ítems del backlog codificados, estos fueron analizados por el Equipo *Scrum*, y se llegó a la conclusión de que el sistema de gestión documental sería compuesto por dos aplicaciones, la primera aplicación se enfocaría en los ítems **O-003** y **O-004;** y la segunda en los ítems **O-001, O-002** y **O-005**.

La siguiente tarea fue definir la arquitectura y las tecnologías. Las tecnologías seleccionadas para implementar el proyecto fueron PHP y Node.js a nivel de servidor y JavaScript para las interfaces de usuario, como se muestra en el Figura 14.

Para la *aplicación de convenios* se optó por Choko, por su gran flexibilidad a nivel de interfaz de usuario y Solr como servidor de búsqueda. Por otro lado, para el *portal institucional* se optó por elegir Drupal 8 con React para la interfaz**.**

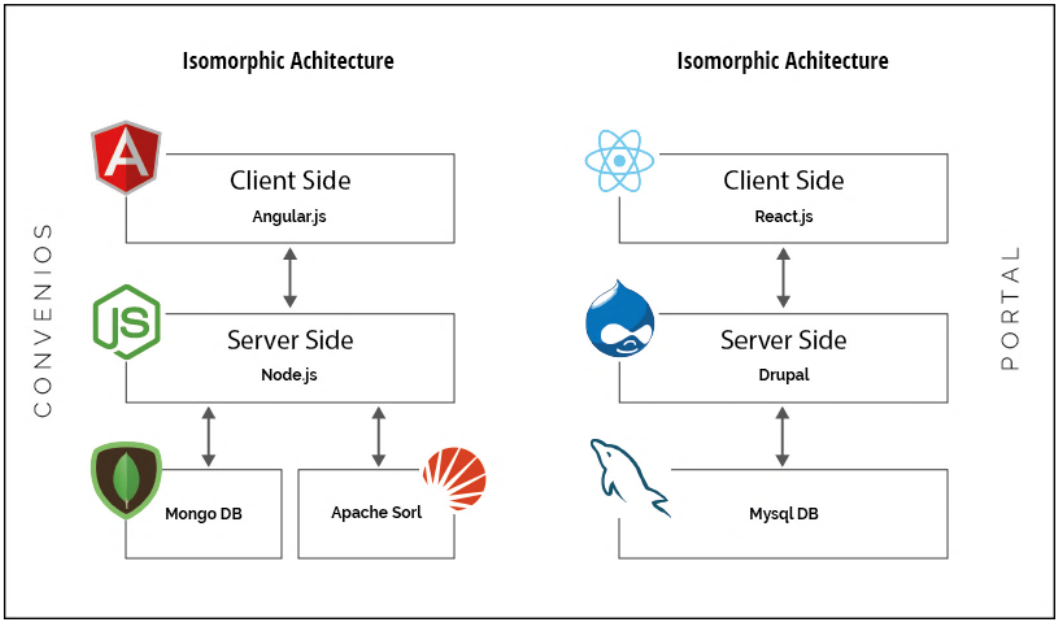

Figura 14: Arquitectura y tecnologías del sistema

En esta fase también fueron seleccionadas las herramientas que fueron utilizadas a lo largo de todo el desarrollo del proyecto para automatizar las tareas mecánicas del desarrollo. Para el aprovisionamiento de ambientes de desarrollo local, stage (pruebas) y de producción, se seleccionó Docker debido a su eficacia en el consumo de recursos y practicidad.

Para el control de versiones de código se eligió *Git* y como sistema de administración de los repositorios remotos se seleccionó Bitbucket. También fue definido el modelo de ramificaciones y gestión de lanzamientos basado en Vincent Driessen, el cual se muestra en la Figura 15. Donde toda *feature branch* (rama de funcionalidad) debe partir de la rama *master,* una vez finalizada se fusiona con la rama *stage* para ser validada; y finalizada su validación será fusionada con *master*; toda fusión fue realizada utilizando con el comando *rebase,* con el fin de mantener el histórico del proyecto linear.

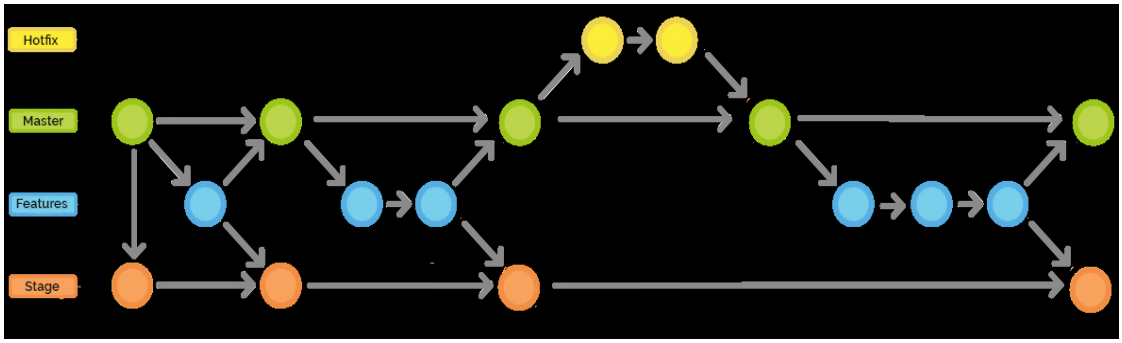

Figura 15: Modelo de ramificaciones de la propuesta del flujo de trabajo de Git

También se optó por utilizar SemaphoreCI como herramientas la integración continua y deploy.

Finalizada la definición de la arquitectura y tecnologías a ser utilizadas, se continuó con la organización de las columnas del tablero Kanban, los acuerdos del flujo de los ítems en el tablero, los roles y también responsabilidades de los miembros del equipo sobre cada columna del tablero; todo ello para el acompañamiento y consolidación de las actividades de los sprints backlog, durante el desarrollo de todo el proyecto.

Las columnas del tablero, organización de tareas y responsabilidades del equipo se detallan en los Cuadros 14 y 15 respectivamente.

Cuadro 14: Descripción de las columnas y la organización de las tareas del tablero Kanban

Columna	Tareas
Product backlog	Todas las tareas del backlog.
Sprint backlog	Tareas que conforman el *sprint*
Ready to development	Tareas listas para ser desarrolladas por el Equipo.
Development	Tareas que están siendo desarrolladas.
Ready for acceptance	Tareas que están listas para ser validadas.
Acceptance	Tareas que están siendo validadas.
Ready for delivery	Tareas que fueron validadas y listas para ser entregadas.
Done	Tareas que fueron entregadas.

Fuente: Los ejecutores.

Cuadro 15: Descripción de las columnas y las responsabilidades del equipo

Columna	Responsabilidad
Product backlog	Líder del proyecto y Entrenador.
Sprint backlog	Líder del proyecto y Entrenador.
Ready to development	Entrenador.
Development	Equipo Scrum.
Ready for acceptance	Equipo Scrum.
Acceptance	Líder del proyecto y Entrenador
Ready for delivery	Líder del proyecto y Entrenador.
Done	Equipo Scrum.

Fuente: Los ejecutores.

Para la codificación de las *features,* ítems del backlog, se definieron 2 grupos de códigos divididos por aplicación el primero **CK** para convenios y el segundo **OC** para portal.

En el proyecto, el ciclo representado por la fase de Explorar y adaptar fue repetido veinticuatro veces. En cada iteración (sprint), el *líder del proyecto* priorizo las tareas en columna de *Ready to development* en coordinación con el cliente.

4.2.2. Explorar y adaptar

En la fase Explorar y adaptar fueron realizados el desarrollo de los ítems del backlog del producto. Al término de cada iteración fueron producidos los incrementos del producto (conocido como lote de producción en la manufactura) de forma iterativa. Es decir, un nuevo incremento del producto se entregó a los clientes al término de cada sprint.

En el estudio de caso, conforme se ha descrito anteriormente, fueron realizadas veinticuatro iteraciones (sprints). El inicio de cada sprint era marcado por la realización de la Reunión de Planeamiento del sprint.

Esas reuniones eran conducidas siempre seguido del mismo padrón. El Cliente, juntamente con el Líder del Proyecto y el equipo, seleccionaron los ítems del backlog del producto que componían el backlog del sprint y serían trabajados en esa iteración.

Después del planeamiento, se cerraba la Reunión del Planeamiento del Sprint. Los miembros del equipo se reunían posteriormente con el cliente

para esclarecer dudas sobre los requisitos, o todavía, para validar las alternativas del desarrollo.

A lo largo de cada sprint, fueron realizadas las Reuniones Diarias (Dailies). Estas reuniones eran realizadas al inicio del día de trabajo, y nunca duraban más de 30 minutos. Cada miembro del equipo respondía las tres preguntas:

- ¿Qué hizo ayer?
- ¿Qué va hacer hoy?
- ¿Tiene algún impedimento?

A partir de las respuestas a esas preguntas permitía informar, al líder del proyecto y a los miembros del equipo, la situación de las tareas en desarrollo y tomar medidas de reubicación de desarrolladores, bloqueo de tareas y re-priorización, para no detener la fluidez del desarrollo.

Una de las actividades que también se realizaba en estas reuniones es hablar con anticipación de las tareas que están listas para ser desarrolladas, con el fin de reforzar el entendimiento y aclarar las dudas que surgen en el equipo de desarrollo.

Al final de cada sprint fue realizada una reunión, con una duración nunca superior a una hora, donde eran realizados los cierres de cada iteración presentando el incremento del producto. En estas reuniones el equipo traía informaciones y consideraciones sobre la estructura de trabajo, y el Entrenador (que en este caso también era el Líder del Proyecto) ajustaba

el modelo propuesto y lo adecuaba para el contexto y la realidad del proyecto.

En esta reunión, realizada al final de cada sprint, también era hecha la presentación del incremento del producto al cliente y al gerente. La presentación era conducida por el Líder del Proyecto, pero todo el equipo participaba. Después de la presentación, el cliente podía revisar el backlog del Producto, añadiendo o removiendo ítems.

La lista de todas las tareas desarrolladas del backlog del producto durante los 24 sprints se encuentran en el anexo D, de la misma forma las tareas no priorizadas están en el Anexo E.

4.2.3. Cierre

Al final del último sprint (veinticuatro) el proyecto fue considerado como concluido. Durante todo el proyecto también se identificaron otros procesos que quedaron fuera del ámbito de la investigación, estos son detallados en la sección de Recomendaciones y Sugerencias para su futura implementación.

En las figuras 16 y 17 se muestran los diagramas de entidad relación del sistema.

Cabe resaltar que los artefactos de análisis eran producidos sólo cuando eran necesarios o se percibía que agregaban valor, a lo largo del proyecto; sin embrago cuando estos no eran necesarios, el equipo del proyecto los ignoraba.

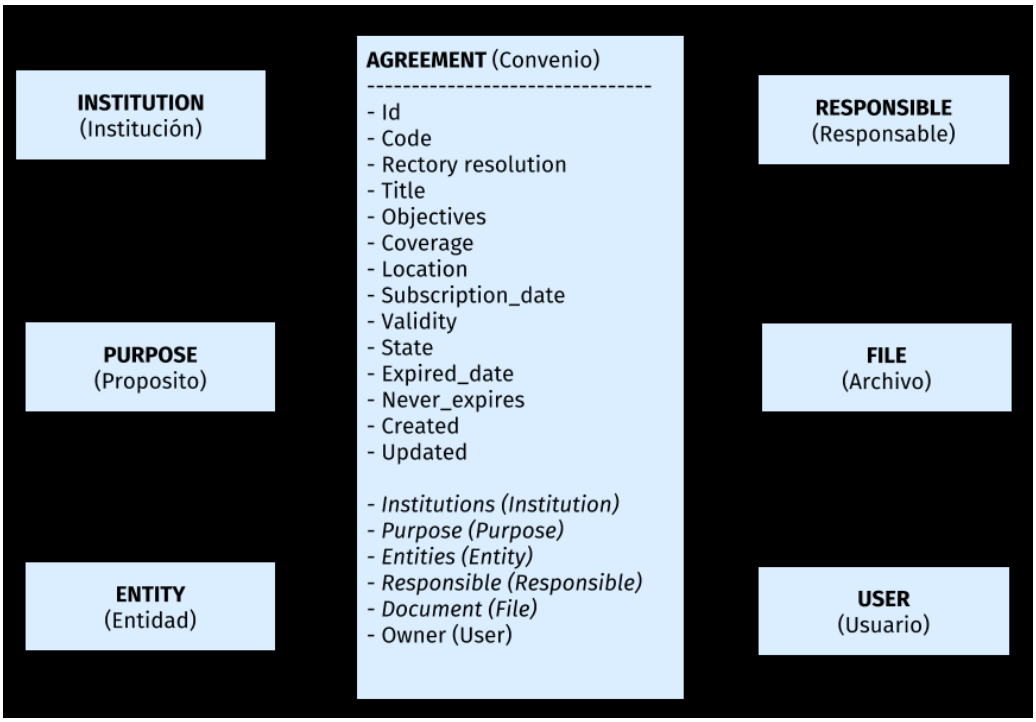

Figura 16: Diagrama de entidad relación del sistema - aplicación de convenios

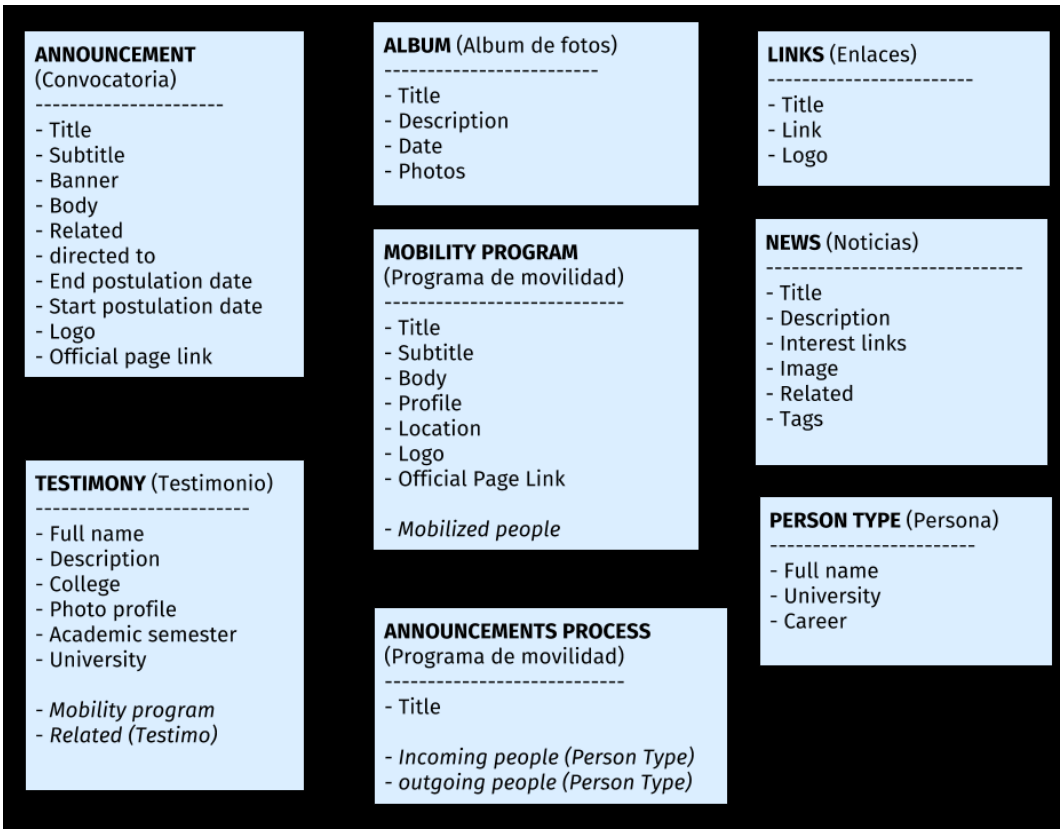

Figura 17: Diagrama de entidad relación del sistema – plataforma web

4.3. ANÁLISIS DE SATISFACCIÓN DE LOS USUARIOS RESPECTO AL SISTEMA IMPLANTADO

Los cuestionarios referentes a la satisfacción del usuario respecto al sistema implementado tuvieron como media 28.10 puntos, de una escala de 35 puntos. Entonces, se afirma que el nivel de satisfacción de los usuarios fue alto respecto a la facilidad del uso del sistema.

Cuadro 16: Estadísticos descriptivos de la variable satisfacción del usuario

Estadístico	Valor
N	60
Media	28.10
Desviación estándar	3.96

Fuente: Los ejecutores.

En el Cuadro 17 son mostradas las medias en orden decreciente de los ítems referentes a la satisfacción del usuario, donde se puede observar que ítem 7, *tiempo de respuesta del sistema* que se refiere a la velocidad de carga y transición de páginas del sistema en el navegador, recibió la mayor valoración. El ítem 4 tiene la menor valoración, es debido a que no se cuenta con información completa en algunas secciones del portal como programas de movilidad, por otro lado, no se cuenta con el personal suficiente para la administración del sistema.

Cuadro 17: Medias de los ítems de la medida de satisfacción de usuario

Orden	Ítem	Satisfacción de usuario	Media
1	7	Tiempo de respuesta del sistema (velocidad de carga)	4.20
2	1	Navegación por el sistema	4.15
3	3	Claridad de las páginas y listados	4.15
4	6	Frecuencia de actualización de las informaciones	4.05
5	2	Claridad de los mensajes	4.00
6	5	Adecuación de la presentación (responsivo)	4.00
7	4	Cantidad de información	3.55

Fuente: Los ejecutores.

De acuerdo a la encuesta de satisfacción de usuario que se aplicó a los usuarios, se observa que el 10% de ellos tuvieron un nivel de satisfacción medio respecto a la facilidad de uso del sistema, mientras que el 55% y el 35% tuvieron un nivel alto y muy alto respectivamente.

Cuadro 18: Cuadro de distribución de frecuencias de la variable satisfacción de usuario

Nivel de satisfacción	fi	Hi	%
Medio	6	0.10	10.0
Alto	33	0.55	55.0
Muy alto	21	0.35	35.0
	60	1.00	100

Fuente: Los ejecutores.

4.4. ANÁLISIS DE LA UTILIDAD DEL SISTEMA IMPLANTADO

Los cuestionarios referentes a la utilidad del sistema, respecto al punto de vista del personal de la oficina, tuvieron como media 28 puntos, en una escala de 35 puntos. Por tanto, se afirma que el nivel de utilidad del sistema desde el punto de vista organizacional y funcional es alto.

Cuadro 19: Estadísticos descriptivos de la variable utilidad del sistema

Estadístico	Valor
N	4
Media	28.00
Desviación estándar	1.41

Fuente: Los ejecutores.

En el Cuadro 20 son mostradas las medias en orden decreciente de los ítems referentes a la utilidad del sistema, donde se puede observar que el ítem 1, *mejoría en el flujo de trabajo* recibió la mayor valoración, porque con el sistema se redujo el tiempo de ejecución de las tareas y actividades. El ítem 6, disponibilidad del sistema tuvo la menor valoración, y no significa que sistema quede fuera de servicio; muy por el contrario, fue debido a la latencia de la red dentro del edificio administrativo universitario.

Cuadro 20: Medias de los ítems de la medida de utilidad del sistema

Orden	Ítem	*Utilidad del Sistema*	Media
1	1	Mejoría en el flujo de trabajo	4.50
2	5	Reducción en el tiempo de ejecución de las tareas	4.25
3	3	Adquisición de nuevos conocimientos	4.00
4	2	Mejor utilización del tiempo	4.00
5	7	Mejoría en el ambiente de trabajo	4.00
6	4	Autonomía en relación a otros	3.75
7	6	Disponibilidad del sistema (siempre on-line)	3.50

Fuente: Los ejecutores.

De acuerdo a la encuesta de utilidad del sistema aplicada al personal de la Oficina de Cooperación Nacional e Internacional, se observó que el 25% de ellos consideró que el nivel de utilidad del sistema es medio, mientras que el 75% del personal de la oficina afirmó que el nivel es alto.

Cuadro 21: Cuadro de distribución de frecuencias de la variable utilidad del sistema

Nivel de Utilidad	fi	hi	%
Media	1	0.25	25%
Alta	3	0.75	75%
	4	1	100%

Fuente: Los ejecutores.

CONCLUSIONES

En los principales procesos y actividades de las unidades de la Oficina de Cooperación Nacional e Internacional, se determinó que la *Unidad de Convenios Nacionales e Internacionales* y la *Unidad Movilidad de Estudiantes, Docentes, Investigadores y Personal Administrativo* fueron las que presentaron mayores deficiencias en la gestión documental referente a convenios, programas de movilidad y otras informaciones relacionadas a ésta última como son convocatorias, becas y testimonios.

El análisis de satisfacción de los usuarios sobre la facilidad del uso del sistema implantado a través de las encuestas; determinó que el nivel de satisfacción es alto, obteniendo una media de 28.1 puntos en una escala de 35 puntos; además el 90% de los usuarios afirmaron que el nivel de satisfacción es alta y muy alta.

El análisis de la utilidad del sistema implementado desde el punto de vista organizacional y funcional a través de la perspectiva del personal de la oficina, determinó que el nivel de la utilidad del sistema es alto, alcanzando una media de 28 puntos en una escala de 35 puntos; y así mismo se observó que el 75% del personal afirmaron que tiene un nivel de utilidad alta.

Se afirma que la evaluación de la gestión documental de la Oficina de Cooperación Nacional e Internacional de la Universidad Nacional del Altiplano a través de un sistema, mejoró favorablemente en la *distribución más eficiente y control sobre los documentos,* proporcionando una búsqueda y recuperación de documentos más rápida. Además, se tiene autonomía en la publicación y difusión, facilitando que los estudiantes, docentes y otros usuarios puedan

acceder de forma oportuna a la información; como también una mejora general en el proceso de trabajo y la eficiencia organizacional.

RECOMENDACIONES Y SUGERENCIAS

Se sugiere crear una aplicación que recomiende, las universidades de destino, de acuerdo a sus cursos, carrera profesional y semestre académico, para facilitar la postulación de los estuantes a los programas de movilidad.

Se recomienda utilizar las notas ponderadas de todos los estudiantes de la universidad para clasificarlos y promover su participación a los programas de movilidad a través de sus correos electrónicos.

Se recomienda utilizar la técnica del *Mapeo de Historias de Usuario* (User Story Mapping) para analizar los procesos y actividades de gestión documental de oficinas administrativas, como complemento al modelo presentado por Franco (2007).

Se recomienda coordinar la difusión y la publicación de convocatorias, becas, ofertas académicas con la oficina de Imagen Institucional de la Universidad.

Se sugiere sistematizar el seguimiento de la ejecución de los convenios suscritos; para poder medir el impacto que estos tienen en la Universidad Nacional del Altiplano.

Se sugiere realizar futuros estudios de Minería de Datos (Data Mining) con la información de estudiantes movilizados, por carrera profesional, área de estudio, género, edad y otras variables de interés.

BIBLIOGRAFÍA

Andrade, G. y Falk, J. A. (2001). *Eficácia de sistemas de informação e percepção de mudança organizacional: um estudo de caso*. Revista de administração contemporánea, 5(3):53-84.

Bautista J. (2009). *Programación extrema (XP).*

Beck, K. (1999). *Extreme Programming explained: Embrace change. Reading, Mass*.

Bitbucket. (2016). *Documentación.* Recuperado de https://docs.bitbucket.org

Bustelo, C. (2011). *Serie ISO 30300: Sistema de gestión para los documentos*.

Cantú, S. O., y Zapata, Á. R. P. (2006). *¿Que es la Gestión de la Innovación y la Tecnología (GInnT)?.* Journal of Technology Management & Innovation, 1(2), 64-82.

Casas, I. (2013). *Taller de Herramientas Computacionales para Ingeniería*.

Choko. (2016). *About*. Recuperado de http://choko.org/about

Definición. (2017). Definición de gestión Recuperado de …….. https://definicion.mx/gestion/

Drupal. (2017). *Documentation: Understanding Drupal 8 - Overview*. Recuperado de https://www.drupal.org/docs/8/understanding-drupal-8/overview

Eduardo, F. (2007). *Um modelo de gerenciamento de projetos baseado nas Metodologias Ágeis de Desenvolvimento de software e nos princípios da produção enxuta*. PhD thesis, Universidade de São Paulo.

Gauchi, V. (2012). *Aproximación teórica a la relación entre los términos gestión documental, gestión de información y gestión del conocimiento*. Revista española de documentación científica, 35(4):531-554.

Gil, F. (2011). *Experto en Drupal 7*.

Haddad, F. y del Huerto, M. (2014). *Tutorial básico de Trello*.

Izquierdo, C. y Paolo, J. (2015). *Sistema informático web de trámite documentario para la UGEL de Zarumilla tumbes utilizando los frameworks AngularJS y Spring MVC*.

Laudon, K. y Laudon, J. (2004). *Sistemas de información gerencial: administración de la empresa digital. Pearson Educación*.

Llorens, P. (2014). *Metadirectorios bajo node.js*.

Lopez-Pellicer, F. y Bejar, R. y. L. M. A. y. N.-I. J. y. Z.-S. F. J. (2015). *Github como herramienta docente.* In Actas de las XXI Jornadas de la Enseñanza Universitaria de la Informática, pages 66-73. Universidad Oberta La Salle.

Martín, M. (2010). *Filosofía Lean aplicada a la ingeniería del software*. Universidad de Sevilla.

McGinnis, B. (2017). *Architecture Blog: Isomorphic Architecture*. Recuperado de https://brianmcginnisarchitect.wordpress.com/systems-architecture/isomorphic-architecture/

Min-Seok, P. (2011). *Information Technology and value creation in the Public Sector Organizations*.

MongoDB. (2016). *Documentation*. Recuperado de https://docs.mongodb.com

Orostegui, F. (2013). *Mejores prácticas de despliegue continuo para aplicaciones Synfony2*.

Taller Blog. (2016). *Introdução ao Docker*. Recuperado de ..... https://blog.taller.net.br/introducao-ao-docker/

Patton, J. (2016). *User Story Mapping.* Recuperado de: …… https://jpattonassociates.com/user-story-mapping

Poppendieck, M. y Poppendieck, T. (2003). *Lean software development: an agile toolkit. Addison-Wesley Professional*.

Quispe, G. (2014). *Sistema informático de gestión administrativa para la Coordinación de Investigación de la Facultad de Ingeniería Estadística e Informática de la UNA Puno*. Universidad Nacional del Altiplano de Puno.

Roberge, M. (2006). *Lo esencial de la gestión documental: sistema integrado de gestión de los documentos analógicos y de los documentos electrónicos*. Number 025.1 R638e.

Romero, F. (2014). *Actualización y generación de módulos para una web universitaria sobre Drupal*.

Sangeap D., A. (2015). Tutorial/curso web components.

Schwaber, Ken y Sutherland, J. (2011). *Guia de Scrum. La guía definitiva de scrum*: Las reglas del juego.

Shapiro, R. (2014). *Software & Information Industry Association*.

Shingo, S. (1981). Study of Toyota *Production System from Industrial Engineering Viewpoint: Produce What Is Needed, When It's Needed*.

Sutherland, J. y Schwaber, K. (1995). *The Scrum Methodology*. *In Business object design y implementation*: OOPSLA workshop.

Zapata, P. y Bermúdez, M. (2012). *Prototipo de una aplicación web para un centro gerontológico*. Universidad Tecnológica de Pereira, Disponible en: http://recursosbiblioteca.utp.edu.co/tesisd/textoyanexos/0053L847p.pdf, pages 32-33.

ANEXOS

A: PLANTILLA DE HISTORIA DE USUARIO

Historia de Usuario .....

Yo como [usuario, cliente, operador, otros.]: …….....

Quiero / necesito / deseo: …….....

Para que / a fin de que / de modo que: …….....

Criterio de aceptación: …….....

Prioridad: / 1 / 2 / 3/ 4 / 5 /

B: ENCUESTA DE SATISFACCIÓN DE USUARIO

Encuesta de satisfacción de usuarios respecto al sistema de gestión documental de la Oficina Nacional e Internacional de la UNA-Puno

Marque con una [X] sólo una respuesta que considere conveniente.

I. SATISFACCIÓN DEL USUARIO.

En cuanto a la satisfacción de usuario respecto al sistema, cómo evalúa los siguientes ítems:

1. Navegación por el sistema

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

2. Claridad de los mensajes

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

3. Claridad de las páginas y listados

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

4. Cantidad de información

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

5. Adecuación de la presentación (responsivo)

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

6. Frecuencia de actualización de las informaciones

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

7. Tiempo de respuesta del sistema (velocidad de carga)

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

C: ENCUESTA DE SATISFACCIÓN DE USUARIO

Encuesta de utilidad del sistema de gestión documental de la Oficina Nacional e Internacional de la UNA-Puno

Marque con una [X] sólo una respuesta que considere conveniente.

I. UTILIDAD DEL SISTEMA.
En cuanto a la utilidad del sistema cómo evalúa los siguientes ítems:

1. Mejoría en el flujo de trabajo

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

2. Mejor utilización del tiempo

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

3. Adquisición de nuevos conocimientos

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

4. Autonomía en relación a otros

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

5. Reducción en el tiempo de ejecución de las tareas

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

6. Disponibilidad del sistema (siempre on-line)

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

7. Mejoría en el ambiente de trabajo

   a. Deficiente **b.** Mala **c.** Regular **d.** Bueno **e.** Muy bueno

D: ÍTEMS DESARROLLADOS DEL BACKLOG DEL PRODUCTO EN LOS SPRINTS

Convenios

I) DISEÑO:

- ✓ OC-006-01**:** Selección de diseños de referencia para la aplicación.
- ✓ OC-006-07**:** Diseñar mockups para el diseño desktop de la página.
- ✓ OC-004-100: Crear mockups para el diseño de la página principal de la aplicación convenios.
- ✓ OC-004-27: Diseño del panel de convenio para la lista de convenios.

II) INFRAESTRUCTURA:

- ✓ OC-004-12: Crear ambiente de pruebas con: Protractor, Selenium y Moca.
- ✓ OC-004-27: Configurar integración continua del ambiente de producción.

III) REPORTES:

- ✓ CK-105: Generar reporte en formato XLSX.

IV) CONVENIOS:

- ✓ OC-004-3: Mejorar el algoritmo de vigencia de los convenios. Debe ser posible definir la vigencia de en días, años y semanas.
- ✓ OC-004-5: Eliminar los nombres de convenios incorrectos.
- ✓ OC-004-6: Modificar el tipo Responsable por Responsables en plural, en el formulario de registro y edición de convenios.
- ✓ OC-004-11: Adicionar descripciones a los campos del formulario de registro y edición de convenios.
- ✓ OC-004-13: Actualizar nombres de los campos del formulario de registro de convenios.
- ✓ OC-004-14: Permitir que el campo tipo de institución sea multi select (select-ui).

- ✓ OC-004-15: Permitir que el campo objetivo tenga opciones de formato de fuente, utilizar *Sumer note*.
- ✓ OC-004-16: Agregar los campos cargo y nombre al responsable del convenio.
    - Crear un tipo para poder almacenar los datos del responsable (cargo, nombre).
    - Éste tipo será un *inline reference* múltiple en el convenio.
- ✓ OC-004-23: Mejorar el template de la lista de convenios. quitar el botón *Eliminar* convenio.
- ✓ OC-004-28: Adicionar el campo *file(archivo)* para poder adjuntar archivos pdf.
- ✓ OC-004-32: Mejorar la eliminación de documentos en Solr.
- ✓ OC-100: Mejorar el reporte de convenios.
- ✓ CK-101: Mejorar el campo vigencia.
- ✓ CK-102: Mejora de estilos de los formularios.
- ✓ CK-104: Adicionar botón "Imprimir" en la lista de convenios.
- ✓ CK-200: Mejorar el campo archivo del convenio.
- ✓ CK-201: Mejorar del campo de vigencia en el formulario de convenios.
- ✓ CK-201: Borrar los convenios duplicados.
- ✓ CK-202: Adicionar por resolución.
- ✓ CK- 203: Adicionar botón para *Eliminar* convenios.
- ✓ CK-204: Eliminar campo de *resolución*.
- ✓ EF-03: Mejorar el formulario de convenios.

V) BÚSQUEDA:

- ✓ POC: Validar la integración de Node.js y Solr.
- ✓ OC-004-7: Incrementar el rango del filtro de años, máximo 30 años.
- ✓ OC-004-18: Adicionar filtro por vigencia
- ✓ OC-004-19: Actualizar el formulario de búsqueda.

- ✓ OC-004-26: Agregar filtro por código y agregar el campo código al tipo *Convenio*.
- ✓ OC-004-30: Disminuir el tamaño de letra en la barra de búsqueda.

VI) MIGRACIÓN:

- ✓ OC-004-5: Actualizar el campo responsable de los convenios registrados.
- ✓ OC-004-20: Migrar los convenios de Mysql a MongoDB.
- ✓ OC-004-21: Migrar el repositorio de Github a Bitbucket:

VII) NAVEGACIÓN:

- ✓ OC-004-29: Fijar la sección del pie de página en la parte baja de la aplicación

---

Portal

I) DISEÑO / COMPONENTES

- ✓ DI-1: Diseñar los Mockups del portal.
- ✓ DI-2: Diseñar los Layouts del portal.
- ✓ OC-114: Actualizar la página principal del sitio.
- ✓ OC-125: Mejorar el pie de página.
- ✓ OC-137: Mejorar los estilos del sitio.
- ✓ OC-153: Mejorar el panel de títulos.
- ✓ OC-155: Adicionar estilos responsivos para Tabletas y Móviles.
- ✓ OC-158: Adicionar un título a la página.
- ✓ OC-169: Agregar links a los títulos del pie de página.
- ✓ OC-305: Actualizar el tema con la segunda version.
- ✓ OC-144: Agregar la componente Breadcrumbs a las páginas principales.

II) INFRAESTRUCTURA:

- ✓ OCN-08: Configurar ambientes de desarrollo local con Docker.

- ✓ OCN-09: Configurar la integración continua de la aplicación con SemaphoreCI.

III) CONVOCATORIAS:

- ✓ V2-1: Crear tipo de contenido para registrar Convocatorias, con los siguientes campos: título, cuerpo, logo y documentos adjuntos.
- ✓ V2-5: Crear la vista para listar convocatorias.
- ✓ OC-101: Adicionar endpoint para servir convocatorias.
- ✓ OC-115: Actualizar Panel de convocatorias para utilizar un Slider
- ✓ OC-124: Crear página para mostrar la información de convocatorias.
- ✓ OC-152: Mejorar Slider de convocatorias.
- ✓ OC-156: Adicionar link de descarga para el reporte de convenios.
- ✓ OC-163: Configurar urls amigables para las convocatorias.
- ✓ OC-307: Mejorar las convocatorias.

IV) TESTIMONIOS:

- ✓ OC-112: Crear tipo de contenido para almacenar Testimonios.
- ✓ OC-113: Crear panel para listar testimonios.
- ✓ OC-118: Mejorar la componente que lista los testimonios.
- ✓ OC- 127: Crear página para visualizar a lista de todos testimonios.
- ✓ OC-150: Adicionar botón “Ver más testimonios”.
- ✓ OC-302: Crear la página visualizar cada testimonio.
- ✓ OC-303: Disminuir la cantidad del bloque de testimonios mostrados.
- ✓ OC-308: Mejorar los estilos de la página de testimonios.

V) CONVENIOS:

- ✓ OC-120: Refactorar endpoint de convenios.
- ✓ OC-130: Adicionar panel convenios a la página principal.
- ✓ OC-133: Adicionar modal y formulario para la búsqueda para los convenios.
- ✓ OC-135: Agregar estilos a la página de convenios.

- ✓ OC-136: Adicionar panel para cada convenio.
- ✓ OC-141: Adicionar el botón “ver más convenios”.
- ✓ OC-143: Resolver bug de limpiar el formulario de convenios.
- ✓ OC-157: Mejorar el reporte de convenios.
- ✓ OC-160: Adicionar componente para el modal responsivo.
- ✓ OC-320: Agregar paginación a la lista de convenios.
- ✓ OC-321: Adicionar filtro por resolución al formulario de búsqueda de convenios.

VI) PROGRAMAS DE MOVILIDAD:

- ✓ V2-3: Crear un tipo de contenido para almacenar programas de movilidad con los siguientes campos: título, logo y descripción.
- ✓ V2-6: Crear vista para listar programas de movilidad.
- ✓ OC-02: Crear panel de listar programas de movilidad en la página principal.
- ✓ OC-111: Actualizar endpoint de programas de movilidad.
- ✓ OC-117: Refactoring de los programas de movilidad.
- ✓ OC-131: Mejorar los programas de movilidad.
- ✓ OC-164: Adicionar una vista para cada programa de movilidad.
- ✓ OC-170: Adicionar un menú para los programas de movilidad.

VII) NOSOTROS:

- ✓ V2-4: Crear entidad para almacenar Personal con los siguientes campos: nombre, foto, oficina/unidad, cargo, teléfono, celular y email.
- ✓ OC-122: Crear página para mostrar la misión, visión y objetivos de la oficina (Acerca de nosotros).
- ✓ OC-123: Crear página para mostrar información de funciones de la oficina.
- ✓ OC-127: Mejorar página de personal de la oficina.

VIII) CONTACTO:

- ✓ OC-151: Crear formulario de contacto.

- ✓ OC-161: Almacenar datos del formulario de contacto.
- ✓ OC-165: Crear mapa de ubicación de la oficina.

IX) ALBUM:

- ✓ OC-132: Adicionar panel de imagen.
- ✓ OC-140: Añadir estilo de imagen para tener un recorte fijo en las imágenes del álbum.
- ✓ OC-168: Adicionar página para mostrar las imágenes del álbum.
- ✓ OC-301: Incrementar la cantidad del bloque de imágenes a ser visualizados.
- ✓ OCN-02: Adicionar link de "ver más" al panel de imágenes.

X) ANALYTICS:

- ✓ OC-154: Adicionar soporte de *Google* Analytic.
- ✓ OC-167: Adicionar etiquetas *Metatags*.
- ✓ OC-310: Adicionar *Google Analytics Events* para rastrear como interactúan los usuarios con nuestro sitio.

XI) NAVEGACIÓN:

- ✓ OC-142/ OC-139: Adicionar el campo "*Enlaces relacionado*" a los tipos de contenidos testimonio y convocatoria.
- ✓ OC-145: Adicionar componente Bread Crumbs a las páginas existentes.
- ✓ OC-320: Adicionar componente de paginación a las páginas existentes.
- ✓ OCN-01: Adicionar enlace "Ver más" al panel de últimos convenios.
- ✓ OCN-05: Ejecutar el evento scroll top después de cambiar una página.
- ✓ OC-114: Actualizar el menú de la página principal.

XII) SEO:

- ✓ OC-N01: Adicionar favicon.
- ✓ OCN-04: Registrar la dirección del portal en Google webmaster.

- ✓ OCN-06: Actualizar los datos de la página de Facebook y enlazarlo con el sitio web.
- ✓ OCN-07: Configurar URLs amigables.

XIII) EFICIENCIA:

- ✓ EF-01: Refactoring de módulos de React.
- ✓ OC-102: Refactoring de modulos React.
- ✓ OC-159: Mejorar el rendimiento de carga de la página.

E: ÍTEMS NO PRIORIZADOS DEL BACKLOG DEL PRODUCTO

Convenios

- ✓ OC-004-1: Adicionar filtro por rango de años.
- ✓ OC-004-2: Mejorar el filtro por facultades.
- ✓ OC-004-4: Crear un CRUD para el registro de instituciones. Agregar las opciones: Banco e Institutos
- ✓ OC-004-8: Actualizar los datos de las entidades, instituciones, propósitos y responsables.
- ✓ OC-004-10: Ocultar los campos que no están en uso de los formularios de edición y registro de convenios.
- ✓ OC-004-24: Utilizar tags para los campos: instituciones, propósitos y entidades.
- ✓ OC-005-2: Mejorar el reporte de PDF.

Portal

I) NOTICIAS:

- ✓ JAN-21: Adicionar página de noticias.
- ✓ OC-119: Crear componente de Noticias.
- ✓ OC-121: Adicionar endpoint para las noticias.

II) ENLACES:

- ✓ V2-2: Crear un tipo de contenido para almacenar Enlaces.
- ✓ V2-7: Crear panel para listar los Enlaces de Interés.

III) EVENTOS:

- ✓ WS-4: Crear un tipo de contenido para almacenar eventos con los siguientes campos: título, descripción, imagen y fecha de publicación.