

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA



PONTIFICIA
**UNIVERSIDAD
CATÓLICA**
DEL PERÚ

DESARROLLO DE UNA HERRAMIENTA DE SOPORTE A LA GESTIÓN DE PROYECTOS ÁGILES PARA EQUIPOS DISTRIBUIDOS

Tesis para optar por el Título de Ingeniero Informático, que presenta el bachiller:

Zoila Palza Chávez

ASESOR: Luis Alberto Flores

Lima, Junio del 2013

RESUMEN

La industria de software, es una industria globalizada, por esta razón resulta cada vez más común trabajar con equipos distribuidos, en diferentes locaciones geográficas. Estas organizaciones de Tecnologías de Información continuamente tienen que adaptar sus procesos, reducir costos e incrementar la calidad de sus productos. Es por esto que muchas organizaciones optaron por la adaptación de los procesos de desarrollo para que sean ágiles y sencillos (Yaggahavita, 2011).

El presente proyecto de fin de carrera se realiza con el objetivo de proponer una solución informática de soporte a la gestión de proyectos ágiles de desarrollo de software para equipos distribuidos, sin añadirle una burocracia innecesaria.

En primer lugar se presenta un estudio sistemático de la literatura existente, sobre metodologías ágiles, y de las prácticas recomendadas para ambientes distribuidos globalmente. Y también, se presenta una comparación de la tecnología existente que pretende solucionar el problema encontrado.

El producto final propuesto, se desarrolló mediante iteraciones continuas, de análisis, diseño e implementación, haciendo uso de prácticas ágiles de desarrollo de software y bajo la supervisión del asesor del proyecto de fin de carrera.

La planificación del proyecto se realizó tomando en cuenta los lineamientos de prácticas ágiles. Al inicio del proyecto se realizó una planificación considerando las limitaciones de tiempo con las que se cuenta, pero, la planificación detallada de las actividades en cada iteración se resolvió al inicio de la misma.







ÍNDICE

1.	Generalidades.....	1
1.1.	Definición del problema	1
1.2.	Objetivo general	4
1.3.	Objetivos específicos.....	4
1.4.	Resultados esperados	4
1.5.	Marco Conceptual	4
1.5.1.	Inicios de las metodologías ágiles	5
1.5.2.	Metodologías Ágiles	6
1.5.3.	Metodología Scrum	6
1.5.4.	Scrum distribuido.....	8
1.5.5.	Prácticas de Scrum en entornos distribuidos	9
1.6.	Estado del Arte	10
1.6.1.	Herramientas existentes	10
1.6.2.	Discusión sobre los resultados.....	13
1.6.3.	Casos de éxito.....	16
1.7.	Descripción y Justificación de la solución	19
1.7.1.	Alcance	19
1.7.2.	Limitaciones	20
1.7.3.	Justificación.....	20
2.	Planificación del proyecto	22
2.1.	Procesos de Gestión e ingeniería.....	22
2.1.1.	Viabilidad.....	26
2.2.	Plan de proyecto.....	27
3.	Requisitos	30
3.1.	Actores	30
3.2.	Pila del Producto	31
3.2.1.	Leyenda	37
3.3.	Sprint 1	37
3.3.1.	Pila del Sprint	37
3.3.2.	Gráfico Burn-Down	40
3.3.3.	Retrospectiva del Sprint.....	40
3.4.	Sprint 2.....	41
3.4.1.	Pila del Sprint	41

3.4.2.	Gráfico Burn-Down	43
3.4.3.	Retrospectiva del Sprint.....	43
3.5.	Sprint 3.....	44
3.5.1.	Pila del Sprint	44
3.5.2.	Gráfico Burn-Down	46
3.5.3.	Retrospectiva del Sprint.....	46
3.6.	Sprint 4.....	47
3.6.1.	Pila del Sprint	47
3.6.2.	Gráfico Burn-Down	48
3.6.3.	Retrospectiva del Sprint.....	48
4.	Diseño.....	49
4.1.	Arquitectura de la solución.....	49
4.1.1.	Acrónimos	49
4.1.2.	Definición de la Arquitectura.....	50
4.1.3.	Representación de la arquitectura.....	50
4.1.4.	Vista lógica	51
4.1.5.	Vista de Implementación	54
4.1.6.	Vista de Despliegue.....	55
4.2.	Diseño de la Interfaz gráfica	55
5.	Construcción.....	61
5.1.	Construcción	61
5.1.1.	Tecnología: Lenguaje PHP.....	61
5.1.2.	Framework: CakePHP.....	62
5.2.	Pruebas.....	63
5.2.1.	Sprint 1.....	64
5.2.2.	Sprint 2.....	68
5.2.3.	Sprint 3.....	71
5.2.4.	Sprint 4.....	74
6.	Conclusiones y Recomendaciones	1
6.1.	Conclusiones.....	1
6.2.	Recomendaciones y Trabajos futuros.....	2
	Referencias	3
	Anexos	

Anexo A: Prototipos de Diseño

Anexo B: Listado y Definición de técnicas de métodos ágiles

Anexo C: Mecanismos de Colaboración y Comunicación



ÍNDICE DE TABLAS

Tabla 1.1: CUADRO COMPARATIVO DE LAS PRINCIPALES HERRAMIENTAS EXISTENTES (THE LINUX FOUNDATION).....	13
Tabla 2.1: CUADRO DE ACTIVIDADES.....	25
Tabla 2.2: TABLA DE COSTOS ESTIMADOS DEL PROYECTO	27
Tabla 3.1: PILA DEL PRODUCTO.....	31
Tabla 3.2: TÉCNICA MOSCOW	37
Tabla 3.4: RETROSPECTIVA DEL SPRINT 1	40
Tabla 3.5: PILA DEL SPRINT 2	41
Tabla 3.6: RETROSPECTIVA DEL SPRINT 2	43
Tabla 3.7: PILA DEL SPRINT 3	44
Tabla 3.8: RETROSPECTIVA DEL SPRINT 3	46
Tabla 3.9: PILA DEL SPRINT 4	47
Tabla 5.1: PRUEBAS DE ACEPTACIÓN DEL SPRINT 1	64
Tabla 5.2: PRUEBAS DE ACEPTACIÓN DEL SPRINT 2	68
Tabla 5.4: PRUEBAS DE ACEPTACIÓN DEL SPRINT 4	74

ÍNDICE DE ILUSTRACIONES

Ilustración 1.1: EQUIPOS DISTRIBUIDOS CON HORAS DE TRABAJO SUPERPUESTAS (WOODWARD, SURDEK, & GANIS, 2010).....	8
Ilustración 1.2: EQUIPOS DISTRIBUIDOS SIN HORAS DE TRABAJO SUPERPUESTAS (WOODWARD, SURDEK, & GANIS, 2010).....	9
Ilustración 1.3: DESARROLLO DE PORTAL V6.0 (WOODWARD, SURDEK, & GANIS, 2010).....	17
Ilustración 2.1: ESTRUCTURA DE DIVISIÓN DEL TRABAJO.....	28
Ilustración 2.2: DIAGRAMA DE GANTT.....	29
Ilustración 3.1: DIAGRAMA DE ACTORES.....	30
Ilustración 3.2: GRÁFICO BURN-DOWN DEL SPRINT 1.....	40
Ilustración 3.3: GRÁFICO BURN-DOWN DEL SPRINT 2.....	43
Ilustración 3.4: GRÁFICO BURN-DOWN DEL SPRINT 3.....	46
Ilustración 3.5: GRÁFICO BURN-DOWN DEL SPRINT 4.....	48
Ilustración 4.1: VISTA GENERAL DE LA ARQUITECTURA.....	50
Ilustración 4.2: Modelo-Vista-Controlador.....	50
Ilustración 4.3: DIAGRAMA DE CLASES DE DISEÑO.....	53
Ilustración 4.4: DIAGRAMA DE COMPONENTES.....	54
Ilustración 4.5: DIAGRAMA DE DESPLIEGUE.....	55
Ilustración 4.6: PROTOTIPO DE INFORMACIÓN DEL PROYECTO.....	56
Ilustración 4.7: PROTOTIPO DE VISUALIZACIÓN DE LA PILA DEL PRODUCTO (1).....	57
Ilustración 4.8: PROTOTIPO DE VISUALIZACIÓN DE LA PILA DEL PRODUCTO (2).....	57
Ilustración 4.9: PROTOTIPO DE VISUALIZACIÓN DE LA PILA DEL PRODUCTO (3).....	58
Ilustración 4.10: PROTOTIPO DE REVISIÓN DEL SPRINT.....	58
Ilustración 4.11: PROTOTIPO DE Retrospectiva.....	59
Ilustración 4.12: PROTOTIPO DE CHAT.....	59
Ilustración 4.13: PROTOTIPO DE FORO.....	60
Ilustración 5.1: MVC CON CAKEPHP.....	63
Ilustración 5.2: PROCESO DE PRUEBAS POR SPRINT.....	63

1. GENERALIDADES

El presente capítulo se inicia con una definición del problema y luego se aborda los aspectos generales y fundamentales del proyecto, como los objetivos y los resultados del mismo.

1.1. DEFINICIÓN DEL PROBLEMA

El uso de metodologías ágiles para el desarrollo de software es una creciente tendencia, las empresas de desarrollo adoptan estas metodologías para lograr una mejor respuesta a los cambios frecuentes de requerimientos, reducir el tiempo de salida del software al mercado, y ofrecer software que con mayor precisión satisfaga las necesidades de sus clientes (Sutherland, Schoonheim , & Rijk, 2009). Cuando las metodologías ágiles fueron introducidas a mediados de los 90's (Woodward, Surdek, & Ganis, 2010), era algo común contar con equipos de desarrollo colocados, es decir todo el equipo del proyecto estaba en un mismo lugar; hoy en día, un equipo de desarrollo distribuido en diferentes locaciones es casi una norma (VersionOne, 2010), y los miembros del equipo pueden incluso, estar dispersos en varios continentes.

Las metodologías ágiles, promueven un mecanismo iterativo para producir software que se basan en una constante comunicación entre desarrolladores y clientes. El visionario ágil, Kent Beck, puso en tela de juicio el costo tradicional de la curva de cambio, en los proyectos de software, evidenciado por Barry Boehm más de veinte años antes. El modelo de Beck, propone que el costo del cambio puede ser bajo, incluso al final del ciclo de vida del proyecto. Para lograr este “bajo” costo en la curva de cambio, las metodologías ágiles promueven prácticas de ingeniería que permiten un cambio rentable (Fowler, 2004). Martin Fowler, afirma que las pruebas constantes y la integración continua son las principales prácticas ágiles que permiten obtener las ventajas de producción rápida y mínimos diseños up-front (Fowler, 2004).

Considerando la competencia que existe en una industria tan dinámica, las organizaciones de tecnologías de información (TI), continuamente tienen que adaptar sus procesos, reducir costos e incrementar la calidad de sus productos. Esto favoreció a que muchas organizaciones optaran por el uso de subcontratación y la adaptación de los procesos de desarrollo para que sean ágiles y sencillos (Moore, 2004).

Empresas de Estados Unidos, Europa o Japón, suelen hacer subcontratación del desarrollo de software en países de Europa Oriental, Rusia o del medio Oriente (Suttherland, Schoonheim, Kumar, Pandey V., & Vishal, 2009). Las tres principales ventajas que se busca son reducir los costos laborales, captación de talento que no esté disponible localmente y aumentar o disminuir el tamaño del proyecto, sin despidos (Suttherland, Schoonheim, Kumar, Pandey V., & Vishal, 2009).

Cuando estas organizaciones inician el uso de procesos ágiles y subcontratación, es cuando se cree que pueden unir lo mejor de dos mundos, podrían reducir costos usando subcontratación y satisfacer las necesidades de los clientes usando métodos ágiles de desarrollo (Yaggahavita, 2011).

A menudo, cuando se realiza la subcontratación, los equipos de desarrollo pueden estar distribuidos a nivel mundial, esta característica muchas veces tiene un impacto significativo en la comunicación, coordinación y control, y mitigar el impacto representa un desafío (Dullemond & Gameren, 2009).

En Scrum, un popular método ágil de gestión de proyectos, se sostiene que los procesos simples definidos por sí solos no se pueden utilizar para gestionar

eficazmente los proyectos de software complejos y dinámicos; por este motivo, en Scrum se propone un enfoque empírico y adaptativo de gestión, que se emplea para medir y ajustar periódicamente el proceso y lograr el producto final deseado. Como resultado, los planes del proyecto son continuamente inspeccionados y adaptados a la realidad del proyecto (Szalvay, 2004).

Uno de los artefactos más importantes en Scrum es la pila del producto (*product backlog*), una lista de requisitos dinámica; los componentes de la pila del producto pueden ser añadidos o eliminados durante el proyecto. Es necesario establecer la prioridad de estos componentes, los cuales suelen ser historias de usuario (Schwaber & Sutherland, 2011), de tal manera que los de mayor prioridad sean completados primero.

Incluso en equipos distribuidos, los aspectos más importantes para crear un equipo son la comunicación, visión compartida del producto, participación activa y las metas compartidas (Sutherland, Schoonheim, & Rijk, 2009). Los equipos colocados, pueden realizar actividades de interacción diarias, como la comunicación continua, resolución de problemas en equipo e incluso compartir temas de índole personal, las cuales crean relaciones personales más fuertes, lo que contribuye a construir un equipo más único y solidario. En el caso de un equipo distribuido, al no contar con la presencia física ni la interacción diaria, los miembros del equipo pueden tardar en conocer las debilidades y fortalezas de sus otros pares, lo cual puede ocasionar una difícil auto-organización del equipo (Moore, 2004).

Considerando lo expuesto anteriormente, cuando se trabaja con equipos distribuidos es necesario contar con una herramienta en la cual se pueda observar claramente la situación del proyecto, que ayude a la pronta adopción y adaptación del equipo con la metodología, y a aumentar la interacción de los equipos que se encuentren en diferentes lugares geográficos, reduciendo las diferencias geográficas y horarias con la co-presencia virtual.

Se debe tener en claro, que en ningún momento se puede dejar de lado la cultura ágil, es decir, que se debe contar con una herramienta que contribuya al acercamiento y a la comunicación constante de los miembros del equipo, al dinamismo de la pila del producto, a la contribución del propietario del producto y principalmente no debe añadirle una burocracia innecesaria al proceso.

1.2. OBJETIVO GENERAL

El objetivo del presente proyecto es implementar una herramienta informática de soporte a la gestión de proyectos ágiles de desarrollo de software; orientada a equipos distribuidos que utilicen Scrum.

1.3. OBJETIVOS ESPECÍFICOS

Los objetivos específicos del presente proyecto son:

- **Objetivo Específico 1:**
Identificar mecanismos de comunicación apropiados. Estos mecanismos son necesarios, ya que se necesita facilitar la interacción entre los miembros del equipo.
- **Objetivo Específico 2:**
Establecer las diversas opciones de visualización de la pila del producto (*product backlog*), considerando que deberán ser amigables con el usuario.
- **Objetivo Específico 3:**
Proveer un entorno colaborativo para la aplicación de técnicas de métodos ágiles, como gráficos burn-down y estimación de póker (*planning poker*).

1.4. RESULTADOS ESPERADOS

Considerando los objetivos específicos mencionados anteriormente, a continuación se muestran los resultados esperados del presente proyecto:

- **Resultado Esperado del Objetivo Específico 1:**
Listado de mecanismos de comunicación a implementar.
- **Resultado Esperado del Objetivo Específico 2:**
Diseño Final de Prototipos.
- **Resultado Esperado del Objetivo Específico 3:**
Listado y definición de técnicas de métodos ágiles que se implementarán.

1.5. MARCO CONCEPTUAL

Existen diferentes aspectos en el funcionamiento de un proceso de software, desde un proceso claramente definido y riguroso como el modelo cascada con un equipo

distribuido jerárquicamente hasta un proceso ágil con un entorno altamente iterativo e incremental (Rosenberg, Stephens, & Collins-Cope, 2005).

Las decisiones que se toman en el desarrollo colaborativo de software se pueden categorizar (Rosenberg, Stephens, & Collins-Cope, 2005):

- **Proceso lógico:** Consiste en la secuencia de pasos para llegar desde el punto inicial (requisito) hasta el punto final, un sistema de software trabajando.
- **Planificación y estrategia de progreso:** Esta etapa determina como el software es entregado al cliente.
- **Comunicación:** Se ocupa de las cuestiones de gestión de equipos con problemas de comunicación, tales como medios de comunicación, definición de roles, motivación del equipo, etc.
- **Prácticas de Trabajo:** Se encarga de los problemas de comunicación en un proyecto, tales como reuniones de equipo, revisión, etc.

El desarrollo de software incremental e iterativo se centra en la entrega de software en intervalos cortos y en gradualmente construir las funcionalidades en su totalidad. La planificación requiere un proceso iterativo de decisiones acerca de qué características o funcionalidades del proyecto se entregarán en cada iteración del proyecto (Rosenberg, Stephens, & Collins-Cope, 2005).

1.5.1. INICIOS DE LAS METODOLOGÍAS ÁGILES

En el transcurso de los años 80's, los procesos tradicionales de desarrollo de software comenzaron a ser criticados por ser lentos, burocráticos y excesivamente documentados. Experiencias con proyectos fuera de presupuesto y con retraso en las entregas comenzaron a preocupar a las persona acerca de la manera apropiada de gestionar proyectos de desarrollo de software. Los constantes cambios en los requerimientos, la creciente popularidad de internet, la creciente complejidad de los proyectos y los ambiciosos requisitos de los usuarios trajeron nuevos e innovadores enfoques sobre un proceso liviano para el desarrollo de software (Harley, 2004).

El término “desarrollo ágil de software” fue usado y publicado por primera vez en el Manifiesto Ágil (Agile Manifesto, 2002). Este término es usado como un paraguas

para una familia de desarrollo de software liviano emergente, tales como Programación Extrema (XP) y Scrum.

1.5.2. METODOLOGÍAS ÁGILES

Las metodologías ágiles se adaptan a las situaciones del mundo real con requisitos cambiantes, es una línea base que utiliza pequeños incrementos, iteraciones cortas, con enfoque conducido por la retroalimentación.

En el manifiesto Ágil se promueven cuatro valores principales de las metodologías Ágiles (Agile Manifesto, 2002).

- A los **individuos y su interacción**, por encima de los procesos y las herramientas.
- El **software que funciona**, por encima de la documentación exhaustiva.
- La **colaboración con el cliente**, por encima de la negociación contractual.
- La **respuesta al cambio**, por encima del seguimiento de un plan.

La gestión de proyectos ágiles enfatiza dos conceptos importantes. El primero es que el riesgo se minimiza centrándose en iteraciones cortas de resultados claramente definidos. La segunda es la comunicación directa con los socios en lugar de crear abundante documentación del proyecto. De esta manera el equipo se adapta rápidamente a lo impredecible y a los constantes cambios de requerimientos en los proyectos de desarrollo de software (Abrahamsson, Salo, Ronkainen, & Warsta, 2002).

1.5.3. METODOLOGÍA SCRUM

Scrum, un marco de trabajo ágil para la realización de proyectos complejos de desarrollo de software (Scrum Alliance), es un proceso iterativo e incremental basado en el trabajo en equipo; apoyándose en iteraciones cortas conocidas como "Sprints". Se enfoca en ayudar a líderes de proyectos a gestionar equipos de desarrolladores altamente calificados (Rosenberg, Stephens, & Collins-Cope, 2005).

Scrum se basa en tres componentes principales: los roles, los artefactos y las reuniones:

- **Roles**

En Scrum, se diferencian tres roles (Herranz, y otros, 2011):

- Scrum Mánager (*Scrum master*), es responsable de promulgar los valores y las prácticas de Scrum y la eliminación de impedimentos.
- Propietario del producto (*product owner*), es el responsable de comunicar la visión del producto al equipo de desarrollo. Representa al cliente y sus intereses en la definición y priorización de las funcionalidades del producto. Constituye el principal canal de comunicación entre el cliente y el equipo de desarrollo.
- Equipo de desarrollo (*Scrum team*), es el equipo multifuncional responsable de desarrollar el producto. Los roles dentro del equipo cambian dependiendo de las necesidades de cada iteración.

- **Artefactos**

Los artefactos de Scrum incluyen:

- Pila del producto (*product backlog*), es lo equivalente al catálogo de requisitos, con la diferencia que ésta evoluciona a lo largo del desarrollo. Frecuentemente los ítems de la pila del producto se detallan como historias de usuario (Schwaber & Sutherland, 2011).
- Pila del sprint (*sprint backlog*), lista de tareas a realizar por el equipo de desarrollo en cada sprint (Herranz y otros, 2011).
- Incremento (*increment*), producto potencialmente entregable que es desarrollado por el equipo en cada Sprint (Deemer y otros, 2010).
- *Gráfico Burn-Down*, representación gráfica del trabajo restante, y el progreso realizado, a nivel de sprint y del proyecto en general (Herranz, y otros, 2011).

- **Reuniones**

En las reuniones de Scrum se definen la pila del producto, los objetivos principales del proyecto y se realiza el seguimiento del progreso del equipo, además, permite a los miembros del equipo comprometerse con los demás y con el Scrum Master. Estas reuniones son:

- Planificación del sprint (*sprint Planning*), en esta reunión se define la pila del sprint, a partir de las prioridades en la pila del producto.

- *Daily Scrum*, reunión diaria donde el equipo revisa las tareas realizadas del sprint, las que se harán durante el día y las necesidades o impedimentos que se puedan presentar.
- Revisión del Sprint (*Sprint Review*), en esta reunión el equipo muestra el propietario del producto el incremento.
- Retrospectiva (*Retrospective*), reunión de 'mejora continua' donde el equipo analiza los problemas encontrados y los aspectos que se pueden mejorar.

1.5.4. SCRUM DISTRIBUIDO

Estudios recientes (VersionOne, 2010) indican que más del 50% de los proyectos ágiles tienen miembros de equipo distribuidos geográficamente. En estos equipos se pueden encontrar dos posibles tipos (Yaggahavita, 2011):

- Distribuidos con horas de trabajo superpuestas
- Distribuidos sin horas de trabajo superpuestas

En el primer caso, compartir horario de trabajo ayuda al equipo a crear una sensación de un único equipo; mientras que en el segundo caso, el equipo tiene que realizar un mayor esfuerzo para establecer comunicación constante, como tener reuniones fuera del horario de trabajo.

1.5.4.1. Equipos Distribuidos con horas de trabajo superpuestas

En este caso, los miembros del equipo tienen por lo menos algunas horas en un día normal de trabajo, en las cuales puedan interactuar. Por ejemplo, si el equipo de desarrollo consta de ocho miembros, y cinco se encuentran en Inglaterra y los otros tres en India, solo tienen 3:30 horas de trabajo superpuestas; tal como se muestra en la Ilustración 1.1. Las reuniones de revisión deben ser programadas durante estas horas (Woodward, Surdek, & Ganis, 2010).

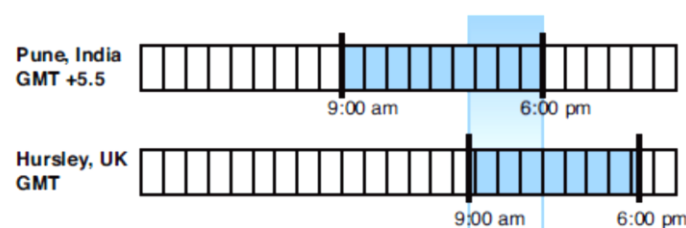


ILUSTRACIÓN 1.1: EQUIPOS DISTRIBUIDOS CON HORAS DE TRABAJO SUPERPUESTAS (WOODWARD, SURDEK, & GANIS, 2010)

Uno de los desafíos más comunes es estar seguros que las reuniones se realizan eficientemente, no es lo mismo comunicarse con una persona personalmente, que remotamente, en este caso, no se aprecia el lenguaje corporal, ni se puede asegurar un completo compromiso en la actividad. Las diferencias culturales y de lenguaje también son desafíos a considerar; en un escenario extremo, los miembros del equipo no comparten el mismo lenguaje nativo, como por ejemplo, si el equipo se encuentra dividido entre Alemania, España y Estados Unidos (Woodward, Surdek, & Ganis, 2010).

1.5.4.2. Equipos Distribuidos sin horas de trabajo superpuestas

En este caso, los equipos no tienen oportunidad de interactuar durante un día normal de trabajo. Por ejemplo, como se puede apreciar en la Ilustración 1.2, el equipo puede estar distribuido entre California, Texas y China; parte del equipo en Estados Unidos, no comparte horas de trabajo comunes con la parte del equipo en China.

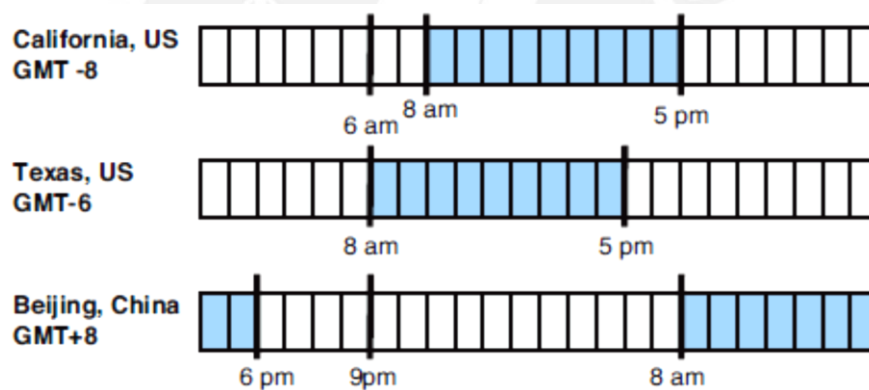


ILUSTRACIÓN 1.2: EQUIPOS DISTRIBUIDOS SIN HORAS DE TRABAJO SUPERPUESTAS (WOODWARD, SURDEK, & GANIS, 2010)

Para estos equipos, cada una de las reuniones de Scrum es un reto, incluso si logran programar reuniones fuera del horario de trabajo, la comunicación durante el Sprint no se realizará de manera directa, si no de manera asíncrona, con herramientas como correo electrónico (Woodward, Surdek, & Ganis, 2010).

1.5.5. PRÁCTICAS DE SCRUM EN ENTORNOS DISTRIBUIDOS

En un entorno distribuido, todas las prácticas estándar de Scrum – los roles, reuniones y artefactos – están presentes. Sin embargo, puede ser necesario ajustar la forma en la que dichas prácticas se aplican, para superar diferencias de

ubicación geográfica y de zona horaria. A continuación se muestra brevemente como las principales prácticas de Scrum se ven afectadas en entornos distribuidos (Sutherland, Schoonheim , & Rijk, 2009):

- La duración del Sprint puede afectar la comunicación y el adecuado entendimiento de los requisitos.
- La toma de decisiones del equipo podría producirse en ausencia del propietario del producto, cuando este se encuentra en una locación geográfica diferente al del equipo, con solo un poco de superposición de tiempo.
- Los deberes del Scrum Master de entrenar al equipo, ayudando a eliminar los obstáculos y la protección del equipo de la interrupción van a estar esencialmente ausentes si el Scrum Master está ubicado muy lejos del equipo.
- Cuando el equipo está dividido en múltiples locaciones, el nivel de coordinación, cooperación y el trabajo en equipo, que es necesario para entregar software cada 4 semanas o menos, es incluso más demandante. Si los equipos no están lo suficientemente unidos, pueden surgir descoordinaciones y falta de comunicación, lo cual puede incrementar malos entendidos.
- Si una parte del equipo, *onshore*, trabaja directamente con el propietario del producto, él probablemente mantendrá una comunicación más estrecha con esta parte del equipo, dejando en desventaja de información al resto del equipo *offshore*.

1.6. ESTADO DEL ARTE

En esta sección se presentarán las principales herramientas que existen actualmente para la gestión de proyectos ágiles, específicamente aquellos que utilicen Scrum como metodología base; luego se procede a la comparación y el análisis de las principales funcionalidades.

1.6.1. HERRAMIENTAS EXISTENTES

A continuación se muestran las principales funcionalidades ofrecidas por herramientas existentes para la gestión de proyectos ágiles.

1.6.1.1. Version One

Página web: <http://www.versionone.com>

Principales Funcionalidades:

- Diferentes niveles de planificación: Product, Release y Sprint. En el nivel *Product Planning* es donde se encuentra el product backlog.
- Los User Stories, además de contar con estimado y prioridad (*low, medium, high*), también se puede registrar información más detallada como propietario o complejidad.
- En *Release planning*, se pueden crear subproyectos, estableciendo el tiempo de inicio y fin del mismo. El *Sprint planning* funciona de manera similar con la creación de los *sprints* y la creación de tareas (*task*).
- Soporta múltiples equipos y múltiples proyectos.

1.6.1.2. ScrumDo

Página web: <http://www.scrumDo.com>

Principales Funcionalidades:

- El estado de los *Stories* es simple de cambiar (*ToDo, Doing, Needs Review* y *Done*) y fácil de visualizar.
- Manejo simple de prioridades, click y arrastre.
- Miembros del equipo pueden ser notificados del cambio en sus *stories*.
- Cuenta con predicciones, tomando en cuenta la velocidad con la que avanza el equipo y los puntos restantes.

1.6.1.3. Scrumdesk

Página web: <http://www.scrumdesk.com>

Principales Funcionalidades:

- Soporta múltiples equipos, orientado a equipos distribuidos.
- Proporciona diferentes vistas del Backlog, haciendo menos engorrosa la vista si es que se cuenta con una cantidad grande de historias de usuarios (*user stories*), o proporcionando una vista estilo pizarra de tareas (*taskboard*) para una vista más detallada.
- Uso de *planning poker* para la planificación del proyecto.

- La pizarra de tareas del sprint tiene, a diferencia de las tres columnas tradicionales (ToDo, In Progress, Done), cuatro columnas (ToDo, Checked out, Solved, Completed).
- Los impedimentos que se presenten en el Sprint pueden ser anotados en cualquier momento, cada uno está conectado con una historia de usuario.
- *Retrospective*, funciona como un pool en donde los miembros del equipo presentan ideas y estas pueden someterse a votación.

1.6.1.4. ScrumWorks

Página web: <http://www.collab.net/products/scrumworks/>

Principales Funcionalidades:

- Manejo dinámico de la Pila del producto (*Product backlog*), soporta estimación de esfuerzo de cada requerimiento identificado
- Aplicación web y de escritorio.
- Seguimiento de impedimento, según fechas y responsabilidades.
- Reportes personalizables, por defecto cuenta con *sprint* y *release burndown*.
- Diferenciación entre tiempos de equipos y tiempos individuales, permitiendo el soporte para el control de tiempos en programación en pares.

1.6.1.5. IceScrum

Página web: <http://www.icescrum.org/en/>

Principales Funcionalidades:

- Un proyecto puede tener múltiples productos, cada uno de estos productos tiene su propio *backlog* y un *roadmap*. Cada *backlog* contiene *user stories*, defectos y *technical stories*.
- Cada sprint tiene su propio *backlog*. Los *user stories* contienen además la definición de tareas y pruebas de aceptación.
- Incluye *task board* para cada sprint, estos *tasks* pueden ser redistribuidos.
- Ofrece Planning Poker para la estimación y para establecer prioridades en el *backlog*.
- Permite crear nuevos roles personalizados, además de los predefinidos, *Product Owner*, *Scrum Master* y *Team Member*.

1.6.1.6. Greenhopper y Jira

Página web: <http://www.atlassian.com/software/greenhopper/>

Principales Funcionalidades:

- Permite utilizar diferentes boards para lograr diferentes vistas; *planning board*, corresponde a la pizarra de mantenimiento del backlog; *Task board*, permite mantener un seguimiento de cada tarea.
- Generación de diferentes gráficos como Burn-Down, Burn-Up, gráfico de horas y de velocidad.
- Seguimiento de versiones entregadas, *releases*.
- Generación de notas de versiones y gráfico de versiones.
- Manejo de múltiples equipos.
- Seguimiento de tiempo, con el uso de seguimiento de Jira incorporado.

1.6.2. DISCUSIÓN SOBRE LOS RESULTADOS DE LA REVISIÓN DEL ESTADO DEL ARTE

Como se puede observar en la Tabla 1.1, las herramientas más completas son *VersionOne*, *GreenHopper*, *ScrumDesk* y *ScrumWork*; y son las que implican mayor inversión económica. *ScrumDo* y *IceScrum* sin las funcionalidades de múltiples entregables y múltiples proyectos no permiten gestionar un proyecto a gran escala, su uso es solo recomendable cuando los proyectos son pequeños.

VersionOne, es la única de las herramientas comparadas que no está dedicada exclusivamente a Scrum, si bien las metodologías ágiles se rigen bajo los mismos principios ágiles, en algunas ocasiones es necesario contar con artefactos específicos de cada metodología. Además, se debe considerar que las organizaciones dedicadas a desarrollo ágil, no suelen hacer inversiones en herramientas de gestión ágil, les basta el uso de pizarras, y *VersionOne* es una de las herramientas más caras en el mercado.

ScrumDo y *ScrumDesk*, son herramientas que se enfocan en el despliegue visual de la pila del producto, y permiten la creación de varios equipos, que trabajen paralelamente. *ScrumDo*, cuenta con la funcionalidad de chat, pero está bloqueada, a menos que se adquiera un plan de mayor costo del que se menciona en el cuadro comparativo anteriormente mostrado (Tabla 1.1)

ScrumWork, no permite crear roles de usuario personalizables, esto limita la posibilidad de utilizar Scrum como marco de trabajo adaptativo.

Como última herramienta a comparar se presenta *GreenHopper*, el cuál no es una herramienta de gestión propiamente dicha, es una extensión de una muy conocida herramienta de seguimiento de errores y planificación de proyectos Jira; juntas ofrecen un soporte para gestión de proyectos ágiles de desarrollo de software. A pesar de ser una extensión muy potente, *GreenHopper* no cuenta con soporte a las

TABLA 1.1: CUADRO COMPARATIVO DE LAS PRINCIPALES HERRAMIENTAS EXISTENTES (THE LINUX FOUNDATION)

	Version One	ScrumDo	ScrumDesk	ScrumWork	IceScrum	GreenHopper & Jira
Versión revisada	Winter'12	-	5.2.10	5.0	R4#4	-
Costo	Ultimate edition: \$39 mensuales por usuario	Silver Plan: \$39.95 mensuales	\$15 mensuales por usuario	Pro - Hosted: \$25 mensuales	0	10 usuarios por \$20 mensuales
Establecimiento de prioridades en el <i>backlog</i>	Ranking	Ranking	Ranking. MoSCow	Ranking	Ranking	Ranking
Puntos en <i>Stories</i>	Si	Si	Si	Si	Si	Si
Gráficos Burn Down	Si	Si	Si	Si	Si	Si
Jerarquización de <i>backlog (Epics)</i>	Si	Si	Si	Si	Si	Si
Múltiples entregas	Si	No	Si	Si	No	Si

	Version One	ScrumDo	ScrumDesk	ScrumWork	IceScrum	GreenHopper & Jira
Múltiples proyectos	Si	Si	Si	Si	No	Si
Roles de usuario	PO, SM, Team Member	PO, SM, Team Member	PO, SM, Team Member, roles personalizados.	PO, SM, Team Member	PO, SM, Team Member, Stakeholder, roles personalizados.	Project Lead, Team, roles y permisos personalizados
Reportes	Si	No	Si	Si	No	Si
Mecanismos de comunicación	No	Si, cuenta con un Chat	Si, cuenta con un Chat	No	No	No
Soporte a las reuniones de Scrum	No	No	No	No	No	No

reuniones de Scrm, ni promueve la colaboración de los miembros de equipos mediante comunicación en línea; por otro lado es importante tomar en cuenta que la posibilidad de contar con diversas pizarras de visualización de la pila del producto y de su desarrollo, otorga un dinamismo adicional al equipo de trabajo en cuanto al manejo y la gestión del proyecto; atributo con el que no cuentan las herramientas previamente mencionadas.

Finalmente, se debe agregar que la herramienta propuesta, incluirá mecanismos de interacción, que no presentan las herramientas mencionadas anteriormente. Los equipos que utilizan Scrum distribuido y hacen uso de estas herramientas, usualmente, tienen que usar herramientas adicionales que les ayuden a mantener una comunicación constante como Skype o Gtalk. Con la herramienta propuesta, estará integrada la visualización del estado del proyecto, de los artefactos de Scrum y se podrá llevar a cabo de una manera integrada las reuniones de Scrum y la comunicación de todo el equipo.

1.6.3. CASOS DE ÉXITO

A continuación se presentan dos casos de éxito, en donde se ha llevado a la práctica apropiadamente Scrum distribuido, en el primer caso se muestra en IBM, una empresa que ha logrado incorporar ágil, dentro de su cultura organizacional; en el segundo caso se muestra, el desarrollo de software para la línea de ferrocarriles holandeses utilizando metodologías ágiles y equipos distribuidos.

1.6.3.1. IBM

IBM es un organización muy grande y diversa, con empleados en más de 200 países (Woodward, Surdek, & Ganis, 2010). Cuenta con ingenieros de software de diferentes culturas trabajando en casi todos los husos horarios.

IBM Software Group tiene trabajando en cada producto un promedio de 120 miembros por equipo. Los equipos más pequeños cuentan con 50 miembros, mientras que los más grandes pueden llegar a contar con 600 personas. Un ejemplo de un proyecto mediano, es IBM WebShere Portal V6.0, que cuenta con 338 miembros en el equipo de desarrollo, la distribución se puede observar a detalle en la Ilustración 1.3.

Durante los años 80, al igual que otras compañías de desarrollo, en IBM se promovió el uso del modelo de desarrollo en cascada; fue en los 90 cuando la compañía se enfocó en migrar a métodos de desarrollo evolutivo; se comenzó a utilizar una versión adaptada de RUP (Rational Unified Process). En el año 2003,

un grupo de cuatro trabajadores, empezaron una comunidad ágil interna, Agile@IBM, haciendo uso de wikis y de redes sociales (Woodward, Surdek, & Ganis, 2010).

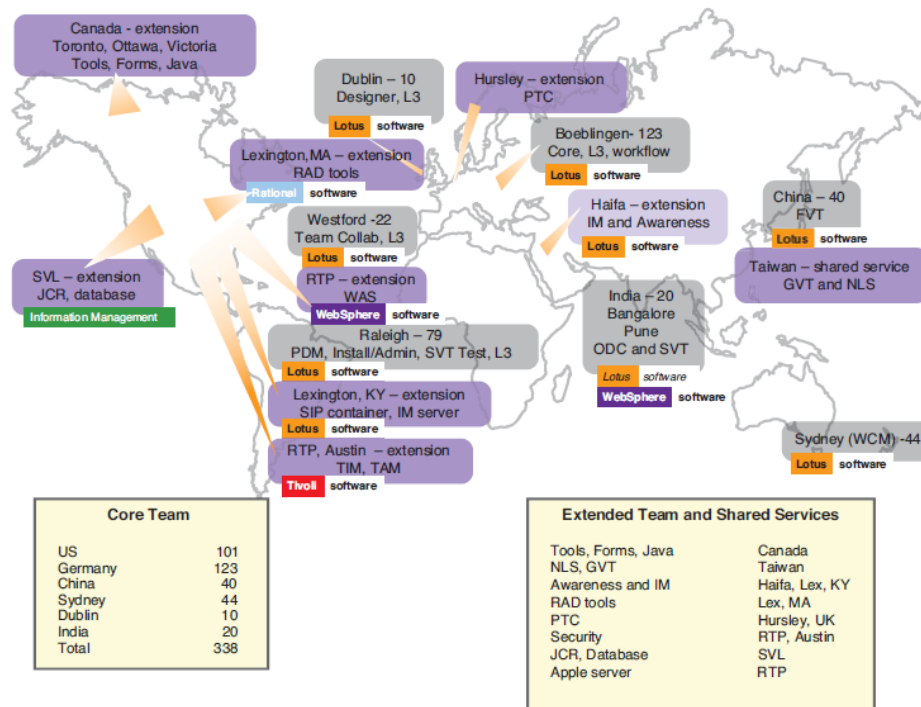


ILUSTRACIÓN 1.3: DESARROLLO DE PORTAL V6.0 (WOODWARD, SURDEK, & GANIS, 2010)

En Abril del 2006, IBM Software Group compartió por primera vez experiencias y conocimientos en una conferencia, Technical Leadership Exchange Conference; a partir de este momento la adopción de metodologías ágiles creció considerablemente dentro de la compañía, en enero del 2007 solo 5 de 550 proyectos eran ágiles, para el 2008, el número de proyectos incrementó a 220 (Woodward, Surdek, & Ganis, 2010).

1.6.3.2. Dutch Airways (Ferrocarriles Holandeses)

- *Antecedentes*

Dutch Airways, ofrece servicio de transporte a 1.2 millones de pasajeros diariamente; en el año 2005, se necesitó un sistema de información que provea a los pasajeros información de los viajes más precisa. El sistema a implementar se

llamaría PUB. En un primer momento, se contrató a una empresa para el desarrollo que utilizaría un modelo en cascada tradicional para la ejecución del proyecto; el contrato fue cancelado, ya que la empresa luego de tres años no pudo entregar un sistema funcionando. Luego de este fracaso, el cliente optó por contratar a una empresa que utilizara metodologías ágiles en el desarrollo (Mulder & Van Vliet, 2008).

- *Solución*

En primer lugar se realizó el kick-off del proyecto para que estuviera listo todo lo necesario antes del primer Sprint, este kick-off tuvo una duración de 3 semanas y contó con la participación del jefe de proyecto, un arquitecto y un Scrum Master.

Luego del kick-off, se debería decidir quién sería el propietario del producto. No se contaba con una persona que tuviera los conocimientos necesarios del negocio, ni que hubiera estado involucrada previamente en el uso de Scrum, por esta razón se decidió contar con dos analistas de negocio que trabajaban para el cliente y contribuyeron a la toma de requisitos; ellos contaban con la experiencia del negocio necesaria.

Los nuevos propietarios del producto, habían trabajado en la toma de requisitos, pero estos no estaban adecuados para su uso en Scrum, por este motivo, se tuvo que trabajar durante un par de semanas con el Scrum Master en el entrenamiento de los propietarios del producto y en el uso de historias de usuario.

Al inicio, el equipo contaba con cinco desarrolladores; en el primer Sprint dos desarrolladores indios, se unieron al grupo y se familiarizaron con la aplicación y con el resto de equipo del desarrollo. Luego de seis semanas, los desarrolladores indios regresaron a su país, a partir de esto se crearon dos equipos de Scrum, uno en India y otro en Holanda. Luego el equipo de desarrollo en India escaló a seis miembros; esto permitió mantener una velocidad alta y calidad en el desarrollo.

Se logró conservar una comunicación constante entre los miembros de los equipos, utilizando Skype constantemente, contaban con cámaras web, micrófonos, audífonos y pantallas de gran tamaño, de esta manera se podían realizar reuniones con todo el equipo.

Para realizar el seguimiento del proyecto, se utilizó ScrumWorks, ya que para equipos distribuidos resulta mucho más beneficioso el uso de una herramienta en

lugar de una pizarra; aquí también se pudo discutir sobre la pila del producto (*product backlog*) con los propietarios.

El usuario requería de documentación, la cual tenía ciertas restricciones, tenía que seguir el estándar MIL-STD-498 y debía estar escrita en holandés; los miembros del equipo holandés no estaban familiarizados con este estándar, motivo por el cual se decidió contratar un técnico especializado (Mulder & Van Vliet, 2008).

- *Logros*

El cliente se mostró satisfecho con el producto final. Además, el cliente contrató una empresa para que auditara el software, cuyas conclusiones fueron (Mulder & Van Vliet, 2008):

- El mantenimiento del producto es muy bueno.
- La calidad del código fuente es muy alta.

1.7. DESCRIPCIÓN Y JUSTIFICACIÓN DE LA SOLUCIÓN

A continuación se muestra el alcance del proyecto, limitaciones y la justificación.

1.7.1. ALCANCE

El proyecto se relaciona con las empresas dedicadas al desarrollo de software, específicamente aquellas que utilicen metodologías ágiles de desarrollo, y que tengan equipos distribuidos en diferentes locaciones geográficas.

Se implementará una herramienta automatizada de gestión de proyectos ágiles de desarrollo de software, que satisfaga las necesidades de los usuarios finales de una manera sencilla y conservando la cultura ágil.

La herramienta desarrollada permitirá la gestión de proyectos ágiles Scrum, teniendo en cuenta los roles y los artefactos involucrados en este tipo de proyecto. Se espera lograr una mayor interacción de los miembros del equipo y con los usuarios finales, a través de los mecanismos de interacción implementados en la herramienta, como el chat o la videoconferencia. Estos mecanismos, además se podrán usar en las reuniones de Scrum.

La herramienta permitirá almacenar la pila del producto, a partir del registro de historias de usuarios, registrar la priorización de elementos de la pila del producto y

seleccionar aquellos que formarán parte del siguiente sprint. Los miembros del equipo podrán registrar los tiempos estimados de las historias de usuario y registrar los cambios de estado de los mismos; de esta manera se calculará la velocidad estimada y se contrastará con la velocidad real con la que se realizaron las actividades detalladas en la historia de usuario; este resultado podrá ser visualizado mediante gráficos Burn Down.

Se pondrá especial cuidado con el despliegue visual, pues es necesario crear un ambiente ágil que simule las técnicas utilizadas en el mundo real. Para cumplir esto último, se contará con diferentes formatos de despliegue de los artefactos, para otorgarle al usuario final, diversas maneras de visualización de los mismos.

1.7.2. LIMITACIONES

Las limitaciones del presente proyecto son:

- Si bien, los usuarios podrán crear y compartir proyectos, no se espera poder crear dependencias entre ellos.
- Los estados de las historias de usuarios, tendrán que ser actualizados manualmente por los usuarios; pues no se espera implementar ninguna extensión adicional, para ningún IDE existente en el mercado, la cual determine el progreso del proceso de desarrollo de software.

1.7.3. JUSTIFICACIÓN

A continuación se presentan las razones por las cuales se justifica el desarrollo del proyecto:

La solución propuesta propone el uso de una herramienta automatizada, para la gestión de proyectos Scrum, especialmente aquellos en los que el equipo de trabajo se encuentre distribuido en diferentes lugares geográficos. Se plantea utilizar mecanismos de interacción como el chat y los foros, para que los miembros del equipo se vean beneficiados con la comunicación constante, reduciendo de esta manera las diferencias geográficas.

Los proyectos de Scrum, en equipos colocados, no presentan problemas con el uso de herramientas clásicas como pizarras y post-it, para llevar un seguimiento del proyecto. Se plantea usar una metáfora de esta forma de trabajo, y usar un

despliegue visual que sea familiar, tanto para el usuario experto en la metodología como para aquel que recién se inicia en ella.

Al realizar una investigación de la literatura, no solo de la metodología original, sino además de las buenas prácticas recomendadas para Scrum distribuido, se presentan diversas formas en las que el usuario pueda usar la metodología, haciendo que el uso de Scrum sea dinámico y adaptable a los requerimientos del proyecto. Es por esta razón, que se plantea que la herramienta a implementar sea altamente dinámica y adaptable a cualquier tipo de proyecto, y sea independiente de la naturaleza del mismo.

Además, se propone que la herramienta permita almacenar diversos documentos que puedan requerir en el proyecto. De esta manera el equipo, sería beneficiado con un repositorio común, en el que todos puedan observar la situación del proyecto y acceder a cualquier tipo de documento compartido.



2. PLANIFICACIÓN DEL PROYECTO

En el presente capítulo se muestra la metodología elegida para el desarrollo del proyecto, así como la planificación del mismo.

2.1. PROCESOS DE GESTIÓN E INGENIERÍA

El presente proyecto, se presenta de manera individual, y solo cuenta con una persona para el desarrollo del proyecto, se consideró necesario contar con un marco de trabajo que promueva la auto-gestión y la adaptabilidad. Se debe tomar en consideración, que si bien se cuenta con un alcance definido del proyecto, la planificación y la especificación de los requisitos puede ir cambiando a medida que se avanza en la implementación de la herramienta . Por otro lado, considerando que la herramienta a implementar se basa en Scrum, al utilizarlo también como marco de trabajo del proyecto familiariza al tesista con el mismo, a nivel teórico y práctico. Por las razones previamente expuestas, se ha decidido optar por metodologías ágiles para la gestión del proyecto.

Scrum, como marco de trabajo ágil, pone mucho énfasis en la comunicación e interacción entre los miembros del equipo. Como se mencionó anteriormente, a diferencia de los proyectos tradicionales, el presente proyecto se realiza de manera personal, es por esta razón que se necesita usar una adaptación de este marco de trabajo. Scrum no está enfocado en la documentación exhaustiva, por ello no propone un marco referente en cuanto a la documentación, por el contrario ofrece la libertad de extender el marco de trabajo para proveer la documentación necesaria, a través de entregables que generen valor al cliente. Para el presente proyecto, la documentación necesaria es establecida según los lineamientos del proyecto de fin de carrera y por los resultados esperados mencionados anteriormente (Sección 1.4).

Los roles que se definen en Scrum son (Scrum Alliance):

- El propietario del producto; para este proyecto no se cuenta con un cliente específico con el cuál realizar la toma de requisitos, pues se realiza en un entorno académico. El propietario del producto debe interactuar constantemente con el equipo de desarrollo y debe tener conocimiento pleno de las funcionalidades que deberá tener el producto final; además es indispensable contar con una persona fuera del equipo de trabajo que verifique que los entregables al final de cada iteración, son realmente un producto potencialmente entregable.
Dado que se cuenta con un escenario particular, las actividades que deberían ser realizadas por el propietario del producto fueron adaptadas para la realización del proyecto; las actividades involucradas directamente con el desarrollo del proyecto, como la priorización de historias de usuario, fueron realizadas por el tesista. Además, como se mencionó anteriormente, es necesaria la presencia de una persona externa al equipo, para cumplir con esto se contó con el apoyo del asesor del proyecto de fin de carrera.
- El Equipo, es el encargado del desarrollo del producto. En el presente proyecto, solo se contará con una persona para el desarrollo.
- El Scrum Mánager (*Scrum Master*), como no es recomendable que el Scrum Master sea el mismo que el propietario del producto; pero algunas veces este rol puede ser tomado por algún miembro del equipo de desarrollo. Considerando la naturaleza del proyecto, este rol será asumido por la persona, miembro del equipo de desarrollo.

Los artefactos de Scrum, serán utilizados de la siguiente manera:

- El kick-off, etapa inicial de la planificación del proyecto. En primer lugar, se procede con un estudio sistemático de la situación actual del problema, tanto de la literatura existente como de las herramientas actuales de gestión de proyectos ágiles. Al finalizar esta etapa, se tiene como resultado el documento de estado del arte.
- Pila del producto: la pila del producto está compuesto por ítems, los cuáles varían a lo largo de la realización del proyecto, se añaden o se eliminan algunos según se considere necesario. Al término del proyecto se obtiene la pila del producto final, con los ítems debidamente priorizados; en el presente proyecto los ítems de la pila del proyecto serán historias de usuario.
- Pila del sprint (*Sprint backlog*), es una parte de la pila del producto. Para asegurar, las entregas parciales del producto, el ciclo de desarrollo se divide en iteraciones, conocidas como sprints. El contenido de cada pila de sprint es determinado realizando una estimación inicial de las prioridades de los ítems de la pila, la pila del producto inicial se distribuye en las pilas de los sprints.
- Incremento, es el producto potencialmente entregable, uno de los objetivos de scrum, es obtener al final de cada sprint un producto potencialmente entregable que el usuario pueda utilizar.

Scrum (Suttherland,Jeff: 210),propone reuniones, que permiten que el equipo de trabajo comparta su conocimiento, así como establecer relaciones de equipo, para promover así una auto-organización y discutir acerca de los avances del proyecto. Las reuniones de Scrum, serán llevadas a cabo en el presente proyecto de la siguiente manera:

- La reunión (kick-off meeting) reunión o reuniones iniciales del proyecto, en las cuales se determinan los roles del proyecto, se elabora la pila del producto inicial y el propietario del producto establece prioridades a las funcionalidades descritas en las historias de usuario. En este caso particular, las reuniones se realizarán entre el tesista y el asesor del curso.
- Planificación del sprint (*Sprint Planning*), con base en la pila del producto, se seleccionan las actividades que el equipo se compromete a

realizar durante el sprint. Se determinan además las tareas necesarias para el cumplimiento de todas las funcionalidades especificadas en las historias de usuario.

- Revisión del sprint (*Sprint Review*), es una reunión corta en la cual se presenta el incremento del sprint; como en el caso de las reuniones iniciales, las revisiones del sprint se realizaron entre el tesista y el asesor del curso.
- Retrospectiva, se realiza al final de la revisión del sprint, se analiza lo malo y lo bueno del sprint, y se establecen actividades de mejora para el siguiente sprint

A continuación se presenta la Tabla 2.1 con las actividades a realizar para el cumplimiento de los resultados esperados.

TABLA 2.1: CUADRO DE ACTIVIDADES

Objetivos Específicos	Actividades	Resultados Esperados
Identificar mecanismos de interacción apropiados.	Planificación y estimación de las historias de usuario en el tercer sprint.	Listado de mecanismos de interacción a utilizar.
Determinar el despliegue visual más óptimo para la Pila del producto.	Para el diseño de prototipos se necesita: <ul style="list-style-type: none"> - Analizar los despliegues de la pila del producto más usados, tanto en las herramientas existentes como en la realidad, y determinar cuál o cuáles son los más óptimos a implementar. - Establecer una paleta de colores, que sea amigable con el usuario. - Diseñar los prototipos y validarlos. De ser necesario se harán correcciones del diseño. 	Diseño Final de Prototipos.

<p>Proveer un entorno colaborativo para la aplicación de técnicas de métodos ágiles, como gráficos burn-down y estimación de póker.</p>	<ul style="list-style-type: none"> - Determinar el flujo de técnicas de métodos ágiles. - Identificar los elementos con los que se relacionan y determinar el cambio que puede significar en ellos. -Describir las condiciones en las que será posible utilizar cada una de estas técnicas. 	<p>Listado y definición de técnicas de métodos ágiles que se implementarán.</p>
---	--	---

2.1.1. VIABILIDAD

A continuación se muestra la viabilidad temporal, económica y técnica de la solución propuesta.

2.1.1.1. Viabilidad temporal

Para el presente proyecto de desarrollo, se contó con cuatro meses de elaboración del anteproyecto, y con un tiempo límite de cinco meses para la implementación de la herramienta, en el plan de proyecto detallado en la siguiente sección, se estima una duración de sprint, de 27 días laborales, de dos horas y media cada uno, es decir, cada Sprint tendrá una duración de casi 70 horas. Durante dos ciclos académicos se contó con el tiempo suficiente para realizar lo esperado al final de cada Sprint. Es por esta razón, el proyecto se consideró viable temporalmente.

2.1.1.2. Viabilidad económica

El presente proyecto utilizará herramientas de software libre para su realización. Será desarrollado en PHP, utilizando NetBeans, como IDE. Estas herramientas no generan un costo adicional para el desarrollo del producto final.

En el caso del equipamiento necesario, se dispone de una computadora personal portátil, con el software necesario ya instalado. Como se cuenta con todas las herramientas, el único costo que supone el proyecto son las horas trabajadas. En Scrum la unidad de medida no son horas, sino días; por esta razón se estimará el costo en función de los días trabajados y de gastos adicionales; como se indicó en el punto anterior, cada día corresponde a dos horas y media de trabajo. A continuación, en la Tabla 2.2, se presentan costos estimados para el proceso de desarrollo de la herramienta propuesta.

TABLA 2.2: TABLA DE COSTOS ESTIMADOS DEL PROYECTO

Fases	Días	Costo/ día	Costo (S/.)
• Kick-Off	16	S/. 60	S/. 960.00
• Sprint 1	27	S/. 80	S/. 2,160.00
• Sprint 2	27	S/. 80	S/. 2,160.00
• Sprint 3	27	S/. 80	S/. 2,160.00
• Sprint 4	27	S/. 80	S/. 2,160.00
Sub-Total	124		S/. 9,600.00

Gasts Adicionales	Costo (S/.)
• Luz	S/. 1,000.00
• Internet	S/. 500.00
• Contingencia	S/. 1,000.00
Sub-Total	S/. 2,500.00

TOTAL	S/. 12,100.00
--------------	----------------------

2.1.1.3. Viabilidad técnica

La solución planteada es realizar el desarrollo web de una herramienta de soporte a la gestión de proyectos ágiles, con el uso de CakePHP como framework para el desarrollo de aplicaciones web. NetBeans como IDE, permite realizar el desarrollo adecuado de todas las funcionalidades que se desean implementar.

Se hará uso de tecnología existente para el desarrollo de la solución propuesta, por esta razón el proyecto se considera técnicamente viable.

2.2. PLAN DE PROYECTO

A continuación se muestra la estructura de descomposición del trabajo del proyecto:



ILUSTRACIÓN 2.1: ESTRUCTURA DE DIVISIÓN DEL TRABAJO

Como se puede observar, en la Ilustración 2.1, los sprints cuentan con cuatro actividades principales. Por la naturaleza de la metodología a utilizar, no se puede saber de antemano las actividades a realizarse en cada de sprint; estas se determinan en cada una de los planificaciones de sprint. El detalle de la planificación de cada sprint, se muestra a medida

En la Ilustración 2.2, se presenta un Diagrama de Gantt, en el cual se muestra el espacio temporal del desarrollo del proyecto. Si bien no se puede saber que actividades se realizará en cada sprint, se puede delimitar la duración de los mismos, y así estimar el tiempo necesario para la culminación apropiada del desarrollo de la herramienta propuesta.

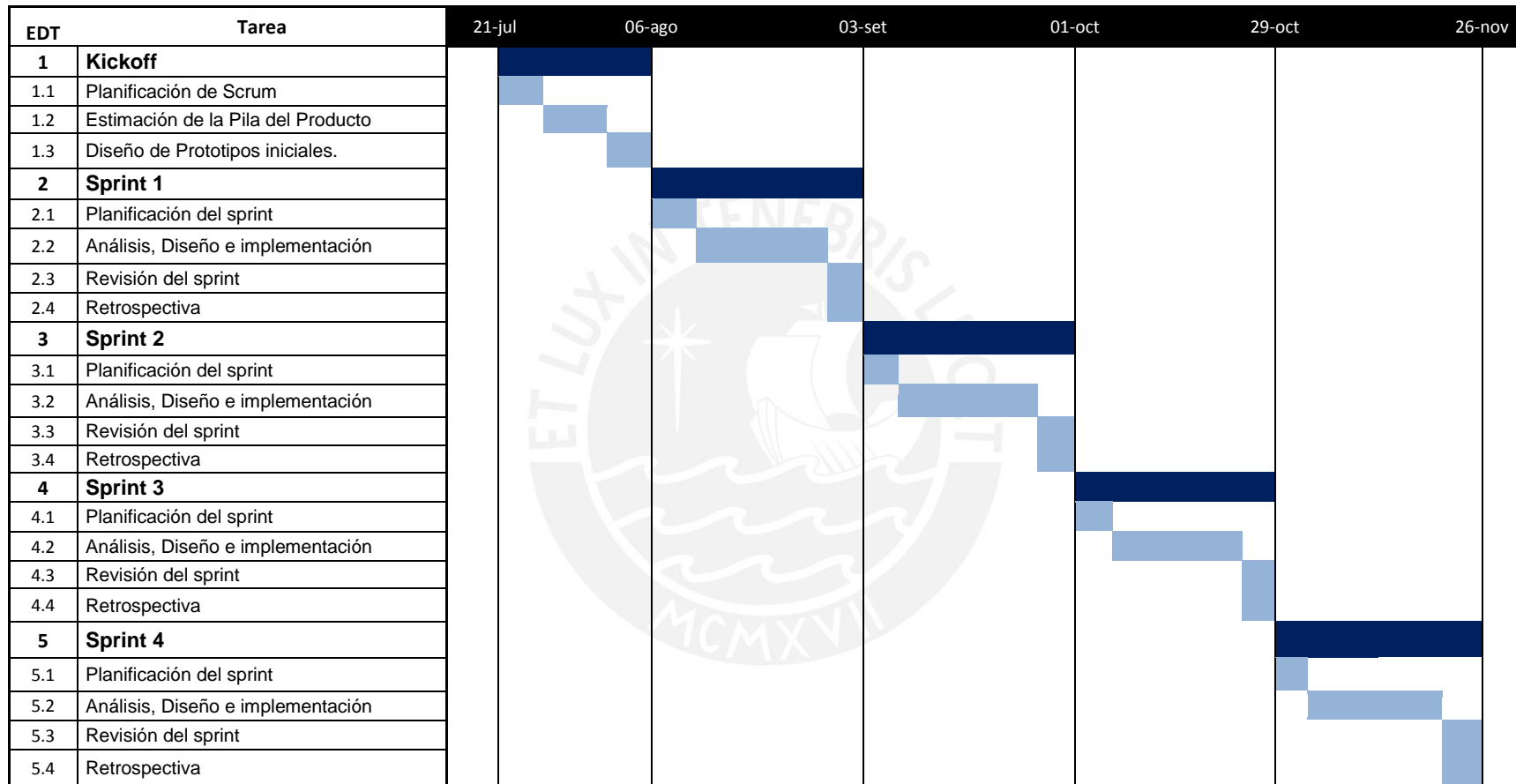


ILUSTRACIÓN 2.2: DIAGRAMA DE GANTT

3. REQUISITOS

Considerando que para el proceso de desarrollo de la herramienta se utilizará el marco de trabajo Scrum, las funcionalidades que requeridas en la herramienta a implementada se muestran a manera de ítems de la pila del producto; como se indicó en el anteriormente, los ítems se mostrarán en forma de historias de usuario.

3.1. ACTORES

En la Ilustración 3.1 se muestran los actores identificados



ILUSTRACIÓN 3.1: DIAGRAMA DE ACTORES

Como se puede observar, los actores corresponden directamente a los roles que se utilizan en Scrum.

3.2. PILA DEL PRODUCTO

A continuación se muestra la pila del producto de la herramienta desarrollada. En el Anexo B y Anexo C, se muestran los mecanismos de colaboración y las técnicas de métodos ágiles que se implementaron.

TABLA 3.1: PILA DEL PRODUCTO

Historias de Usuario				
#Historia	Como un ...	Quiero ...	Para ...	Prioridad (MoSCoW)
1	Usuario	Iniciar sesión	Ingresar al sistema	M
2	Usuario	Enviar invitaciones	Invitar a los miembros del equipo a unirse al proyecto	S
3	Usuario	Administrar un Proyecto	Iniciar nuevos proyectos ágiles	M
4	Usuario	Compartir proyectos	Añadir miembros al equipo	M
5	Miembro del equipo	Visualizar todas las historias de usuario de un proyecto		M
6	Usuario	Comunicarme con otro usuario mediante la herramienta	Compartir dudas o avances sobre el proyecto	S

Historias de Usuario

#Historia	Como un ...	Quiero ...	Para ...	Prioridad (MoSCoW)
7	Usuario	Crear un Sprint	Crear las iteraciones que va a tener el proyecto	M
8	Miembro del equipo	Crear Historia de Usuario	Agregar ítems a la pila del producto	M
9	Scrum Mánager	Agregar roles	Agregar roles personalizados, además de los roles por defecto.	S
10	Scrum Mánager	Administrar roles	Cambiar los roles de los miembros del equipo en diferentes momentos del proyecto	M
11	Scrum Mánager	Administrar sprints	Crear sprints en el proyecto, modificar el tiempo de duración de cada uno, etc.	M
12	Scrum Mánager	Programar y llevar a cabo las reuniones		C

Historias de Usuario

#Historia	Como un ...	Quiero ...	Para ...	Prioridad (MoScow)
13	Scrum Mánager	Programar y llevar a cabo la reunión de planificación del Sprint.		C
14	Scrum Mánager	Programar y llevar a cabo la reunión de planificación de la Pila del producto.		C
15	Scrum Mánager	Programar y llevar a cabo la Retrospectiva		C
16	Propietario	Cambiar las prioridades a las historias de usuario	Mantener las prioridades reales de las historias de usuario, utilizando el método MosCow.	M
17	Miembro del equipo	Agregar tareas a las historias de usuario	Indicar las tareas necesarias para completar cada historia de usuario, así como indicar la persona encargada de realizar dicha tarea	S

Historias de Usuario

#Historia	Como un ...	Quiero ...	Para ...	Prioridad (MoSCoW)
18	Miembro del equipo	Agregar comentarios a las historias de usuario	Considerar comentarios adicionales en las historias de usuario.	C
19	Miembro del equipo.	Diferenciar por colores las tareas y los comentarios	Poder resaltar o diferenciar unas de otras.	C
20	Miembro del equipo	Cambiar el estado de las historias de usuario	Actualizar el estado de las historias de usuario.	M
21	Miembro del equipo	Asignarse una historia de usuario	Elegir la historia de usuario que desea realizar.	M
22	Miembro del equipo	Cambiar el sprint de una historia de usuario	Especificar en cual Sprint se va a realizar la historia de usuario.	M
23	Miembro del equipo	Iniciar reuniones entre dos o más miembros del equipos	Comunicarse y dialogar acerca de un tema específico.	C
24	Miembro del equipo	Administrar una pila de sprint	Establecer la iteración en la que se realizará la implementación de la historia de usuario	M

Historias de Usuario

#Historia	Como un ...	Quiero ...	Para ...	Prioridad (MoSCoW)
25	Usuario	Crear una organización	Poder crear proyectos, indicando el nombre, fecha de inicio y fecha límite, dentro del ámbito de una organización.	M
26	Miembro del equipo	Acceder a archivos compartidos en el proyecto.		C
27	Usuario	Crear equipos dentro de la organización	De manera que los miembros de los equipos formados, puedan ser asignados a un proyecto determinado	M
28	Usuario	Administrar las historias de usuario (vista 2)	Administrar la pila del producto y la del sprint, desde una vista de "pizarra"	M
29	Usuario	Administrar las historias de usuario (vista 3)	Administrar la pila del producto y la del sprint, desde una vista de "tarjeta".	M

Historias de Usuario

#Historia	Como un ...	Quiero ...	Para ...	Prioridad (MoScow)
30	Miembro del equipo	Ingresar la prioridad de una historias de usuario	Utilizando la técnica de estimación de póker, ingresar la prioridad estimada de la historia de usuario	M
31	Usuario	Aceptar invitaciones	Aceptar las invitaciones de otros usuarios, para unirse a un proyecto, equipo u organización.	S
32	Miembro del Equipo	Visualizar Gráficos Burn Down y Burn Up	Poder visualizar el progreso.	M
33	Miembro del equipo	Participar en el foro del proyecto	Compartir opiniones o iniciar discusiones acerca del proyecto.	M
34	Miembro del equipo	Mantener actualizada una lista de tareas	Visualizar las tareas pendientes.	S

3.2.1. LEYENDA

Para la priorización de la pila de producto mostrada en la Tabla 3.1 se utilizó la técnica MosCoW. A continuación se muestra, a manera general en la Tabla 3.2, la descripción de cada una de las siglas utilizadas.

Tabla 3.2:

TÉCNICA	Significado	Descripción
MOSCOW	Prioridad	
M	MUST	Es indispensable que la herramienta cuente con esta funcionalidad.
S	SHOULD	Además debería contar con esta funcionalidad.
C	COULD	Puede contar con esta funcionalidad, pero sin afectar al resto.
W	WOULD	Podría contar con la funcionalidad en el futuro.

3.3. SPRINT 1

En las siguientes secciones se mostrará el detalle del desarrollo de los cuatro sprints del proyecto.

3.3.1. PILA DEL SPRINT

Considerando las prioridades establecidas en la pila del producto, en la planificación del primer sprint se establecieron las historias necesarias de desarrollar tomando en cuenta las prioridades de las historias y considerando que al final de este primer sprint se debe tener un producto potencialmente entregable. En la Tabla 3.3, también se muestran las estimaciones y los tiempos verdaderos que tomó la implementación de cada uno de los ítems de la pila.

Se debe tener en cuenta que en el presente proyecto se considera que cada día corresponde a 2:30 horas de trabajo; las estimaciones se realizaron teniendo en cuenta esta restricción.

TABLA 3.3: PILA DEL SPRINT 1

Historias de Usuario						
#Historia	Como un..	Quiero..	Para ..	Prioridad	Estimación (en días)	Tiempo (en días)
1	Usuario	Iniciar sesión	Ingresar al sistema	M	3	4
25	Usuario	Crear una organización	Poder crear proyectos, dentro del ámbito de una organización	M	3	3
27	Usuario	Crear equipos dentro de la organización	De manera que los miembros de los equipos formados, puedan ser asignados a un proyecto determinado	M	3	2
3	Usuario	Administrar un Proyecto	Iniciar nuevos proyectos ágiles	M	3	2
4	Usuario	Compartir proyectos	Añadir miembros al equipo	M	2	2
7	Usuario	Crear un Sprint	Crear las iteraciones que va a tener el proyecto	M	2	2

Historias de Usuario						
#Historia	Como un..	Quiero..	Para ..	Prioridad	Estimación (en días)	Tiempo (en días)
8	Miembro del equipo	Crear Historia de Usuario	Agregar ítems a la pila del producto	M	2	2
5	Miembro del equipo	Visualizar todas las historias de usuario de un proyecto		M	3	3
11	Scrum Mánager	Administrar sprints.	Crear sprints en el proyecto, modificar el tiempo de duración de cada una, etc.	M	3	2
22	Miembro del equipo	Cambiar el sprint de una historia de usuario	Especificar en cual Sprint se va a realizar la historia de usuario.	M	1	1
24	Miembro del equipo	Administrar una pila de sprint	Establecer la iteración en la que se realizará la implementación de la historia de usuario	M	2	1

Como se puede observar en la Tabla 3.3, el tiempo estimado para la implementación de las historias de usuarios fue un poco mayor al tiempo real, esto se pasó principalmente porque al momento de realizarse la estimación no se tenía información de referencia para estimar la velocidad con la que se realizaría la implementación de las historias. A medida que se avanzó con el proyecto, como se podrá observar en las siguientes secciones, los tiempos de estimación y ejecución se fueron emparejando.

3.3.2. GRÁFICO BURN-DOWN

A continuación, en la Ilustración 3.2, se muestra el gráfico Burn-Down del Sprint 1, la línea roja muestra el esfuerzo estimado restante, mientras que la línea azul muestra el esfuerzo real restante, de esta manera se puede observar mediante este gráfico si se está avanzando a buen ritmo durante el sprint; si la línea azul se encuentra encima de la línea roja significa que existe un retraso en las actividades, en cambio si la línea roja, se encuentra encima significa que se está avanzando con anticipación.

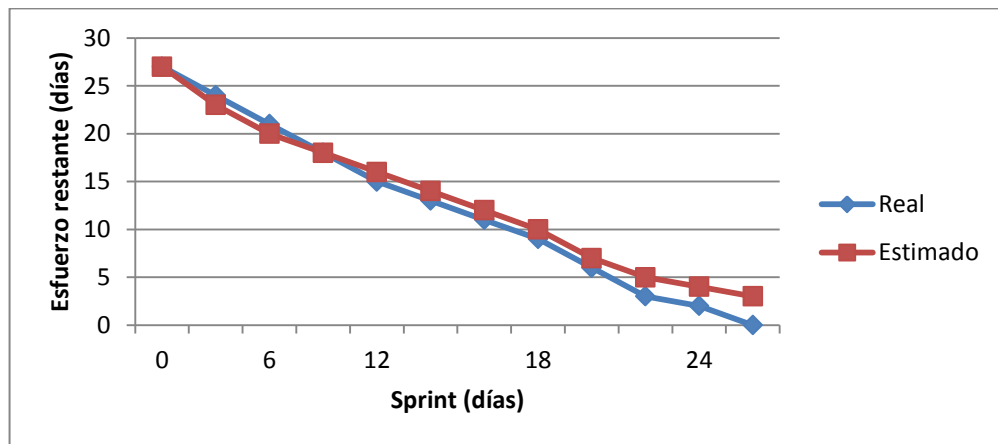


ILUSTRACIÓN 3.2: GRÁFICO BURN-DOWN DEL SPRINT 1

3.3.3. RETROSPECTIVA DEL SPRINT

La retrospectiva del sprint se realiza básicamente para analizar los problemas que se han presentado durante el sprint y poder tomar acciones correctivas. A continuación, en la Tabla 3.4, se muestra el resultado obtenido en la retrospectiva del sprint.

TABLA 3.4: RETROSPECTIVA DEL SPRINT 1

Problema	Causas	Acciones
Al implementar las historias de usuario, el alcance de algunas iba creciendo en medio del desarrollo.	Antes de iniciar la implementación de las historias de usuario, no se establecieron las tareas necesarias para su desarrollo.	A toda historia de usuario se le debe detallar con las tareas que se deben realizar para su cumplimiento.

3.4. SPRINT 2

A continuación se muestran la pila y la retrospectiva del segundo sprint.

3.4.1. PILA DEL SPRINT

Al igual que en el punto anterior, a continuación se mostrará, en la Tabla 3.4, la pila correspondiente al segundo sprint.

TABLA 3.5: PILA DEL SPRINT 2

Historias de Usuario						
#Historia	Como un ...	Quiero ...	Para ...	Prioridad	Estimación (en días)	Tiempo (en días)
2	Usuario	Enviar invitaciones	Invitar a los miembros del equipo a unirse al proyecto	M	2	2
20	Miembro del equipo	Cambiar el estado de las historias de usuario	Actualizar el estado de las historias de usuario.	M	2	1
28	Usuario	Administrar las historias de usuario (vista 2)	Administrar la pila del producto y la del sprint, desde una vista de "pizarra"	M	5	5

Historias de Usuario

#Historia	Como un ...	Quiero ...	Para ...	Prioridad	Estimación (en días)	Tiempo (en días)
30	Miembro del equipo	Ingresar la prioridad de una historias de usuario	Utilizando la técnica de estimación de póker, ingresar la prioridad estimada de la historia de usuario	M	5	7
31	Usuario	Aceptar invitaciones	Aceptar las invitaciones de otros usuarios, para unirse a un proyecto, equipo u organización.	S	3	2
21	Miembro del equipo	Asignarse una historia de usuario	Elegir la historia de usuario que desea realizar.	M	2	2
16	Propietario	Cambiar las prioridades a las historias de usuario	Mantener las prioridades reales de las historias de usuario a lo largo del proyecto.	M	2	1
29	Usuario	Administrar las historias de usuario (vista 3)	Administrar la pila del producto y la del sprint, desde una vista de "tarjeta".	M	4	4

Como se puede observar en la Tabla 3.5, la diferencia de entre los tiempos estimados y reales ha disminuido con respecto al sprint anterior.

3.4.2. GRÁFICO BURN-DOWN

A continuación la Ilustración 3.3 muestra el gráfico Burn Down, correspondiente al segundo sprint.

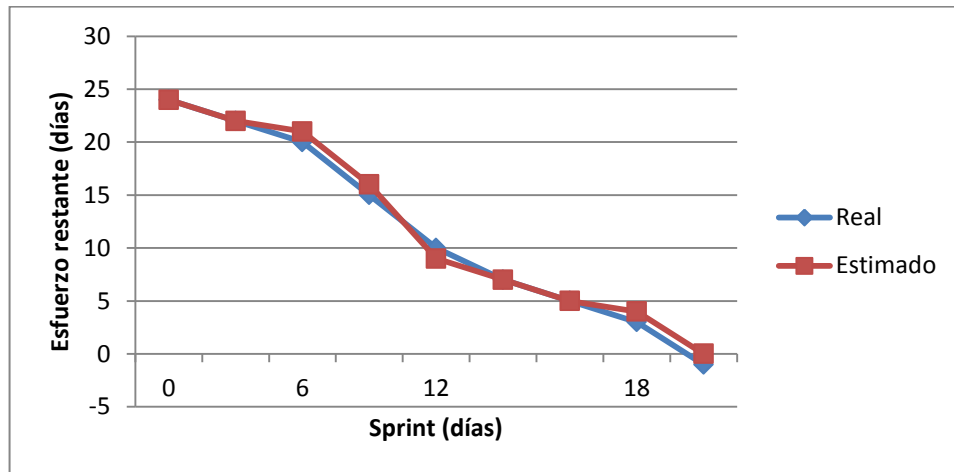


ILUSTRACIÓN 3.3: GRÁFICO BURN-DOWN DEL SPRINT 2

3.4.3. RETROSPECTIVA DEL SPRINT

A continuación, en la Tabla 3.6, se muestra el resultado obtenido en la retrospectiva del sprint.

TABLA 3.6: RETROSPECTIVA DEL SPRINT 2

Problema	Causas	Acciones
En algunas ocasiones, se realizó trabajo extra innecesario.	Al momento de implementar la historia de usuario, no se tuvo claro lo que esperaba el propietario del producto.	Consultar al propietario del producto, todas las historias de usuario en las que se tenga alguna duda respecto a lo que se va a implementar.

3.5. SPRINT 3

A continuación se muestra la pila del tercer sprint, el gráfico burn-down y la retrospectiva del mismo.

3.5.1. PILA DEL SPRINT

En la Tabla 3.7, se muestra la pila correspondiente al tercer sprint.

TABLA 3.7: PILA DEL SPRINT 3

Historias de Usuario					Estimación	Tiempo
#Historia	Como un ...	Quiero ...	Para ...	Prioridad	(en días)	(en días)
6	Usuario	Comunicarme con otro usuario mediante la herramienta	Compartir dudas o avances sobre el proyecto	S	3	3
32	Miembro del Equipo	Visualizar Gráficos Burn Down y Burn Up	Poder visualizar el progreso.	M	3	4
33	Miembro del equipo	Participar en el foro del proyecto	Compartir opiniones o iniciar discusiones acerca del proyecto.	S	8	8

Historias de Usuario

#Historia	Como un ...	Quiero ...	Para ...	Prioridad	Estimación (en días)	Tiempo (en días)
9	Scrum Mánager	Agregar roles	Agregar roles personalizados, además de los roles por defecto.	S	2	2
10	Scrum Mánager	Administrar roles	Cambiar los roles de los miembros del equipo en diferentes momentos del proyecto	M	3	3
17	Miembro del equipo	Agregar tareas a las historias de usuario	Indicar las tareas necesarias para completar cada historia de usuario, así como indicar la persona encargada de realizar dicha tarea	S	2	3
18	Miembro del equipo	Agregar comentarios a las historias de usuario	Considerar comentarios adicionales en las historias de usuario.	C	2	1
34	Miembro del equipo	Mantener actualizada una lista de tareas	Visualizar las tareas pendientes	C	2	2
19	Miembro del equipo	Diferenciar por colores las tareas y los comentarios	Poder resaltar o diferenciar unas de otras.	C	2	2

3.5.2. GRÁFICO BURN-DOWN

A diferencia de los sprints anteriores, en el tercer sprint se presentó un retraso en las actividades; el cuál se puede apreciar analizando la Ilustración 3.4, en el que línea azul que representa al esfuerzo real restante se encuentra por encima de la línea de esfuerzo estimado restante.

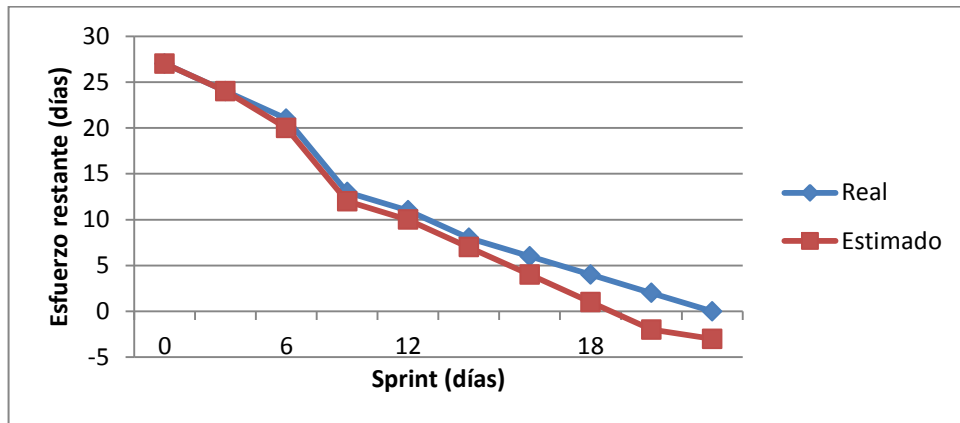


ILUSTRACIÓN 3.4: GRÁFICO BURN-DOWN DEL SPRINT 3

3.5.3. RETROSPECTIVA DEL SPRINT

A continuación, en la Tabla 3.8, se muestra el resultado obtenido en la retrospectiva del sprint.

TABLA 3.8: RETROSPECTIVA DEL SPRINT 3

Problema	Causas	Acciones
Tiempo real de implementación por encima de lo estimado	No se consideraron factores académicos externos que disminuyen la velocidad de implementación de la herramienta.	Evitar en la medida de lo posible que factores externos impacten en el desarrollo del proyecto; en caso esto no pueda evitarse, es necesario contemplarlo durante la planificación del sprint.

3.6. SPRINT 4

3.6.1. PILA DEL SPRINT

En la Tabla 3.9, se muestra la pila correspondiente al cuarto y último sprint.

TABLA 3.9: PILA DEL SPRINT 4

Historias de Usuario						
#Historia	Como un ...	Quiero ...	Para ...	Prioridad	Estimación (en días)	Tiempo (en días)
26	Miembro del equipo	Acceder a archivos compartidos en el proyecto.	Contar con un repositorio de archivos comunes al proyecto	C	5	5
23	Miembro del equipo	Iniciar reuniones entre dos o más miembros del equipos	Comunicarse y dialogar acerca de un tema específico.	C	5	5
12	Scrum Mánager	Programar y llevar a cabo el reuniones		C	5	5
14	Scrum Mánager	Programar y llevar a cabo la reunión de planificación de la Pila del producto.		C	4	4
13	Scrum Mánager	Programar y llevar a cabo la reunión de planificación del Sprint.		C	4	4
15	Scrum Mánager	Programar y llevar a cabo la Retrospectiva		C	5	4
					28	27

3.6.2. GRÁFICO BURN-DOWN

En la Ilustración 3.5, se observa que durante casi todo el sprint, el esfuerzo real restante es similar al esfuerzo estimado restante.

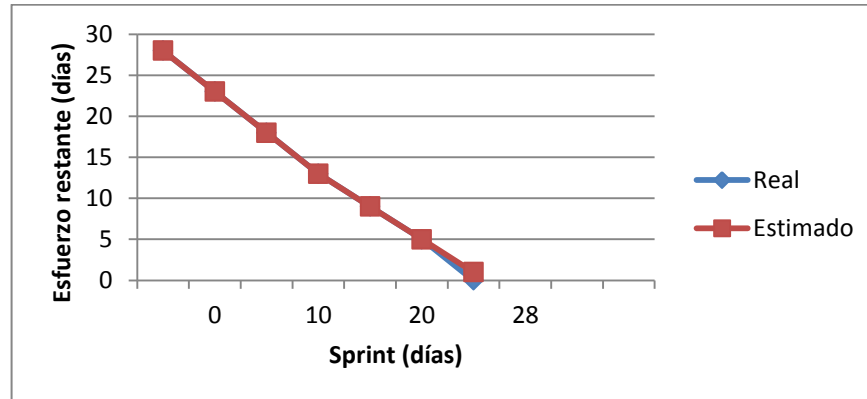


ILUSTRACIÓN 3.5: GRÁFICO BURN-DOWN DEL SPRINT 4

3.6.3. RETROSPECTIVA DEL SPRINT

Durante la realización del sprint 4, no se encontraron inconvenientes; por otro lado se resalta, que el tiempo de estimación con el tiempo real de desarrollo han resultado bastante similares; esto se debe en parte a que con los sprints anteriores se ha ganado experiencia en la estimación y se conoce mejor los tiempos que tardan las funcionalidades en ser implementadas.

4. DISEÑO

En el presente capítulo se detallará el diseño de la solución propuesta y la arquitectura empleada. Asimismo, se explicarán los criterios del diseño de la interfaz gráfica de usuario.

4.1. ARQUITECTURA DE LA SOLUCIÓN

La arquitectura constituye el diseño a alto nivel de una aplicación donde se incluye la definición de componentes e interfaz.

A continuación se presenta una visión general de la arquitectura de la herramienta, por lo que se abarcará diferentes aspectos del sistema.

4.1.1. ACRÓNIMOS

MVC	Modelo Vista Controlador. Es un patrón de diseño de Software que separa la vista, la lógica del negocio y los datos.
CakePHP	CakePHP. Es un <i>framework</i> que implementa el patrón MVC para aplicaciones web con el lenguaje PHP.

4.1.2. DEFINICIÓN DE LA ARQUITECTURA

Dadas las condiciones expuestas en los capítulos anteriores, se eligió una arquitectura Web para el desarrollo de la herramienta, la cual cuenta con los siguientes beneficios:

Los usuarios pueden acceder desde cualquier equipo de cómputo, mediante un navegador web, sin necesidad de instalar ninguna aplicación adicional.

La información está centralizada en un repositorio, de manera que solo se necesita conexión a internet para su acceso.

En la Ilustración 4.1 se muestra la visión general de la arquitectura propuesta.

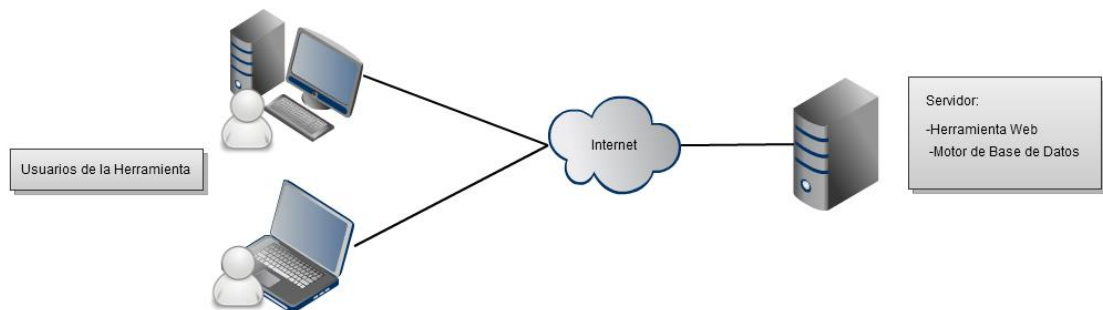


ILUSTRACIÓN 4.1: VISTA GENERAL DE LA ARQUITECTURA

4.1.3. REPRESENTACIÓN DE LA ARQUITECTURA

4.1.3.1. Patrón de Diseño de Arquitectura

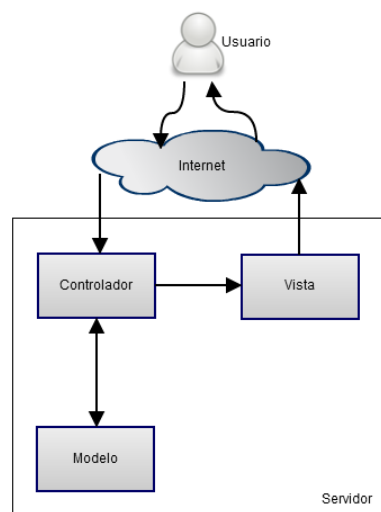


ILUSTRACIÓN 4.2: MODELO-VISTA-CONTROLADOR

El patrón de diseño utilizado en la herramienta es Modelo-Vista-Controlador (MVC), ya que las capas que la conforman se separan adecuadamente y se adaptan a un sistema cliente servidor. En la Ilustración 4.2 se explica de manera gráfica el funcionamiento del patrón de diseño.

4.1.3.2. Enfoques arquitectónicos

Los enfoques arquitectónicos usados para la elaboración del sistema son:

- *Jerarquía de capas*

La herramienta se basa en jerarquía de capas, la cual implementa el patrón de arquitectura MVC, en donde cada capa representará cada componente del mencionado patrón.

- *Orientado a Objetos*

Las entidades del negocio, sus gestores y la aplicación en general se registrarán bajo un paradigma orientado a objetos

- *Orientado a Eventos*

Los controles Web encargados de interactuar con el usuario utilizarán eventos en el servidor y en el entorno cliente para lograr este fin.

- *Repositorio*

Para el desarrollo de la herramienta se empleará una base de datos, la cual servirá de repositorio

- *Arquitectura Web*

El servidor es aquel que contiene la herramienta y es el encargado de brindar respuestas frente a las solicitudes de los usuarios. Los usuarios son aquellos que acceden al sistema mediante una conexión Web y un navegador y envían peticiones al sistema. El servidor también contiene al motor de base de datos.

4.1.4. VISTA LÓGICA

En la Ilustración 4.3 se muestra el diagrama de clases que se han utilizado para la implementación de herramientas, el framework escogido, CakePHP, brinda los siguientes lineamientos para la creación de tablas de base de datos, modelos, controladores y vistas:

- Las tablas de la base de datos se deben nombrar en plural y con minúsculas; en caso se utilice más de una palabra para nombrarla la separación se debe realizar con el símbolo '_'. Por ejemplo: organizations y org_members.
- Los modelos deben ser el singular de la tabla de la base de datos. Además se utiliza un nombre en *CamelCase*, es decir el inicio de cada palabra está delimitado por una mayúscula y no por un espacio. Siguiendo con el ejemplo anterior, las modelos correspondientes a las tablas mencionadas serían: Organization y OrgMember.
- Las clases controladoras se deben nombrar en plural, tomando como referencia el nombre del modelo, seguido de la palabra "Controller"; en este caso también se usará el CamelCase. Las clases controladoras correspondientes al ejemplo serían: OrganizationsController y OrgMembersControllers.
- Las vistas se mantendrán ordenadas mediante un sistema de carpetas, cada carpeta se llamará con el nombre en plural del modelo, y cada archivo corresponderá a una función declarada en el controlador asociado. Por ejemplo, en OrganizationsController existe una función llamada "index"; por lo tanto en a sección de vista existirá una un carpeta llamada "Organizations", la cual tendrá un archivo llamado "index.ctp"; esta extensión
- corresponde a los archivos de vista creados para CakePHP.
- Se debe considerar además que en CakePHP, todos los modelos heredan de una clase "AppModel", y las clases controladoras heredan de "AppController". En la Ilustración 4.3, Diagrama de Clases de Diseño, no se han considerado estas clases de manera que se pueda mostrar de una manera más clara las clases correspondientes a la implementación de la herramienta.

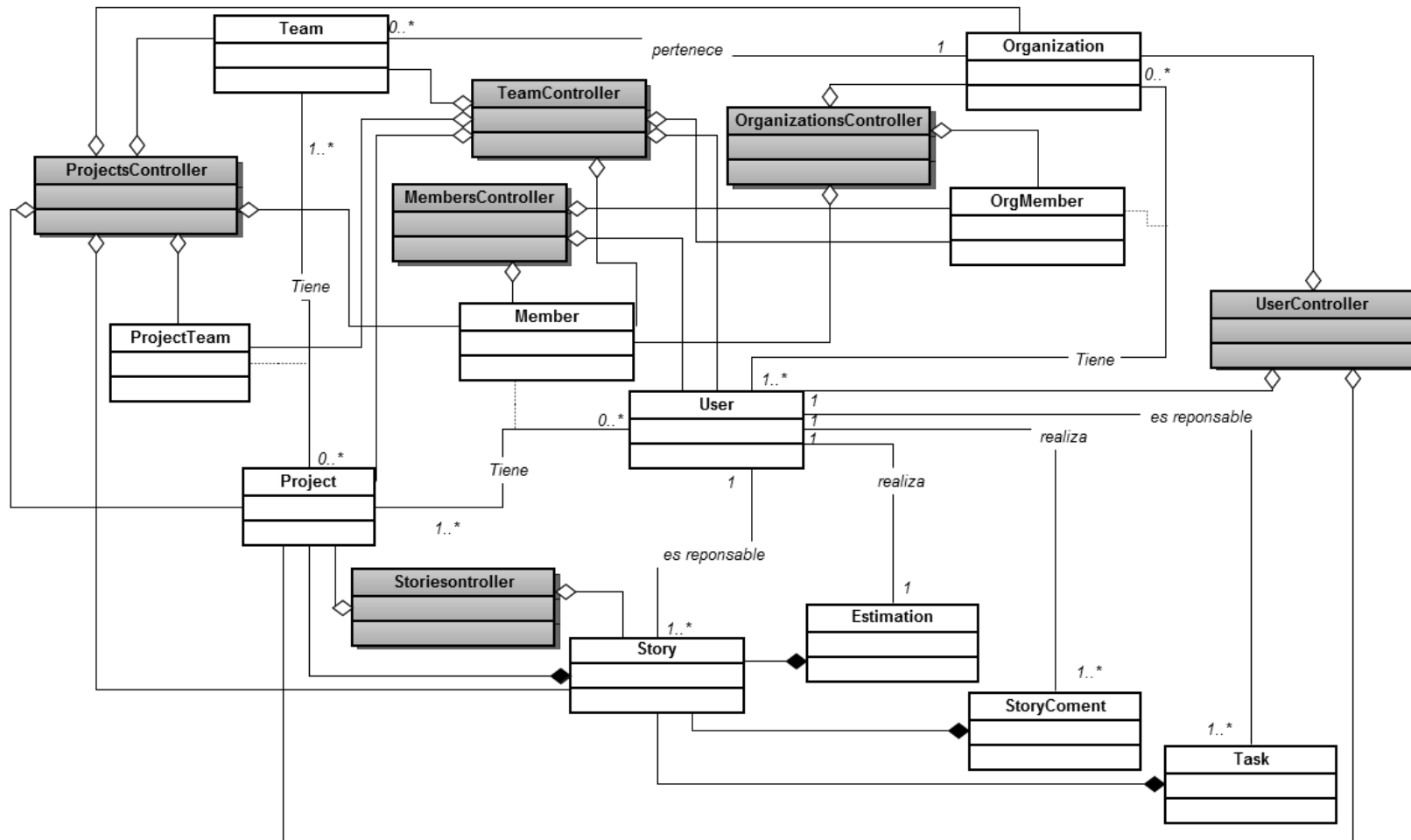


ILUSTRACIÓN 4.3: DIAGRAMA DE CLASES DE DISEÑO

4.1.5. VISTA DE IMPLEMENTACIÓN

Esta vista describe cómo se implementan los componentes físicos de la herramienta.

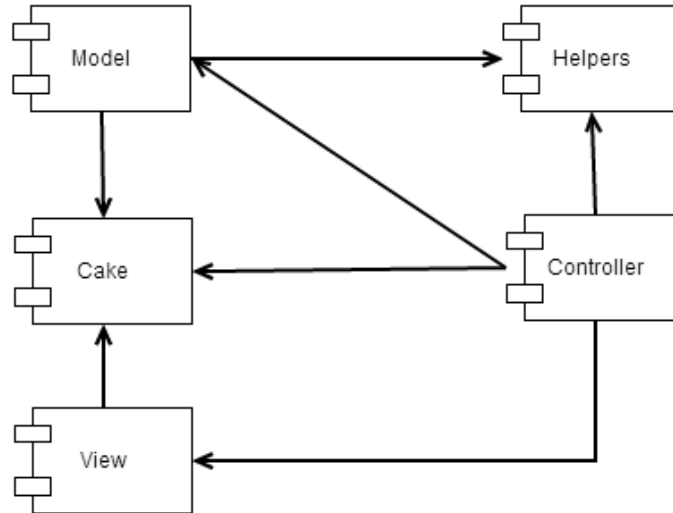


ILUSTRACIÓN 4.4: DIAGRAMA DE COMPONENTES

- **Model:**
 - Permite acceder al almacenamiento de datos. Es independiente al sistema de almacenamiento de los datos.
 - Lleva un registro de las vistas y controladores del sistema.
 - Para los flujos principales del sistema, este componente contiene las siguientes clases: Proyecto, Historia, Equipo, Organización, entre otras.
- **Controller:**
 - Recibe los eventos de entrada.
 - Contiene reglas de gestión de eventos, estas acciones pueden suponer peticiones al modelo o a las vistas.
 - Contiene los archivos relacionados a las siguientes clases: Controlador Proyecto, Controlador Historia, Controlador Equipo, Controlador Organización, entre otras.
- **View:**
 - Recibe los datos de modelo y los muestra al usuario.
 - Tiene un registro de su controlador asociado.

- **Cake:**
 - Este componente contiene las clases base ofrecidas por el marco de trabajo (Framework) que fueron útiles para el desarrollo de la herramienta
- **Helpers:**
 - Contiene las clases que ofrecen operaciones comúnmente utilizadas.
 - Los Helpers que fueron más utilizados fueron: FormHelper y HtmlHelper.

4.1.6. VISTA DE DESPLIEGUE

A continuación se muestra el diagrama de despliegue del sistema con el objetivo de explicar la organización física en este.

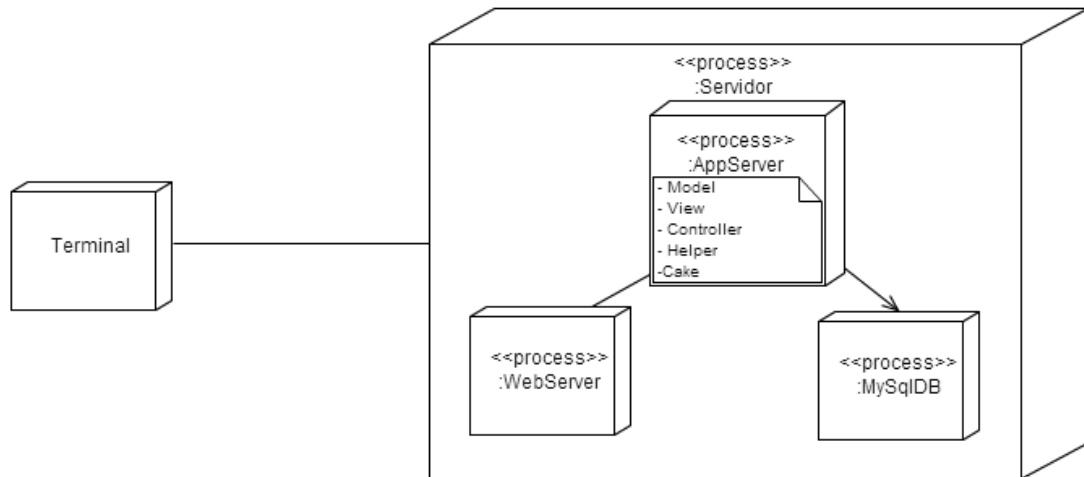


ILUSTRACIÓN 4.5: DIAGRAMA DE DESPLIEGUE

Los nodos que ejecutan componentes del sistema están agrupados en el nodo "Servidor", el cual representa el equipo físico en donde se ejecutan los procesos correspondientes al Servidor Web (Apache), al servidor de aplicaciones (motor de PHP) y al sistema administrador de base de datos utilizado en la solución (MySQL).

4.2. DISEÑO DE LA INTERFAZ GRÁFICA

Los prototipos de las pantallas principales que se obtuvieron luego de la presentación de las pantallas preliminares y los acuerdos que se sostuvo con el propietario del producto, se pueden observar en el Anexo A.

Los criterios utilizados para el diseño de la interfaz se definieron a partir de heurísticas de usabilidad (Nielsen, 2005), de los cuales se seleccionaron los que

más significativos para el contexto del proyecto. Los criterios que se utilizaron fueron:

- Concordancia entre el sistema y el mundo real, se utilizó nomenclatura propia de Scrum.
- Prevención de errores, se identificaron los posibles errores que podrían aparecer y se incluyeron mensajes claros y en lenguaje natural.
- Interfaz explorable, se optó por mantener una barra de navegación principal en la parte superior de la página siempre visible. Además en las pantallas relacionadas a las funcionalidades de gestión del proyecto, se tiene además un menú flotante con enlaces rápidos.
- Diseño minimalista, se consideró que es necesario mantener una línea limpia en el diseño de manera que el usuario no se sienta abrumado por una sobrecarga de colores, por tal motivo el color de fondo escogido fue el color blanco, para el color de la fuente se escogieron tonalidades oscuras y los iconos utilizados fueron monocromáticos.
- Ayuda a los usuarios a reconocer, diagnosticar y recuperarse de los errores; para cumplir con este criterio, se incluyeron mensajes de confirmación y de error, en ambos casos los mensajes fueron claros y concisos.

A continuación, en la Ilustración 4.6 se muestra el prototipo de información general del proyecto

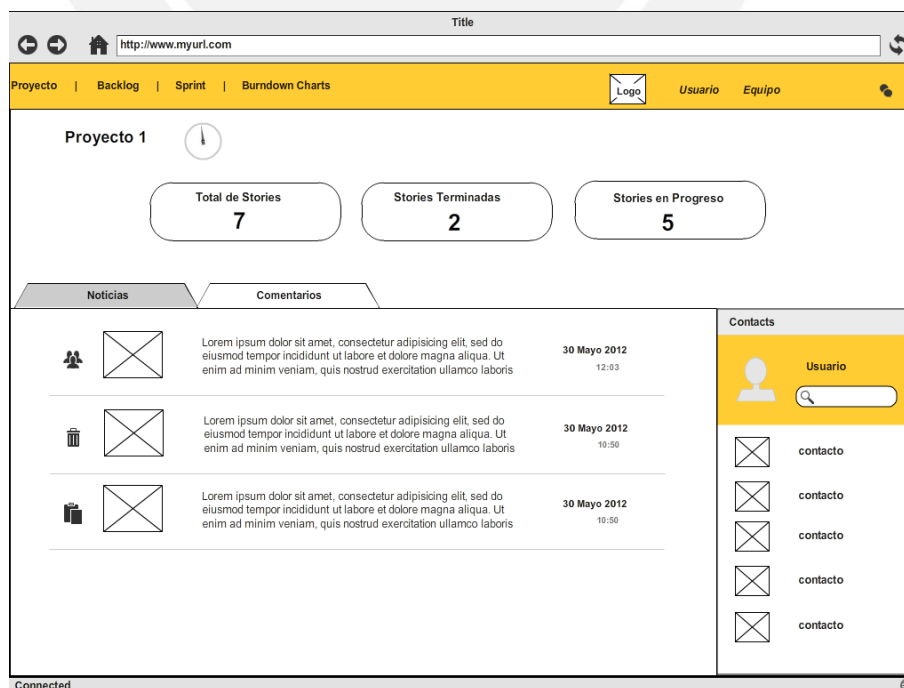


ILUSTRACIÓN 4.6: PROTOTIPO DE INFORMACIÓN DEL PROYECTO

En la Ilustración 4.7, 4.8 y 4.9 se muestran los tres tipos de visualizaciones de la pila del producto con la que cuenta la herramienta.

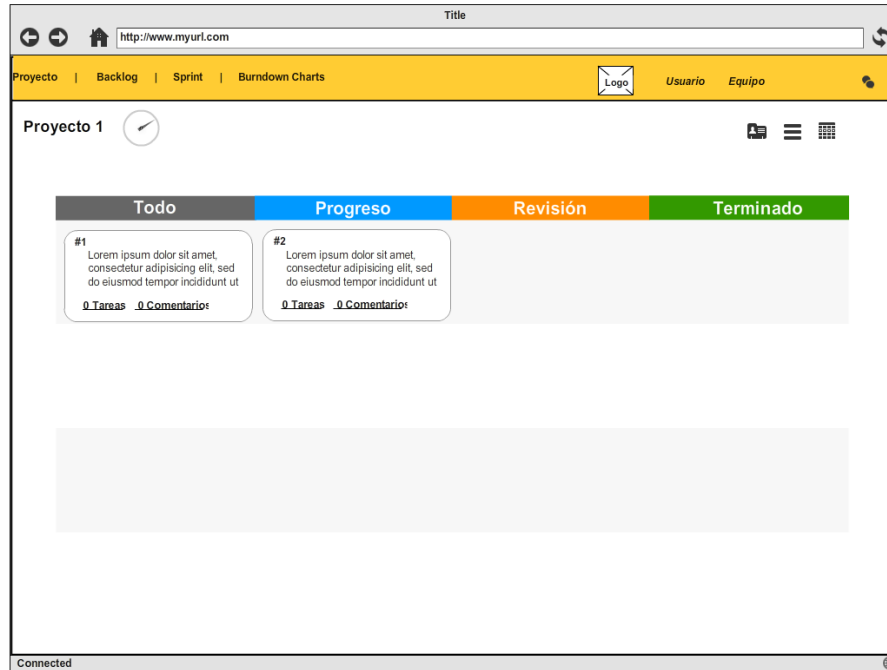


ILUSTRACIÓN 4.7: PROTOTIPO DE VISUALIZACIÓN DE LA PILA DEL PRODUCTO (1)

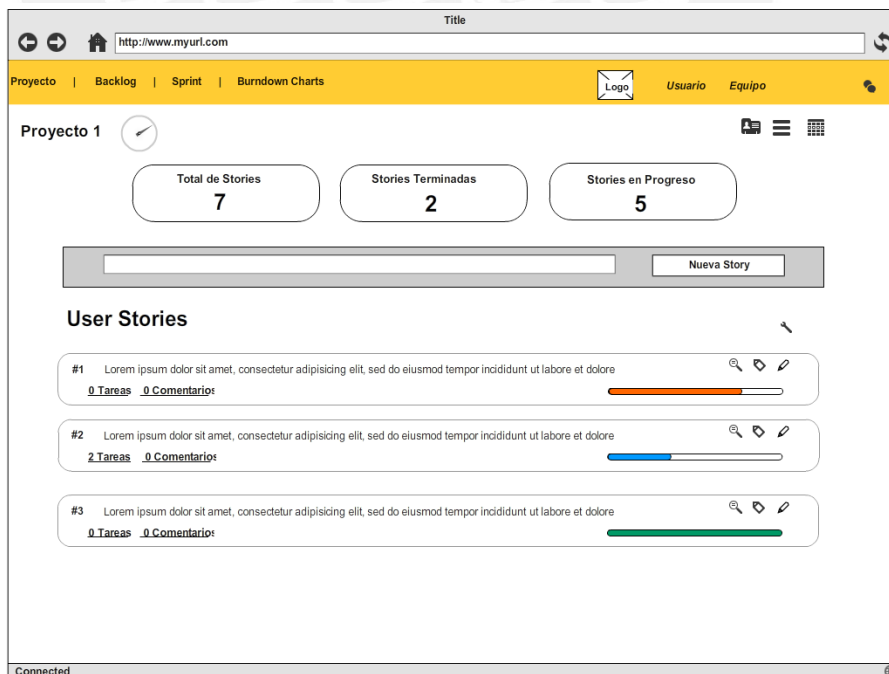


ILUSTRACIÓN 4.8: PROTOTIPO DE VISUALIZACIÓN DE LA PILA DEL PRODUCTO (2)

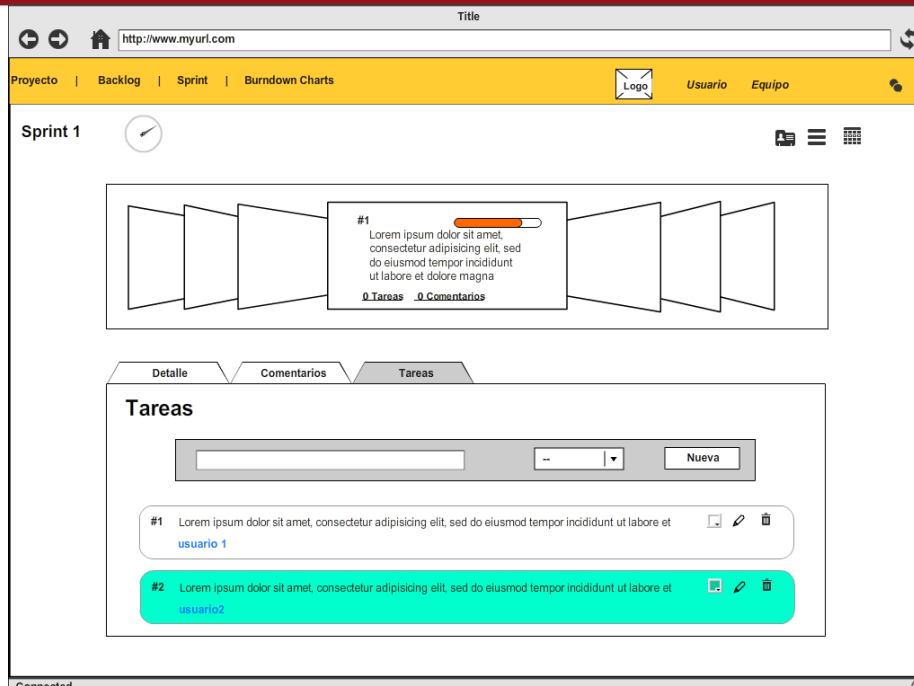


ILUSTRACIÓN 4.9: PROTOTIPO DE VISUALIZACIÓN DE LA PILA DEL PRODUCTO (3)

El presente proyecto tiene como objetivo dar soporte a equipos distribuidos; considerando éste se presentan a continuación prototipos de pantalla correspondientes a reuniones de Scrum como en la Ilustración 4.10 y la Ilustración 4.11 que se muestran la revisión del Sprint y Retrospectiva, respectivamente

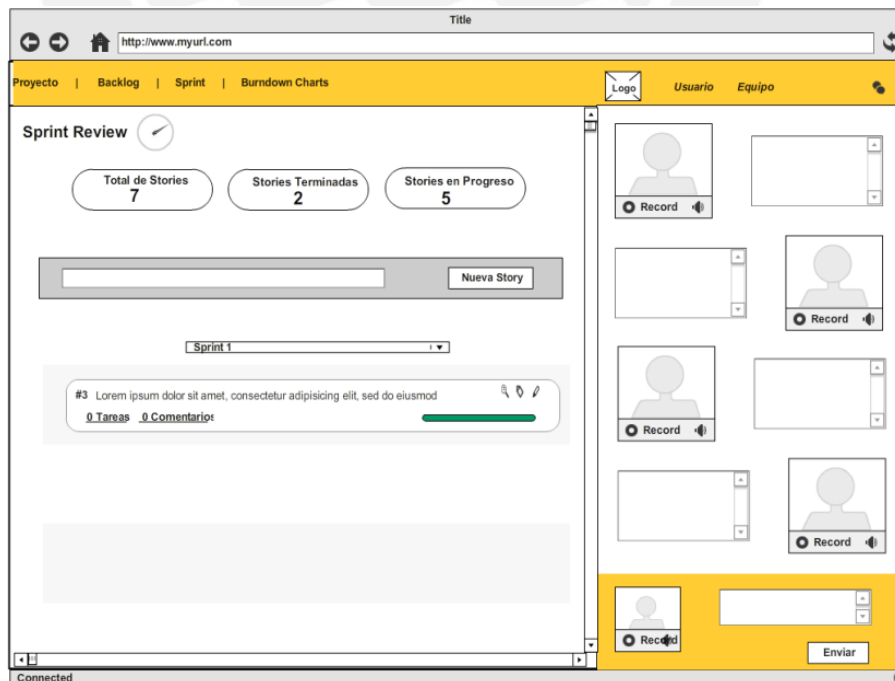


ILUSTRACIÓN 4.10: PROTOTIPO DE REVISIÓN DEL SPRINT

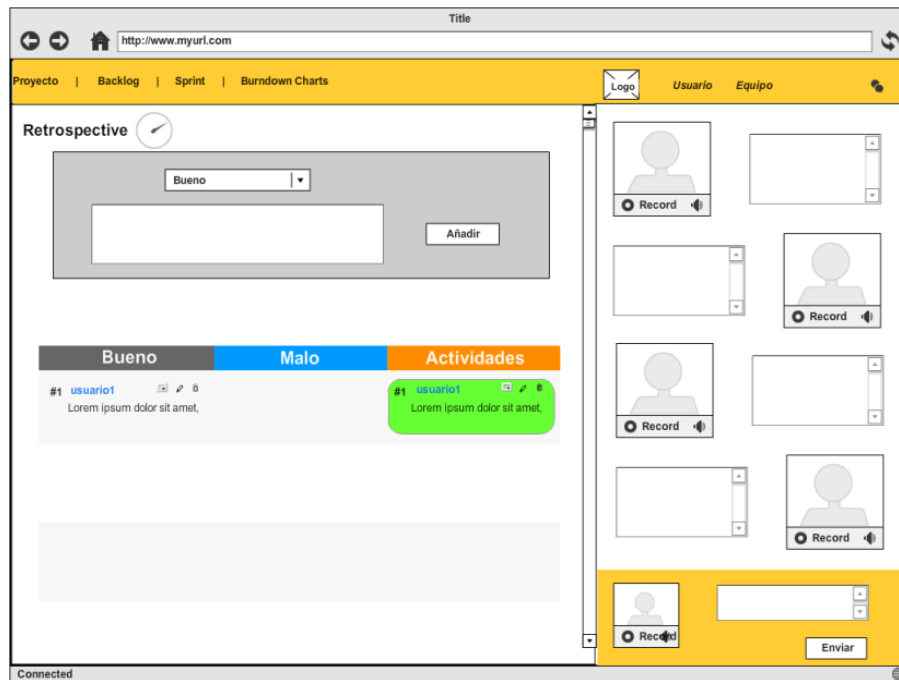


ILUSTRACIÓN 4.11: PROTOTIPO DE RETROSPECTIVA

Para cumplir con el objetivo del proyecto, es importante además el uso de mecanismos de comunicación, cuyos prototipos se muestra en la Ilustración 4.13 y la Ilustración 4.14 respectivamente.

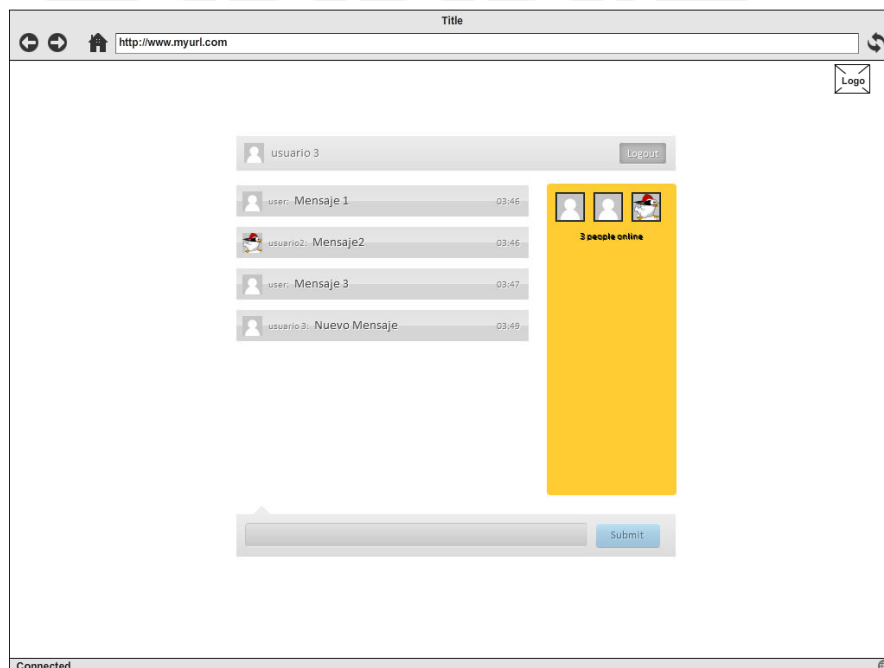


ILUSTRACIÓN 4.12: PROTOTIPO DE CHAT

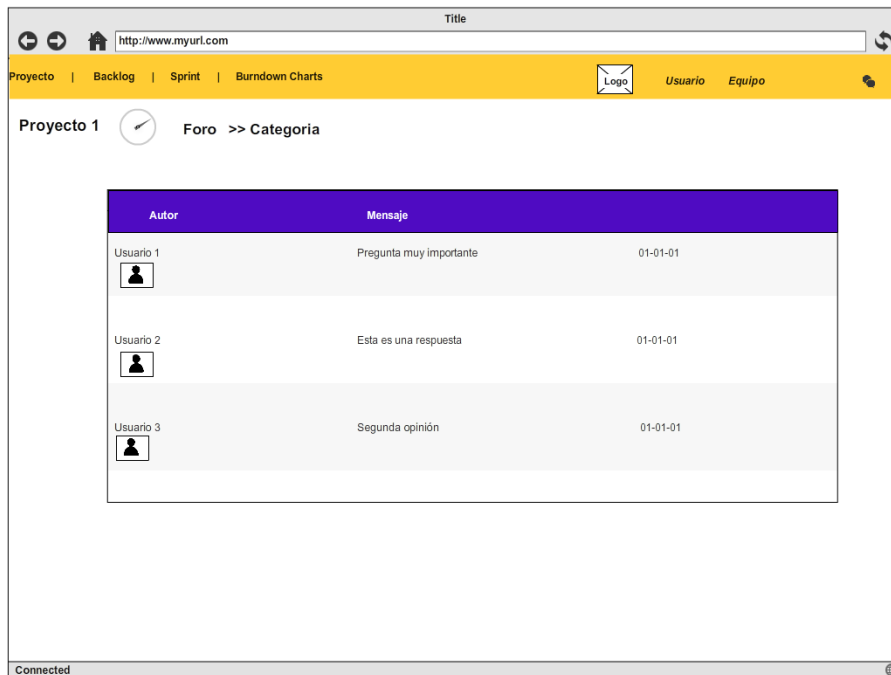


ILUSTRACIÓN 4.13: PROTOTIPO DE FORO

5. CONSTRUCCIÓN

En el presente capítulo se muestran las decisiones tomadas para el desarrollo del producto final.

5.1. CONSTRUCCIÓN

En esta sección se menciona y justifica el uso de las tecnologías y marcos de trabajo (framework) utilizados en la construcción de la herramienta.

5.1.1. *TECNOLOGÍA: LENGUAJE PHP*

PHP es un lenguaje tipo script ampliamente utilizado que es especialmente adecuado para desarrollo web y puede ser embebido en HTML (The PHP Group, 2011). Sus siglas corresponden a “Hypertext Preprocessor” o procesador de hipertexto.

Se determinó realizar el desarrollo de la herramienta en este lenguaje por los siguientes motivos:

- Conocimiento previo del lenguaje: El proyecto tiene fechas de entregas continuas, sin lugar a retraso, por tal motivo se considera ventajoso contar con un conocimiento previo del lenguaje de programación.
- Licencia gratuita: No cuenta con gastos de adquisición ni mantenimiento.
- Soporte a la Programación Orientada a Objetos. Se utilizará la versión 5.3.17 de PHP, la cual soporta el desarrollo orientado a objetos.
- Sitios Web livianos y Portables. Al ser un lenguaje Script, la interpretación y respuesta desde el servidor se realiza de manera óptima. Estos sitios pueden ser publicados y usados desde equipos clientes con distintos sistemas operativos.

5.1.2. *FRAMEWORK: CAKEPHP*

Para el desarrollo de la aplicación se utilizará el framework CakePhp versión 2.2.1, la cual constituye una versión estable para su uso en desarrollos de proyectos Web. CakePHP, inspirado en Ruby on Rails, es un framework para la construcción de sitios web que utilicen una base de datos como fuente de recursos (Cake Software Foundation, 2012).

Los principales beneficios que brinda CakePHP, son los siguientes:

- Propone la utilización de la Arquitectura MVC
- Minimización de tiempos en implementación mediante el acceso a datos, dado que utiliza la técnica del ORM (Mapeo Relacional de Objetos).
- Estructura de aplicaciones (Application Scaffolding), permite al programador hacer uso de un conjunto de convenciones aplicables a la estructura de la base de datos de la aplicación y el framework se encarga de generar el código para la interacción a lo largo de todas las capas de la aplicación.
- Ofrece componentes que implementan el manejo de sesiones, la seguridad y el manejo de peticiones por parte del cliente.
- Ofrece asistentes de construcción de vistas que facilitan el uso de otras tecnologías como AJAX (Asynchronous JavaScript and XML), Javascript, XML, CSS; además de la creación de páginas y formularios Web.

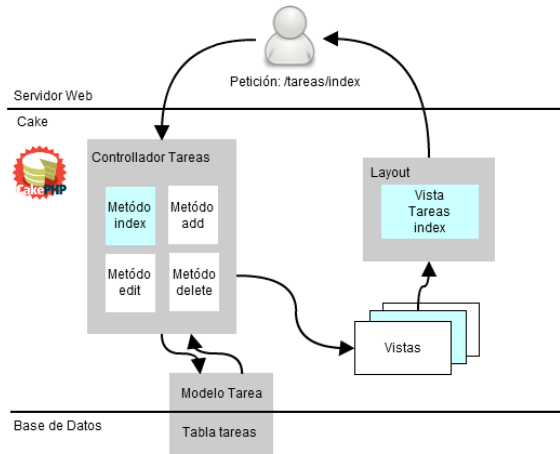


ILUSTRACIÓN 5.1: MVC CON CAKEPHP

Se usó PHP Coding Standard (Hoff, 2002), para los estándares de programación de la herramienta

5.2. PRUEBAS

Considerando que para el presente proyecto los ítems que se utilizaron en la pila del producto fueron historias de usuario, para la realización de las pruebas se usaron las pruebas de aceptación de cada una de estas historias; además antes de finalizado el sprint se realizaron pruebas de integración. Las correcciones necesarias se realizaron inmediatamente después de encontrado un defecto. En la Ilustración 5.1, se puede observar el proceso de pruebas del producto potencialmente entregable al final de cada sprint.

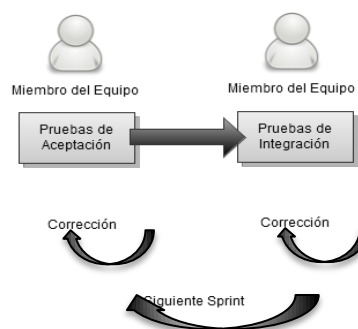


ILUSTRACIÓN 5.2: PROCESO DE PRUEBAS POR SPRINT

A continuación se muestran las pruebas de aceptación realizadas en cada sprint.

5.2.1. SPRINT 1

En la Tabla 5.1, se muestra la pila del sprint 1, en la que se detallan las pruebas de aceptación de cada historia de usuario.

TABLA 5.1: PRUEBAS DE ACEPTACIÓN DEL SPRINT 1

Historias de Usuario				
#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
1	Usuario	Iniciar sesión	Ingresar al sistema	<ul style="list-style-type: none"> - El usuario ingresa una clave y/o nombre de usuario incorrectos, se muestra un mensaje de error indicando que los campos son incorrectos. -El usuario ingreso el nombre de usuario y la clave correcta, se muestra un mensaje de bienvenida.
25	Usuario	Crear una organización	Poder crear proyectos, dentro del ámbito de una organización	-El usuario crea, modifica y elimina organizaciones. Se muestra un mensaje de confirmación.

Historias de Usuario				
#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
27	Usuario	Crear equipos dentro de la organización	De manera que los miembros de los equipos formados, puedan ser asignados a un proyecto determinado	<ul style="list-style-type: none"> - El usuario ingresa los datos necesarios para crear un equipos; solo se considera necesario el nombre del equipo, opcionalmente se ingresa la descripción. Se muestra un mensaje de confirmación. -El usuario al crear un equipo, no ingresa el nombre; se muestra un mensaje de error indicando que el nombre es obligatorio para registrar el equipo.
3	Usuario	Administrar un Proyecto	Iniciar nuevos proyectos ágiles	<ul style="list-style-type: none"> - El usuario ingresa los datos necesarios para crear un proyecto; solo se considera necesario el nombre del proyecto, opcionalmente se ingresa la descripción. Se muestra un mensaje de confirmación. -El usuario al crear un proyecto no ingresa el nombre; se muestra un mensaje de error indicando que el nombre es obligatorio para registrar el proyecto. -Al crearse el proyecto, se crea además un equipo, por defecto con el mismo nombre.

Historias de Usuario				
#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
4	Usuario	Compartir proyectos	Añadir miembros al equipo	<p>-El usuario puede buscar a otros usuarios por nombre de usuario y se agregarlos al equipo.</p> <p>-El usuario puede añadir eliminar usuarios, solo si, él es el creador del equipo, en caso contrario se muestra un mensaje indicando la falta de privilegios.</p>
7	Usuario	Crear un Sprint	Crear las iteraciones que va a tener el proyecto	<p>- El usuario agrega sprints al proyecto. Se muestra un mensaje de confirmación.</p>
8	Miembro del equipo	Crear Historia de Usuario	Agregar ítems a la pila del producto	<p>-El usuario ingresa los datos correspondientes para crear una historia de usuario, el resumen es obligatorio y el detalle opcional. Se muestra un mensaje de confirmación.</p> <p>-El usuario intenta crear una historia de usuario vacía. Se muestra un mensaje de error indicando los campos que se deben llenar obligatoriamente.</p>

Historias de Usuario				
#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
5	Miembro del equipo	Visualizar todas las historias de usuario de un proyecto		-El miembro del equipo puede visualizar las historias de usuarias creadas a modo de lista, a nivel de proyecto y de sprint.
11	Scrum Mánager	Administrar sprints	Crear sprints en el proyecto, modificar el tiempo de duración de cada una, etc.	-El Scrum Mánager puede administrar los Sprints de los proyectos. Se muestra un mensaje de conformidad. -Un miembro de equipo intenta administrar la cantidad de sprints de un proyecto. Se muestra un mensaje error por falta de privilegios.
22	Miembro del equipo	Cambiar el sprint de una historia de usuario	Especificar en cual Sprint se va a realizar la historia de usuario.	-El usuario cambia el sprint, en el que se realizará la historia de usuario. Se guardan los cambios sin errores.
24	Miembro del equipo	Administrar una pila de sprint	Establecer la iteración en la que se realizará la implementación de la historia de usuario	-El usuario crea, modifica y elimina historias de usuario desde la vista de lista de la pila.

5.2.2. SPRINT 2

En la Tabla 5.2, se muestra la pila del sprint 2, en la que se detallan las pruebas de aceptación de cada historia de usuario.

TABLA 5.2: PRUEBAS DE ACEPTACIÓN DEL SPRINT 2

Historias de Usuario				
#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
2	Usuario	Enviar invitaciones	Invitar a los miembros del equipo a unirse al proyecto	<ul style="list-style-type: none"> -El usuario envía invitaciones a otro usuario para que sean parte del equipo. - El estado del usuario se muestra como parte del equipo, con el estado de invitado.
20	Miembro del equipo	Cambiar el estado de las historias de usuario	Actualizar el estado de las historias de usuario.	<ul style="list-style-type: none"> -El usuario crea una historia de usuario, por defecto el estado inicial es <i>ToDo</i>. -El usuario cambio de estado a la historia de usuario. El indicador de estado de la historia, cambia de acuerdo al nuevo estado.

Historias de Usuario				
#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
28	Usuario	Administrar las historias de usuario (vista 2)	Administrar la pila del producto y la del sprint, desde una vista de "pizarra"	<ul style="list-style-type: none"> - El usuario elije ver la pila desde la vista "Pizarra". Puede realizar todas las funciones que en la vista lista (la vista por defecto). - El usuario hace uso del "drag&drop" para actualizar el estado de alguna historia de usuario. La historia de usuario cambia de columna según el estado seleccionado la vista lista (la vista por defecto).
30	Miembro del equipo	Ingresar la prioridad de una historias de usuario	Utilizando la técnica de estimación de póker, ingresar la prioridad estimada de la historia de usuario	<ul style="list-style-type: none"> -El miembro del equipo realiza la estimación de la historia de usuario. Se guarda la estimación del usuario correctamente. -El Scrum Master puede ver el detalle de la estimación, así como el valor promedio calculado.

Historias de Usuario				
#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
31	Usuario	Aceptar invitaciones	Aceptar las invitaciones de otros usuarios, para unirse a un proyecto, equipo u organización.	<ul style="list-style-type: none"> - El usuario recibe la invitación, si desea ser parte del equipos acepta la invitación. - El estado del usuario cambia en el equipo de invitado a colaborador. - El usuario es parte de los proyectos, a los cuales este asignado el equipo.
21	Miembro del equipo	Asignarse una historia de usuario	Elegir la historia de usuario que desea realizar.	La historia de usuario puede ser asignada a cualquier miembro del proyecto.
16	Propietario	Cambiar las prioridades a las historias de usuario	Mantener las prioridades reales de las historias de usuario a lo largo del proyecto.	El propietario del proyecto puede cambiar las prioridades de las historias de usuario.
29	Usuario	Administrar las historias de usuario (vista 3)	Administrar la pila del producto y la del sprint, desde una vista de "tarjeta".	- El usuario elije ver la pila desde la vista "Tarjeta". Puede realizar todas las funciones que en la vista lista (la vista por defecto).

5.2.3. SPRINT 3

A continuación, en la Tabla 5.3, se muestra las pruebas correspondientes al Sprint 3.

TABLA 5.3: PRUEBAS DE ACEPTACIÓN DEL SPRINT 3

Historias de Usuario				
#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
6	Usuario	Comunicarme con otro usuario mediante la herramienta	Compartir dudas o avances sobre el proyecto	<ul style="list-style-type: none"> -El usuario puede acceder al chat de la herramienta. - El usuario invita a otro usuario a iniciar una conversación por chat. - Los usuarios reciben satisfactoriamente los mensajes.
32	Miembro del Equipo	Visualizar Gráficos Burn Down y Burn Up	Poder visualizar el progreso.	<ul style="list-style-type: none"> - En la vista del resumen del proyecto, los miembros del equipo visualizan el burn down chart. - En la vista del resumen del proyecto, el propietario del producto visualiza el burn up chart.

Historias de Usuario

#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
33	Miembro del equipo	Participar en el foro del proyecto	Compartir opiniones o iniciar discusiones acerca del proyecto.	<ul style="list-style-type: none"> -Los miembros del equipo pueden ingresar al foro del proyecto. -Los miembros del equipo pueden iniciar un nuevo tema (<i>topic</i>) en cualquiera de las categorías del foro. -El Scrum Manager puede agregar una nueva categoría al foro. -Los miembros del equipo pueden ingresar respuestas a los comentarios en el foro. -Los miembros del equipo pueden editar o eliminar comentarios en el foro.
9	Scrum Mánager	Agregar roles	Agregar roles personalizados, además de los roles por defecto.	<ul style="list-style-type: none"> -El Scrum Mánager agrega roles al proyecto. -El Scrum Mánager cambia los permisos a los roles. Los cambios se guardan con éxito.
10	Scrum Mánager	Administrar roles	Cambiar los roles de los miembros del equipo en diferentes momentos del proyecto	<ul style="list-style-type: none"> -El Scrum Mánager cambia los roles a los miembros del equipo. Se guardan los cambios con éxito.

Historias de Usuario

#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
17	Miembro del equipo	Agregar tareas a las historias de usuario	Indicar las tareas necesarias para completar cada historia de usuario, así como indicar la persona encargada de realizar dicha tarea	<ul style="list-style-type: none"> - El miembro del equipo agrega una tarea a la historia de usuario. - El miembro del equipo agrega a un responsable de la tarea, en la tarea creada. Los cambios se guardan con éxito. -El miembro del equipo intenta crear una tarea vacía; se muestra un mensaje de error.
18	Miembro del equipo	Agregar comentarios a las historias de usuario	Considerar comentarios adicionales en las historias de usuario.	<ul style="list-style-type: none"> -El miembro del equipo ingresa un comentario con texto a la historia de usuario, se muestra un mensaje satisfactorio. -El miembro del equipo ingresa un comentario vacío. Se muestra un mensaje de error.
34	Miembro del equipo	Mantener actualizada una lista de tareas	Visualizar las tareas pendientes	<ul style="list-style-type: none"> - Cada vez que se le asigna una tarea a un miembro del equipo, se actualiza su lista de tareas pendientes. - El miembro del equipo marca la tarea como realizada. La tarea ya no se muestra en tareas pendientes.

Historias de Usuario				
#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
19	Miembro del equipo	Diferenciar por colores las tareas y los comentarios	Poder resaltar o diferenciar unas de otras.	- El miembro del equipo selecciona el color que se desea agregar a la tarea o comentario; se guardan los cambios.

5.2.4. SPRINT 4

En la Tabla 5.4, se muestran las pruebas de aceptación del cuarto sprint.

TABLA 5.4: PRUEBAS DE ACEPTACIÓN DEL SPRINT 4

Historias de Usuario				
#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
26	Miembro del equipo	Acceder a archivos compartidos en el proyecto.	Contar con un repositorio de archivos comunes al proyecto	<ul style="list-style-type: none"> -El Miembro del equipo crea una nueva carpeta. - Agrega un archivo a la carpeta previamente creada. - El miembro del equipo descarga el archivo agregado.

Historias de Usuario

#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
23	Miembro del equipo	Iniciar reuniones entre dos o más miembros del equipos	Comunicarse y dialogar acerca de un tema específico.	<ul style="list-style-type: none"> - El miembro del equipo crea una nueva reunión con fecha y hora. Además agrega a las personas invitadas. La lista de usuarios está restringida a los miembros del proyecto. - Se inicia una conversación vía chat con los invitados a la reunión.
12	Scrum Mánager	Programar y llevar a cabo el reuniones		<ul style="list-style-type: none"> -El Scrum Mánager programa una reunión diaria con los miembros de los equipos. - Cada miembro indica cómo le va con el proyecto y si tiene algún impedimento.
14	Scrum Mánager	Programar y llevar a cabo la reunión de planificación de la Pila del producto.		<ul style="list-style-type: none"> -El Scrum Mánager y/o Propietario del producto programa una reunión para organizar la pila del producto. - Los cambios realizados en las historias de usuario se guardan con éxito.

Historias de Usuario

#Historia	Como un..	Quiero..	Para ..	Pruebas de Aceptación
13	Scrum Mánager	Programar y llevar a cabo la reunión de planificación del Sprint.		<ul style="list-style-type: none"> -El Scrum Mánager programa una reunión al inicio de cada sprint para planificar la pila del sprint. - Los cambios realizados en las historias de usuario se guardan con éxito.
15	Scrum Mánager	Programar y llevar a cabo la Retrospectiva		<ul style="list-style-type: none"> -El Scrum Mánager programa una reunión de retrospectiva al final del sprint. - Cada miembro indica lo bueno y lo malo que encuentra en el proyecto, Además de las actividades que cree que se deban realizar.

6. CONCLUSIONES Y RECOMENDACIONES

A continuación se describirán las observaciones del proyecto, las conclusiones y las recomendaciones para proyectos de este tipo.

6.1. CONCLUSIONES

Se cumplió con el objetivo principal del proyecto que fue la implementación de una herramienta de soporte a la gestión de proyectos ágiles de desarrollo. Para lograr la culminación con éxito del proyecto de fin de carrera se establecieron iteraciones de desarrollo cuyas fechas se lograron cumplir.

Para tener una visión de la propuesta del proyecto, en un inicio se establecieron prototipos de interfaz de usuario que fueron validadas con el asesor del curso; a partir de estos prototipos se pudieron establecer los ajustes necesario al proyecto para que su alcance se considere como un proyecto de fin de carrera y para que el producto final obtenido pueda considerarse de utilidad para equipos de Scrum distribuido.

En cuanto al cumplimiento de los entregables establecidos, se fijaron reuniones semanales con el asesor del curso al que se le presentaban avances de la

implementación del producto y con el cuál se mantenía un diálogo constante sobre los problemas que se iban presentando y las mejoras que se podían realizar en el producto. Gracias a esta comunicación constante al concluir cada iteración se obtenía un producto potencialmente usable cumpliendo así no solo con el cronograma de entregas, sino además se siguió el marco de trabajo utilizado, Scrum.

6.2. RECOMENDACIONES Y TRABAJOS FUTUROS

Para este tipo de proyectos de desarrollo, que utilicen metodología ágil es importante en primer lugar familiarizarse con los conceptos ágiles y de la metodología a utilizar para poder concluir el proyecto con éxito; ya que las metodologías ágiles al ser adaptativas, si no se cuenta con una rigurosidad en el cumplimiento de los compromisos por parte de los miembros del equipo pueden resultar contraproducentes y llevar al caos.

El presente proyecto puede ser tomado como apertura a la implementación de herramientas de soporte a otras metodologías de desarrollo, o incluso a herramientas de soporte a la gestión de proyectos con escenarios diferentes al desarrollo de software. Por otro lado, se considera que el proyecto puede ser extendido en su implementación para que la herramienta implementada sea apta para su uso en plataformas móviles como Android o iOS.

REFERENCIAS

ABRAHAMSSON, P.; SALO, O.; RONKAIMEN, J. y & WARSTA, J.

2002 "Agile Software development methods. Review and analysis"

VTT Publications 478,pp. página 1-107.

BECK, Kent; BEEDLE, Mike; VAN BENEKUM, Arie; CUNNINGHAM, Ward ;
FOWLER, Martin y Otros

2002 "Agile Manifiesto". Consulta: 01 de Abril de 2012.

<<http://agilemanifesto.org/>>

BOEHM, B. y TURNER, R.

2005 Management Challenges to Implementing Agile Processes in Traditional
Development Organization.

Cake Software Foundation

2012 "The CakePHP cookbook". Consulta 10 de Octubre de 2012

<<http://book.cakephp.org/2.0> >

CHIN, Gary

2004 Agile Project Management: How to succeed in the face of changing project
requirements.

Primera Edición. New York: AMACOM.

ISBN 0-8144-7176-5

DEEMER,Pete; BENEFIELD, Gabrielle; LARMAN, Craig; VODDE,Bas

2010 The Scrum Primer v1.2

DULLEMOND, K. y GAMEREN, B.

2009 "How Technological Support can Enable Advantages of Agile Software
Development in a GSE setting"

Fourth IEEE Internatioanl Conference on Global Software Engineering, pp.
página 143-152).

FOWLER, Martin

2004 "Is Design Dead?". Consulta: 04 de junio de 2012.

<<http://martinfowler.com/articles/designDead.html>>

KONTIO, Yrki; HÖGLUND, Magnus; RYDÉN, Jan y ABRAHAMSSON, Pekka.

2004 "Managing Commitments and Risks: Challenges in Distributed Agile Development".

26th International Conference on Software Engineering (ICSE'04).Scotland, UK.

HERRANZ, Raúl; MAMOGHLI, Noel Mamoghli; YAZYI, Sergio; VERA, José Miguel y Otros

2011 Scrum Distribuido. Consulta : 01 de Abril de 2012

<<http://www.scrummanager.net/ok>>

HOFF, Todd y KRISTIENSEN, Fredrick

2002 PHP Coding Standard

<<http://www.dagbladet.no/development/phpcodingstandard>>

JALALI, Samireh y WOHLIN, Claes

2010 "Agile Practices in Global Software Engineering - A systematic map".

International Conference on Global Software Engineering (ICDSE).New Jersey,USA.

MOORE, Stephanie y BARNETT, Liz

2004 Offshore Outsourcing and Agile Development

Forrester Research

MULDER M. y VAN VLIET, M.

2008 Case study: Distributed Scrum Project for Dutch Railways. Consulta: 01 de Mayo de 2012

<<http://www.infoq.com/articles/dutch-railway-scrum>>

- NIELSEN Jakob
2005 Ten Usability Heuristics
- PALACIO, Juan
2008 Flexibilidad con Scrum: Principios de diseño e implantación de campos de Scrum
- PELRINE, Joseph
2011 On Understanding Software Agility—A Social Complexity Point Of View.
E:CO Vol 13, pp. página 26-37.
- SCRUM ALLIANCE
Scrum Alliance - transforming the world of work. Consulta: 01 de abril de 2012
<<http://www.scrumalliance.org>>
- SUTTHERLAND, Jeff.
2011 The Scrum Papers: Nut, Bolts, and Origins of an Agile Framework
- SUTTHERLAND, J., SCHOONHEIM, G., KUMAR, N., PANDEY V., & VISHAL, S.
2009 "Fully Distributed Scrum: Linear Scalability of production between San Francisco and India".
AGILE '09 Proceedings of the 2009 Agile Conference. Washington DC, USA
.pp. página 277-282
ISBN: 978-0-7695-3768-9
- SUTHERLAND, J., SCHOONHEIM, G., & RIJK, M.
2009 "Fully Distributed Scrum: Replicating Local Productivity and Quality with Offshore Teams.
42nd Hawaii International Conference. Boston, USA.
- REAL ACADEMIA ESPAÑOLA
2001 Diccionario de la Lengua Española. Consulta: 31 de mayo de 2012.
<<http://buscon.rae.es/drae/>>

ROSENBERG, Doug; STEPHENS, Matt y COLLINS-COPE, Mark
2005 Agile Development with ICONIX Process: People, Process, and Pragmatism.

Primera Edición. New York: Apress
ISBN 1-59059-464-9

VERSIONONE

2010 State of Agile Survey 2010. Consulta: 16 de abril de 2012
<http://www.versionone.com/state_of_agile_development_survey/10/>

YAGGAHAVITA,Hasith

2011 Challenges in Applying Scrum Methodology on Culturally Distributed Teams.
Universida Sheffield Hallam, Facultad de Artes, Computación, Ingeniería y Ciencias (ACES).

WOODWARD, Elizabeth; SURDEK, Steffan y GANIS, Matthew

2010 A Practical Guide to Distributed Scum.
Primera Edición. Boston: Pearson Education, Inc.