

ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA DE SISTEMAS

DESARROLLO DE UNA APLICACIÓN WEB PARA VALIDAR REQUERIMIENTOS DE SOFTWARE

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN

LESLIE MISHHELL DÍAZ YANANGÓMEZ

leslie.diaz@epn.edu.ec

PAMELA ABIGAIL PACHACAMA SIMBAÑA

pamela.pachacama@epn.edu.ec

DIRECTORA: MSc. MAYRA DEL CISNE CARRIÓN TORO

mayra.carrion@epn.edu.ec

CODIRECTOR: PhD. MARCO OSWALDO SANTÓRUM GAIBOR

marco.santorum@epn.edu.ec

Quito, julio 2022

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Leslie Mishell Díaz Yanangómez y Pamela Abigail Pachacama Simbaña, bajo mi supervisión.



MSc. Mayra Del Cisne Carrión Toro
DIRECTORA DE PROYECTO

DECLARACIÓN

Nosotros, Leslie Mishell Díaz Yanangómez y Pamela Abigail Pachacama Simbaña , declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.



Leslie Mishell Díaz Yanangómez



Pamela Abigail Pachacama Simbaña

DEDICATORIA

Dedico a Dios mis logros, a mi familia por creer siempre en mí y brindarme su apoyo incondicional, a las personas que ya no están conmigo, pero me dejaron su legado y a todos los que pusieron su mano amiga para que esta meta sea alcanzada.

Pamela

DEDICATORIA

Dedico este trabajo a las personas más importantes en mi vida: mis padres, Carmen y Manuel, quienes sacrificaron su juventud y trabajaron muy duro para criar a sus dos hijas. Su amor incondicional durante todos estos años son el pilar fundamental de quien soy ahora. Aunque no se los diga con frecuencia, estoy muy orgullosa de ser su hija y los amo con todo mi corazón.

A mi hermana, Traecy, por las risas, las peleas y nuestra complicidad. Por estar siempre conmigo y por brindarme su apoyo a lo largo de esta etapa de mi vida.

A mi mascota que en paz descanse, Princesa, quien me acompañó durante casi toda mi carrera universitaria y fue mi fuente de alegría, fuerza y amor incondicional.

Leslie

AGRADECIMIENTO

Agradezco a Dios y a mi familia, que siempre estuvieron apoyándome en cada paso de mi vida. En especial a mi hermana Micaela por demostrarme lo lejos que se puede llegar, a mi hermano Pablo por ser el cómplice de mis aventuras, a mi padre Héctor que sin su apoyo moral y económico no podría continuar con mis estudios, a mi abuelito Gaspar por enseñarme el valor del trabajo, a mi madre por enseñarme el valor de la familia, a mis tíos y tías que fueron mis otras figuras paternas y maternas, y a mis primas y primos que son mis referentes a seguir.

Agradezco a mis amigos, quienes me brindaron su apoyo en los momentos más difíciles, en especial a mi amiga Lilly que es fuente de inspiración y de trabajo duro.

Agradezco a mi directora y codirector de titulación Mayra Carrión y Marco Santórum, por creer en mí y en mi compañera, y por su guía y colaboración en el camino para culminar este proyecto.

Y a todos aquellos que siempre han creído en mí, gracias totales.

Pamela

AGRADECIMIENTO

Agradezco a Dios por permitirme seguir viviendo para culminar este gran logro en mi vida. A mi familia quien estuvo a mi lado brindándome su apoyo incondicional y quienes son mi motivo de superación.

A mi compañera de tesis y amiga Pamela; tu paciencia y confianza fueron el motor para culminar este proyecto. A mi amiga Lilly, quien es mi fuente de inspiración y un ejemplo de lucha y perseverancia.

A mis tutores de titulación, Mayra Carrión y Marco Santórum, por depositar su confianza en nosotras y guiarnos en el desarrollo de esta tesis.

A mi amiga Josselyne, a pesar de la distancia y los años, siempre me escuchó, me tuvo presente y me animó a seguir adelante.

A las personas que conocí en tiempo de pandemia, especialmente David, Ciro y Ali, quienes me ofrecieron su apoyo y me brindaron palabras de ánimo.

A todas esas personas que formaron parte de mi vida, me apoyaron y confiaron en mí. Siempre los tendré en mi memoria.

Y finalmente, pero no menos importante, a todas las mascotas que han sido una parte importante de mi vida, pasada, presente y también las que formarán mi futuro.

Leslie

CONTENIDO

CERTIFICACIÓN	I
DECLARACIÓN	II
ÍNDICE DE FIGURAS	X
ÍNDICE DE TABLAS	XII
RESUMEN	1
ABSTRACT	2
1 INTRODUCCIÓN	3
1.1 Objetivos	4
1.1.1 Objetivo General	4
1.1.2 Objetivos Específicos	4
1.2 Alcance	5
1.3 Marco Conceptual	5
1.3.1 Estudio Conceptual	6
1.3.2 Características de los requerimientos de acuerdo al estudio realizado	22
1.4 Propuesta de Instrumento para el refinamiento de requerimientos	25
1.4.1 Criterios de lenguaje de requerimientos.	26
1.4.2 Atributos de requerimientos.	26
1.5 Resumen del capítulo	27
2 METODOLOGÍA E IMPLEMENTACIÓN	29
2.1 Metodología	29
2.1.1 Equipo de Scrum	30
2.1.2 Valores de Scrum	32
2.1.3 Artefactos y Eventos de Scrum	32
2.2 Fase inicial de Scrum	35
2.2.1 Definición de requerimientos	35
2.2.2 Identificación de roles	36
2.2.3 Historias de usuario	37

2.2.4	Product Backlog	42
2.3	Implementación de Scrum	43
2.3.1	Sprint 0	44
2.3.2	Sprint 1	49
2.3.3	Sprint 2	53
2.3.4	Sprint 3	56
2.4	Fase de lanzamiento	60
2.4.1	Implementación en un ambiente de producción.	60
2.4.2	Retrospectiva del proyecto.	61
2.5	Resumen del capítulo	62
3	APLICATIVO PARA LA VALIDACIÓN DE REQUERIMIENTOS DE SOFTWARE	64
3.1	Aplicativo para la validación de requerimientos de software	64
3.2	Arquitectura, herramientas y recursos utilizados	66
3.2.1	Patrón Modelo-Vista-Controlador (MVC)	66
3.2.2	Arquitectura general de la aplicación	67
3.2.3	Herramientas y recursos utilizados	68
3.3	Caso de estudio	71
3.4	Resumen del capítulo	74
4	EVALUACIÓN Y RESULTADOS	76
4.1	Evaluación de usabilidad con diseño de prototipos	76
4.2	Evaluación de usabilidad con usuario final	78
4.2.1	Primera fase	80
4.2.2	Segunda fase	81
4.3	Resultados	85
4.3.1	Tercera fase	85
4.4	Discusión	87
4.4.1	Cuarta fase	87
4.5	Resumen del capítulo	89
5	CONCLUSIONES, PROSPECTIVAS Y RECOMENDACIONES	90
5.1	Conclusiones	90
5.2	Perspectivas	91
5.3	Recomendaciones	91
6	REFERENCIAS BIBLIOGRÁFICAS	92

ÍNDICE DE FIGURAS

1.1	Estándares y Normas IEEE/ISO/IEC estudiados y analizados.	6
1.2	IEEE 1233 y sus normas de referencia	7
1.3	ISO/IEC/IEEE 29148 y sus normas de referencia	9
1.4	ISO/IEC 25000 y sus normas de referencia	12
1.5	ISO/IEC TR 24766 y sus normas de referencia	14
1.6	Modelo para la calidad interna y externa ISO/IEC 9126-1.	16
1.7	Resultado del proceso de análisis de las propiedades de requerimientos . . .	24
1.8	Plantilla para propósitos - cliente	26
1.9	Plantilla para ideas GamePlay	26
2.1	Esquema de trabajo de SCRUM [31]	30
2.2	Estructura completa de la base de datos	45
2.4	Estructura de usuario y proyecto	46
2.3	Estructura de requerimientos.	46
2.5	Estructura de resultado.	47
2.6	Método gráfico - requerimiento cliente.	47
2.7	Método gráfico - requerimiento GamePlay.	48
2.8	Burndown chart sprint 0	49
2.9	Pantalla de registro de usuario	51
2.10	Pantalla de inicio de sesión	51
2.11	Pantalla de administración de proyectos	52
2.12	Burndown chart sprint 1	53
2.13	Pantalla de subida de archivo plantilla	55
2.14	Pantalla de refinamiento de requerimientos	55
2.15	Burndown chart sprint 2	56
2.16	Pantalla de administración de participantes	58
2.17	Pantalla de resultados del refinamiento de requerimientos	58
2.18	Burndown chart sprint 3	59
3.1	Relación entre los módulos del patrón MVC	67
3.2	Arquitectura general de la aplicación	68

3.3	Registro de usuario.	71
3.4	Interfaz inicial.	72
3.5	Plantilla requerimientos cliente.	72
3.6	Plantilla GamePlay.	73
3.7	Refinamiento de requerimientos	73
3.8	Resultados de refinamiento	74
4.1	Evidencias del primer prototipo de la aplicación	76
4.2	Evidencias del segundo prototipo de la aplicación	77
4.3	Evidencias del tercer prototipo de la aplicación - parte 1	77
4.4	Evidencias del tercer prototipo de la aplicación - parte 2	78
4.5	Evidencias del cuarto prototipo de la aplicación	78
4.6	Protocolo de evaluación de usabilidad.	79
4.7	Reunión de planificación-indicaciones generales.	81
4.8	Reunión de planificación-envió de documentación.	81
4.9	Promedio porcentual por pregunta, encuesta CSUQ	86
4.10	Promedio por escala	86
4.11	Resultados pregunta 1 - CSUQ.	87
4.12	Resultados pregunta 16 - CSUQ.	88
4.13	Resultados pregunta 5 - CSUQ.	88
4.14	Resultados pregunta 7 - CSUQ.	89

ÍNDICE DE TABLAS

1.1	Características de Requerimientos	23
1.2	Clasificación de las propiedades de requerimientos de acuerdo a frecuencia de términos clave	24
1.4	Formato de ingreso de requerimiento.	27
2.1	Historia de usuario épica 1.	36
2.2	Historia de usuario épica 2.	36
2.3	Historia de usuario épica 3.	36
2.4	Roles de Scrum	37
2.5	Historia de Usuario HU01.	37
2.6	Historia de Usuario HU02.	38
2.7	Historia de Usuario HU03.	38
2.8	Historia de Usuario HU04.	39
2.9	Historia de Usuario HU05.	40
2.10	Historia de Usuario HU06.	40
2.11	Historia de Usuario HU07.	41
2.12	Historia de Usuario HU08.	41
2.13	Historia de Usuario HU09.	42
2.14	Historia de Usuario HU10.	42
2.15	Product Backlog inicial	43
2.16	Product Backlog final	43
2.17	Sprint 0 Backlog	44
2.18	Sprint 0 Review	49
2.19	Sprint 1 Backlog	50
2.20	Sprint 1 Review	53
2.21	Sprint 2 Backlog	54
2.22	Sprint 2 Review	56
2.23	Sprint 3 Backlog	57
2.24	Sprint 3 Review	59
2.25	Retrospectiva del proyecto	62

3.1	Herramientas y Recursos	71
4.1	Tareas y subtareas	84
4.2	Cuestionario para evaluar la usabilidad	85

RESUMEN

Este trabajo de titulación presenta una solución de software para la validación de requerimientos, ya que este proceso es una parte esencial en el desarrollo de software [1]. El trabajo está enfocado tanto para proyectos de software comunes como para proyectos dirigidos al desarrollo de juegos serios siendo la base de este, la automatización de la fase de refinamiento de la metodología iPlus propuesta en [2]. Se realizó una revisión documental de las características que debe cumplir un requerimiento para ser considerado bien formado según las normas y estándares establecidos en ISO, IEC e IEEE. El desarrollo de la aplicación siguió el marco de enfoque ágil SCRUM y se utilizó frameworks basados en JavaScript para un mejor acoplamiento de las partes de servidor y cliente. En cuanto a su despliegue se utilizaron los servicios de AWS tanto para el servidor como para la base de datos. Para evaluar la usabilidad del aplicativo web se realizaron pruebas de usabilidad contando con la participación de 6 ingenieros de distintas universidades cuyas áreas de trabajo se relacionan al tema de investigación. El resultado obtenido fue satisfactorio y es concluyente a que la aplicación desarrollada tiene un alto grado de usabilidad consiguiendo 91,5% en el porcentaje global de todas las preguntas planteadas. El objetivo de este proyecto fue desarrollar una aplicación web para la validación de requerimientos de software mediante un enfoque de desarrollo ágil el cual fue cumplido en su totalidad.

PALABRAS CLAVE: Aplicación web, SCRUM, estándar, norma, ISO, IEC, IEEE, enfoque ágil, requerimiento, requerimiento bien formado, iPlus, revisiones documentales

ABSTRACT

This degree work presents a software solution for requirements validation, this process is an essential part of software development [1]. The work is focused on common software projects and those ones aimed at the development of serious games, the basis of which is the automation of the refinement phase of the iPlus methodology proposed in [2]. A documentary review of the characteristics that a requirement must meet to be considered well-formed according to the norms and standards established in ISO, IEC and IEEE was carried out. The development of the application followed the SCRUM agile approach framework and JavaScript-based frameworks were used for a better coupling of the server and client parts. As for its deployment, AWS services were used for the server and the database. To evaluate the usability of the web application, usability tests were carried out with the participation of 6 engineers from different universities whose work areas are related to the research topic. The result obtained was satisfactory and it is conclusive that the developed application has a high degree of usability, achieving 91.5% in the overall percentage of all the questions raised. The objective of this project was to develop a web application for the validation of software requirements through an agile development approach which was fulfilled in its entirety.

KEY WORDS: Web application, SCRUM, standard, norm, ISO/IEC/IEEE, agile approach, requirement, well-formed requirement, iPlus, document reviews

1 INTRODUCCIÓN

Las tecnologías de la información y el Internet como medio de difusión han facilitado la comunicación y servicios a través de Internet. El desarrollo del software se encuentra en auge y constituye una herramienta de apoyo importante para la ejecución de procesos organizacionales [3].

Las metodologías de desarrollo de software guían el proceso de desarrollo del mismo, sin embargo, muchas veces se presentan problemas en cada etapa de desarrollo. De acuerdo con Davey et al. en [4], un proceso de desarrollo de software se torna deficiente debido a que la obtención de requisitos rara vez se hace bien, y una comprensión inexacta o incompleta de los requisitos del usuario ha llevado a la caída de muchos proyectos de software.

Algunos de los elementos que pueden influir en el fracaso de una implementación y que son detectables durante el desarrollo del proyecto son: el incumplimiento de plazos, desviación del alcance original del proyecto entre otros [5]. Según el reporte CHAOS realizado en el 2009 por The Standish Group citado en [6], se muestra que el 12.30 % de proyectos fracasan por requisitos y especificaciones incompletos y un 11.80 % por requisitos cambiantes. El levantamiento de requerimientos es una parte esencial en el proceso de desarrollo de software. De este proceso dependerá en cierta medida si se cumplen o no los objetivos del proyecto dado que el costo o los riesgos que involucra una mala definición de requisitos son grandes y a veces irreparables [1].

Sería de utilidad para cualquier metodología enmarcada en el desarrollo de software tener un instrumento que permita conocer si los requisitos recolectados cumplen con los estándares mínimos que se establecen en las normas ISO/IEC/IEEE dedicadas ello. Además, existen diversos procesos de levantamiento y formulación de requerimientos, sin embargo, en algunas ocasiones, los requerimientos no se formalizan, no se validan, resultan no realizables o no viables [3], [7].

Para evitar los problemas descritos con anterioridad, se debe validar todos los requerimien-

tos obtenidos sin importar la metodología usada, siguiendo parámetros de calidad, alcance, entre otros. Estos parámetros son establecidos por normas de especificación de requerimientos por ejemplo las normas y estándares IEEE 830, IEEE 1233, ISO 29148, ISO 12207, ISO/IEC 9126, ISO/IEC 14598, ISO 15288, ISO 24766, ISO/IEC 25000, ISO 9000, ISO 15504 [8], [9], [10], [11].

Los procesos de requisitos deficientes son responsables de los desafíos y fallas de los proyectos de software, mientras que, por otro lado, una buena recopilación de requisitos contribuye al éxito de los proyectos de software [1]. Una identificación rápida de problemas en la formulación de requerimientos evitará que se realice trabajo en vano o que no cumpla los objetivos de este.

Como respuesta a la problemática antes planteada, proponemos el desarrollo de una herramienta web para la validación de requerimientos de software, acorde a sus características propias y las propiedades establecidas por las normas de especificación de requerimientos citadas.

1.1 OBJETIVOS

1.1.1 Objetivo General

Desarrollar una aplicación web para la validación de requerimientos de software mediante un enfoque de desarrollo ágil.

1.1.2 Objetivos Específicos

1. Estudiar los marcos de referencia de los estándares, normas, metodologías y técnicas para la formulación de requerimientos de software.
2. Proponer un instrumento para la validación de requerimientos de software implementado a través de una aplicación web.
3. Desarrollar una aplicación web mediante el marco de trabajo ágil como SCRUM.
4. Evaluar la aplicación en cuanto a usabilidad mediante un caso de estudio.
5. Implementar la aplicación en un ambiente de producción.

1.2 ALCANCE

El proyecto contempla:

- ❖ Un estudio conceptual acerca de las características necesarias para que un requerimiento sea válido acorde a las normas y estándares descritos en el marco conceptual.
- ❖ El desarrollo de una aplicación web que valide requerimientos de software, la misma que será de utilidad para los equipos de desarrollo de software de cualquier proyecto.
- ❖ Un enfoque ágil para el desarrollo del aplicativo empleando la metodología Scrum, el cual es un marco de trabajo ligero que entrega productos de mayor valor.
- ❖ Un marco de referencia como base para la construcción de un caso práctico.
- ❖ Una fase de lanzamiento en línea por medio de un enlace temporal, para cualquier persona que desee hacer uso del aplicativo.

El proyecto no contempla:

- ❖ En la fase de lanzamiento con un enlace permanente, por lo que el enlace proporcionado en este proyecto solo será vigente por 6 meses. Se contempla un dominio otorgado por la Escuela Politécnica Nacional a futuro.
- ❖ En el aplicativo web no existe una metodología que sea automatizada, por lo que siempre se realiza en base a la opinión de un experto. Se contempla que para trabajos futuros se implemente esta metodología con inteligencia artificial.

1.3 MARCO CONCEPTUAL

El objetivo de este apartado es presentar los resultados del estudio conceptual, el cual fue desarrollado mediante la metodología propuesta por Morales [12] para establecer un marco referencial de las normas IEEE/ISO/IEC relacionadas a la formulación de requerimientos de software, obteniendo como resultado las características que deben cumplir dichos requerimientos. En la figura 1.1 se encuentran las normas y estándares que se estudiaron, mostrando su relación general.

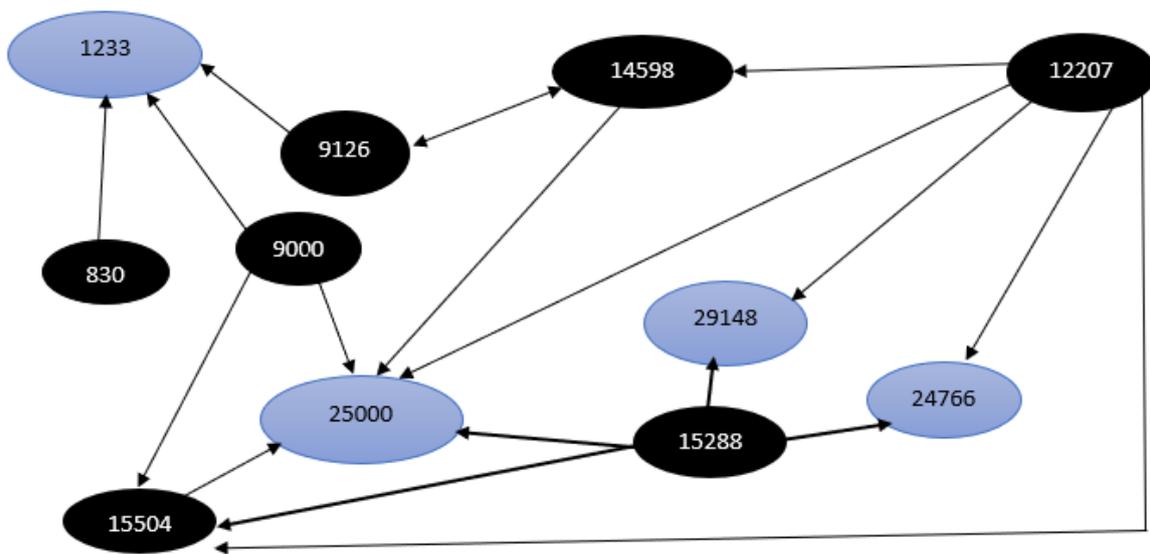


Figura 1.1: Estándares y Normas IEEE/ISO/IEC estudiados y analizados.

Por otro lado, también se plantea realizar la propuesta y análisis del instrumento para la validación de requerimientos de software. El instrumento proporcionado por la MSc. Mayra Carrión [13], perteneciente a la última fase de la metodología iPlus contiene una matriz de refinamiento con características claves que deben cumplir los requerimientos de software.

1.3.1 Estudio Conceptual

El estudio conceptual se realizó mediante la propuesta metodológica de Morales [12], el cual permite seguir un proceso estructural de investigación, acopio, organización, análisis e interpretación de información.

Los pasos propuestos por Morales fueron rediseñados para nuestro estudio de marcos referenciales y son los siguientes:

- ❖ **Acopio de información o de fuentes de información.** En este paso se realizó la recopilación de normas IEEE/ISO/IEC, metodologías y técnicas relacionadas a la formulación de requerimientos de software.
- ❖ **Organización de los datos.** Este paso se llevó a cabo mediante la ayuda de la herramienta Mendeley para la organización de los artículos.
- ❖ **Análisis de los datos.** En este paso se analiza e interpreta los documentos recolec-

tados.

1.3.1.1 Estándar

Un estándar es una serie de recomendaciones, modelos, referencias a seguir para la entrega de un producto, con el fin de que los entregables cumplan con las expectativas del negocio [14].

Dentro de los estándares más aplicados para los requisitos de software se encuentran los siguientes: IEEE 830, IEEE 1233, ISO 29148, ISO 12207, ISO/IEC 9126, ISO/IEC 14598, ISO 15288, ISO 9000, ISO 24766, ISO/IEC 25000, ISO 15504, los cuales serán descritos a continuación.

❖ IEEE 1233 [14]

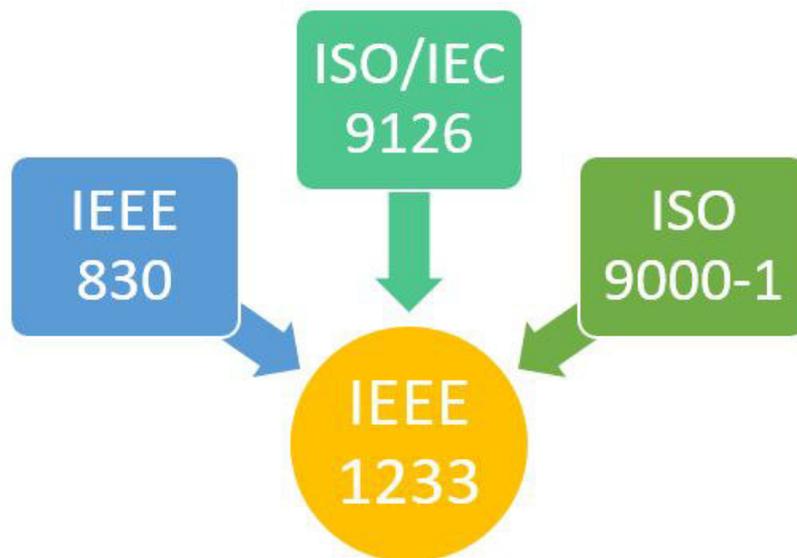


Figura 1.2: IEEE 1233 y sus normas de referencia

IEEE Guide for Developing System Requirements Specifications. Esta guía proporciona orientación para el desarrollo de un conjunto de requisitos que, cuando se cumplan, satisfarán una necesidad expresada. En esta guía, ese conjunto de requisitos se denominará Especificación de requisitos del sistema (SyRS, por sus siglas en inglés). El desarrollo de un SyRS incluye la identificación, organización, presentación y modificación de los requisitos. La colección de requisitos debe tener las siguientes

propiedades:

- a. **Conjunto único.** Cada requisito debe indicarse una sola vez
- b. **Normalizado.** Los requisitos no deben superponerse (es decir, no deben hacer referencia a otros requisitos o a las capacidades de otros requisitos).
- c. **Conjunto vinculado.** Deben definirse relaciones explícitas entre los requisitos individuales para mostrar cómo se relacionan los requisitos para formar un sistema completo.
- d. **Completo** Un SyRS debe incluir todos los requisitos identificados por el cliente, así como los necesarios para la definición del sistema.
- e. **Consistente.** El contenido de SyRS debe ser coherente y no contradictorio en el nivel de detalle, el estilo de las declaraciones de requisitos y la presentación del material.
- f. **Acotado.** Deben identificarse los límites, el alcance y el contexto del conjunto de requisitos.
- g. **Modificable.** El SyRS debe ser modificable. Los requisitos de claridad y no superposición contribuyen a esto.
- h. **Configurable** Las versiones deben mantenerse a lo largo del tiempo y en todas las instancias de SyRS.
- i. **Granulado.** Este debería ser el nivel de abstracción del sistema que se está definiendo.

Un requisito bien formado es una declaración de la funcionalidad del sistema que se puede validar, que un sistema debe cumplir o poseer para resolver un problema del cliente o para lograr un objetivo del cliente, esto se divide en:

- ❖ **Capacidades.** Son los requisitos fundamentales del sistema y representan las características o funciones del sistema que necesita o desea el cliente. Por lo general, una capacidad debe establecerse de tal manera que describa lo que debe hacer el sistema y de modo que sea independiente de la solución. Esto permitirá la consideración de diferentes formas de satisfacer la necesidad o de proporcionar la característica o función.
- ❖ **Condiciones.** Son atributos y características cualitativas o cuantitativas medibles que se estipulan para una capacidad. Califican además una capacidad que se

necesita y proporcionan atributos que permiten formular y declarar una capacidad de manera que pueda validarse y verificarse.

- ❖ **Restricciones.** Son requisitos que se imponen a la solución por las circunstancias, la fuerza o la compulsión. Las restricciones limitan absolutamente las opciones abiertas al diseñador de una solución al imponer fronteras y límites inamovibles.

Cada requisito debe poseer las siguientes propiedades:

- Resumen.** Cada requisito debe ser independiente de la implementación.
- Sin ambigüedades.** Cada requisito debe establecerse de tal manera que pueda interpretarse de una sola manera.
- Trazable.** Cada requisito debe ser factible determinar una relación entre las declaraciones documentadas específicas de necesidad del cliente y las declaraciones específicas en la definición del sistema dada en el SyRS como evidencia de la fuente de un requisito.
- Validable.** Cada requisito debe tener los medios para probar que el sistema satisface los requisitos.

❖ ISO/IEC/IEEE 29148 [15]

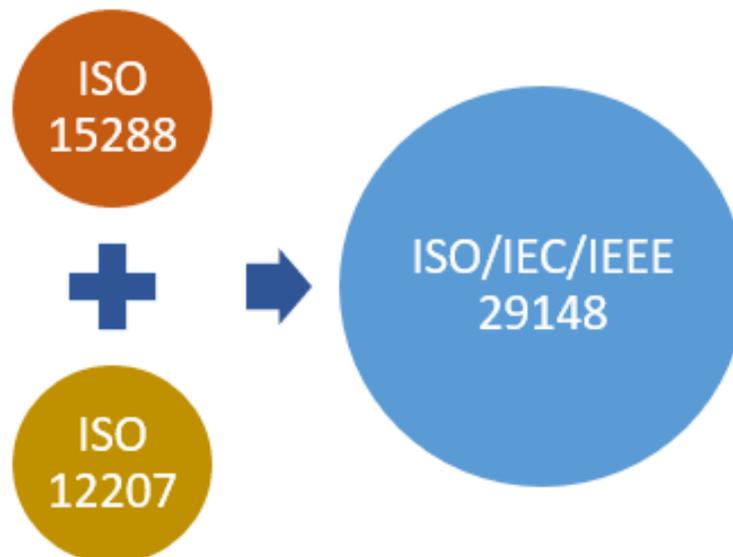


Figura 1.3: ISO/IEC/IEEE 29148 y sus normas de referencia

ring: Contiene destrezas para los procesos y productos relacionados con la ingeniería de requisitos para los sistemas y productos de software y servicios a lo largo del ciclo de vida. Define la construcción de un buen requisito, proporcionando atributos y características de los mismos, y analiza la aplicación iterativa y recursiva de los procesos de requisitos a lo largo del ciclo de vida.

La construcción de requisitos bien formados debe desarrollarse de las partes interesadas, requisitos del sistema y requisitos de los elementos del sistema. Esta práctica ayuda a garantizar que los requisitos capturen con precisión las necesidades de las partes interesadas. Cada requisito de parte interesada, sistema y elemento del sistema deberá poseer las siguientes características:

- ✧ **Necesario.** El requisito define una capacidad esencial, característica, restricción y/o factor de calidad. Si no se incluye en el conjunto de requisitos, existirá una deficiencia en la capacidad o característica, que no se puede cumplir con la implementación de otros requisitos.
- ✧ **Apropiado.** La intención específica y la cantidad de detalles del requisito son apropiados para el nivel de la entidad a la que se refiere (nivel de abstracción apropiado para el nivel de la entidad).
- ✧ **Sin ambigüedad.** El requisito se establece de tal manera que puede interpretarse de una sola manera, de forma simple y fácil de entender.
- ✧ **Completo.** El requerimiento describe suficientemente la capacidad, característica, restricción o factor de calidad necesarios para satisfacer la necesidad de la entidad sin obligación de otra información para comprender el requerimiento.
- ✧ **Singular.** El requisito establece una sola capacidad, característica, restricción o factor de calidad.
- ✧ **Factible.** El requisito se puede realizar dentro de las restricciones del sistema (por ejemplo, costo, cronograma, técnico) con un riesgo aceptable.
- ✧ **Verificable.** El requisito está estructurado y redactado de manera que su realización pueda probarse (verificarse) a satisfacción del cliente en el nivel en que existen los requisitos.
- ✧ **Correcto.** El requisito es una representación precisa de la necesidad de la entidad a partir de la cual se transformó.

- ❖ **Conforme.** Los elementos individuales se ajustan a una plantilla y un estilo estándar aprobados para los requisitos de escritura, cuando corresponda.

Por otra parte, hay ciertas características que se consideran para el conjunto de requisitos de partes interesadas, sistemas y elementos del sistema en lugar de cualquier requisito individual. Estos conjuntos de requisitos brindan una solución consistente que cumple con las intenciones y restricciones de las partes interesadas. Cada conjunto de requisitos para un sistema, software o servicio deberá poseer las siguientes características:

- ❖ **Completo.** El conjunto de requisitos es independiente y describe suficientemente las capacidades, características, restricciones o factores de calidad necesarios para satisfacer las necesidades de la entidad sin necesidad de más información.
- ❖ **Consistente.** El conjunto de requisitos contiene requisitos individuales que son únicos, no entran en conflicto ni se superponen con otros requisitos del conjunto, y las unidades y los sistemas de medición son homogéneos.
- ❖ **Factible.** El conjunto completo de requisitos se puede realizar dentro de las restricciones de la entidad (por ejemplo, costo, cronograma, técnico) con un riesgo aceptable.
- ❖ **Comprensible.** El conjunto de requisitos está redactado de forma que quede claro lo que espera la entidad y su relación con el sistema del que forma parte.
- ❖ **Capaz de ser validado.** Es factible que la satisfacción del conjunto de requisitos conduzca al logro de las necesidades de la entidad dentro de las limitaciones (por ejemplo, costo, cronograma, cumplimiento técnico, legal y regulatorio).

❖ **ISO/IEC 25000 (SQuaRE) [16]**

SQuaRE, del inglés Software Quality Requirements and Evaluation: son una serie de normas que se basan en ISO 9126 e ISO 14598 (evaluación del software); cuenta con los siguientes pasos: recopilar los datos, preparación de los datos y análisis de los datos.

SQuaRE está formada por las divisiones siguientes:

1. ISO/IEC 2500n. División de gestión de calidad. Los estándares que forman esta división definen todos los modelos comunes, términos y referencias a los que se alude en las demás divisiones de SQuaRE.

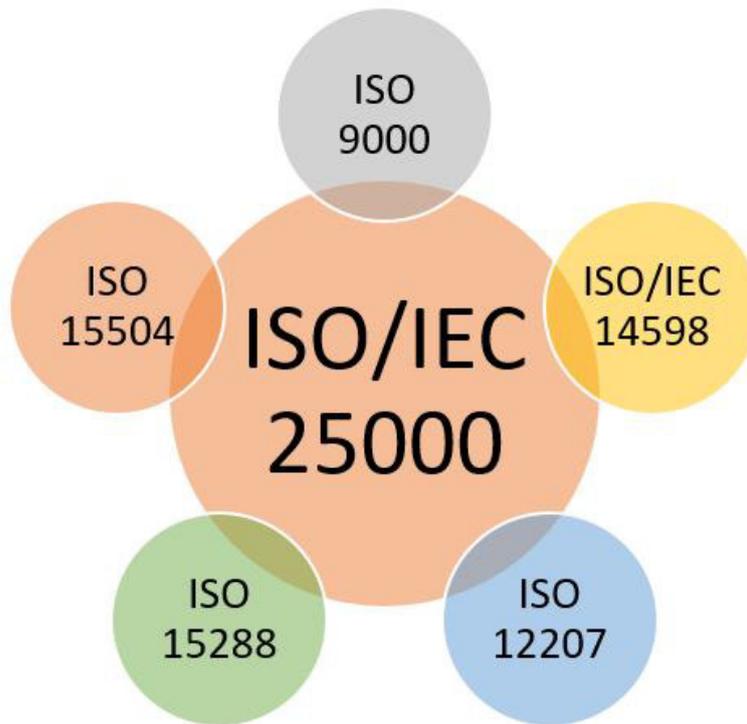


Figura 1.4: ISO/IEC 25000 y sus normas de referencia

2. ISO/IEC 2501n. División del modelo de calidad. El estándar que conforma esta división presenta un modelo de calidad detallado, incluyendo características para la calidad interna, externa y en uso.
3. ISO/IEC 2502n. División de mediciones de calidad. Los estándares pertenecientes a esta división incluyen un modelo de referencia de calidad del producto software, definiciones matemáticas de las métricas de calidad y una guía práctica para su aplicación.
4. ISO/IEC 2503n. División de requisitos de calidad. Los estándares que forman parte de esta división ayudan a especificar los requisitos de calidad. Estos requisitos pueden ser usados en el proceso de especificación de requisitos de calidad para un producto software que va a ser desarrollado o como entrada para un proceso de evaluación. El proceso de definición de requisitos se guía por el establecido en la norma ISO/IEC 15288 (ISO, 2003).
5. ISO/IEC 2504n. División de evaluación de la calidad. Estos estándares proporcionan requisitos, recomendaciones y guías para la evaluación de un producto software, tanto si la llevan a cabo evaluadores, como clientes o desarrolladores.
6. ISO/IEC 25050–25099. Estándares de extensión SQuaRE. Incluyen requisitos para la calidad de productos de software “Off-The-Self” y para el formato común

de la industria (CIF) para informes de usabilidad.

❖ ISO/IEC 25030 [17]

La calidad del sistema es el resultado de la calidad de los elementos del sistema y sus interacciones. Este estándar se enfoca en la calidad del software como parte del sistema. La calidad del software es la capacidad de un producto de software para satisfacer sus necesidades explícitas e implícitas cuando se usa bajo ciertas condiciones. El modelo de calidad de productos de software especificado define las siguientes seis características de calidad:

- **Funcionalidad:** la capacidad de un producto de software para proporcionar una funcionalidad que satisfaga sus necesidades explícitas e implícitas cuando el software se utiliza en condiciones específicas.
- **Confiabilidad:** La capacidad de un producto de software para mantener un nivel específico de logro cuando se usa bajo condiciones específicas.
- **Usabilidad:** La capacidad de un producto de software para ser comprensible, dominado, usable y atractivo para el usuario cuando se usa bajo condiciones específicas.
- **Eficiencia:** La capacidad de un producto de software para proporcionar un rendimiento adecuado en relación con la cantidad de recursos utilizados en condiciones específicas.
- **Mantenibilidad:** La capacidad del producto de software para ser fácil de modificar. Las modificaciones pueden incluir correcciones de software, mejoras o adaptaciones a cambios en el entorno y requisitos y especificaciones funcionales.
- **Portabilidad:** La capacidad de un producto de software para moverse de un entorno a otro.

Este estándar define características adicionales de calidad a nivel del sistema.

❖ ISO/IEC TR 24766 [18]



Figura 1.5: ISO/IEC TR 24766 y sus normas de referencia

La norma ISO/IEC 24766 Tecnología de la información - Ingeniería de sistemas y software - Guía para las capacidades de la herramienta de requisitos proporciona una orientación sobre las capacidades deseables que debería aportar una herramienta de Ingeniería de Requisitos. La ingeniería de requisitos es un proceso esencial de los sistemas y los ciclos de vida del software de ingeniería, por lo cual establecer un criterio de calidad objetivo (o características de calidad) puede ayudar en la detección temprana y corrección de problemas. Las características de calidad también se pueden utilizar como métrica para evaluar los requisitos.

Las características de calidad asociadas para las declaraciones de requisitos individuales se indican de la siguiente manera:

- ❖ **Singularidad.** Cada declaración de requisitos aborda un solo problema con una granularidad acordada.
- ❖ **Corrección.** La declaración de requisitos corresponde a la solicitud del usuario.
- ❖ **Concisión.** Significados breves, sin información innecesaria.
- ❖ **Sin ambigüedad.** La declaración de requisitos tiene una sola interpretación.
- ❖ **Trazabilidad.** La declaración de requisitos es rastreable hacia arriba y hacia abajo hasta los artefactos circundantes.
- ❖ **Cumplimiento.** Ya sea que el requisito sea obligatorio o no
- ❖ **Prioridad.** Se determina la urgencia del requerimiento.
- ❖ **Verificabilidad.** La declaración de requisitos es comprobable.
- ❖ **Abstracción.** La declaración de requisitos se expresa con el nivel de detalle adecuado.
- ❖ **Viabilidad.** La declaración de requisitos es técnicamente factible.

- ❖ **Estabilidad.** La declaración de requisitos no está sujeta recientemente a cambios frecuentes.
- ❖ **Suficiencia.** La declaración de requisitos está suficientemente cubierta en cada dimensión hacia abajo es decir rastreabilidad.
- ❖ **Necesidad.** La declaración de requisitos es necesaria en cada dimensión de trazabilidad ascendente en la que compromete,

Las características de calidad asociadas para conjuntos de declaraciones de requisitos se denotan de la siguiente manera:

- ❖ **Coherencia.** Es la ausencia de conflicto entre un conjunto de requisitos.
- ❖ **Uniformidad.** Es el uso coherente de los términos en un conjunto de requisitos.
- ❖ **Modularidad.** Es la cohesión entre un conjunto de requisitos.
- ❖ **Alcance.** Es un conjunto de requisitos dentro del alcance definido.
- ❖ **Unicidad.** Todas las declaraciones de requisitos en el conjunto son significativamente diferentes.
- ❖ **Compleitud.** Es el conjunto de requisitos que representan una definición completa del producto.
- ❖ **Comprensibilidad.** El conjunto de requisitos es comprensible como un documento.
- ❖ **Asequibilidad.** El conjunto de requisitos es alcanzable con respecto al cronograma y el costo.

❖ **IEEE 830** [19]

IEEE Recommended Practice for Software Requirements Specifications- permite la coherencia en la especificación de requisitos y ayuda a no dejar cabos sueltos. Según el estándar IEEE 830 una Especificación de Requisitos de Software (SRS, por sus siglas en inglés) debe cumplir las siguientes características:

- ❖ Correcto
- ❖ Inequívoco
- ❖ Completo
- ❖ Consistente

- ❖ Clasificado por importancia y/o estabilidad
- ❖ Verificable
- ❖ Modificable
- ❖ Trazable

❖ **ISO/IEC 9126 [20]**

Es la norma para evaluar los productos de software, la norma fue diseñada en los siguientes factores: calidad de proceso, calidad del producto, calidad del software y calidad de uso. Presenta dos modelos de calidad: modelo de calidad interna y externa y modelo referido a la calidad de uso.

La calidad externa de la norma ISO/IEC 9126-1, permite especificar y calificar la calidad de un software desde diferentes puntos de vista, los cuales se relacionan con los requerimientos, uso, adquisición, desarrollo, auditoría de software y mantenimiento [16].

El modelo de calidad para calidad interna y externa de acuerdo a la ISO/IEC 9126-1, categoriza los atributos de calidad de software en seis características, que a su vez se dividen en subcaracterísticas, estos se muestran en la Figura 1.6:

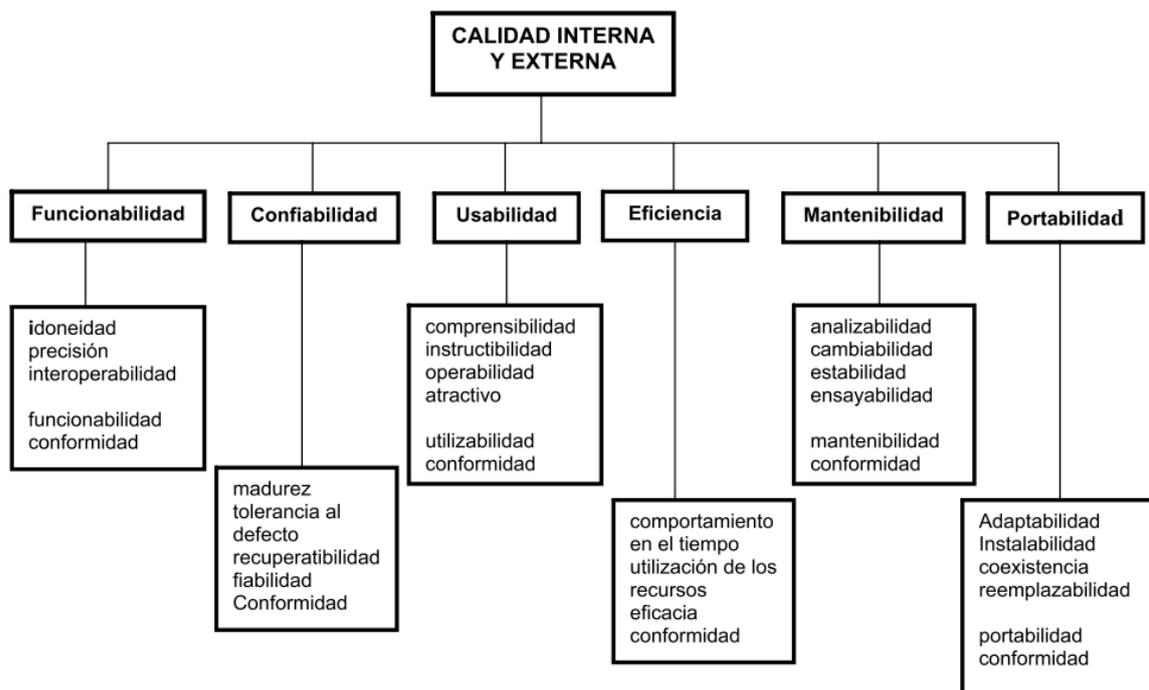


Figura 1.6: Modelo para la calidad interna y externa ISO/IEC 9126-1.

❖ **ISO 9000** [21]

Las Normas ISO 9000 son un conjunto de enunciados que especifican los elementos que se deben integrar en el Sistema de Gestión de Calidad de una organización y como deben funcionar en conjunto para asegurar la calidad de los bienes y servicios que produce la organización.

- ❖ **ISO 9000** Sistemas de Gestión de la Calidad - Definiciones y Fundamentos. Hace referencia a los términos fundamentales utilizados en la familia de normas ISO 9000.
- ❖ **ISO 9001** Sistema de Gestión de la Calidad - Requisitos. Establece los requisitos mínimos que debe cumplir un sistema de Gestión de la Calidad, se puede utilizar para su aplicación interna.
- ❖ **ISO 9004** Sistemas de Gestión de la Calidad - Directrices para la mejora del desempeño: Proporciona ayuda para la mejora del sistema de gestión de la calidad para beneficiar a todas las partes interesadas a través del mantenimiento de la satisfacción del cliente.
- ❖ **ISO 19011** Directrices para la auditoría ambiental y de la calidad: Proporciona directrices para verificar la capacidad del sistema para conseguir objetivos de la calidad definidos.

❖ **ISO/IEC 12207** [22]

Modelos de Ciclos de Vida del Software - Es el estándar para los procesos de ciclo de vida del software de la organización. Este estándar se concibió para aquellos interesados en adquisición de software, así como desarrolladores y proveedores. El estándar indica una serie de procesos desde la recopilación de requisitos hasta la culminación del software.

El propósito del proceso de definición de requisitos de sistema/software de la ISO/IEC 12207 es transformar la vista de las capacidades deseadas de las partes interesadas y orientada al usuario en una vista técnica de una solución que satisfaga las necesidades operativas del usuario. Para definir los requisitos del sistema/software se debe realizar las siguientes tareas:

1. Definir cada función que se requiere que realice el sistema o elemento de software.

2. Identificar los estados requeridos o modos de operación del sistema de software.
3. Definir las restricciones de implementación necesarias.
4. Identificar los requisitos relacionados con los riesgos, la criticidad del sistema de software o las características críticas de calidad
5. Definir los requisitos del sistema/software y los atributos de los requisitos.

❖ **ISO/IEC 14598 [23]**

El estándar ISO/IEC 14598 es actualmente usado como base metodológica para la evaluación del producto software. En sus diferentes etapas, establece un marco de trabajo para evaluar la calidad de los productos de software proporcionando, además, métricas y requisitos para los procesos de evaluación de estos. El mismo consta de seis partes:

1. ISO/IEC 14598-1 Visión General: provee una visión general de las otras cinco partes y explica la relación entre la evaluación del producto software y el modelo de calidad definido en la ISO/IEC 9126.
2. ISO/IEC 14598-2 Planeamiento y Gestión: contiene requisitos y guías para las funciones de soporte tales como la planificación y gestión de la evaluación del producto del software.
3. ISO/IEC 14598-3 Proceso para desarrolladores: provee los requisitos y guías para la evaluación del producto software cuando la evaluación es llevada a cabo en paralelo con el desarrollo por parte del desarrollador.
4. ISO/IEC 14598-4 Proceso para adquirentes: provee los requisitos y guías para que la evaluación del producto software sea llevada a cabo en función a los compradores que planean adquirir o reutilizar un producto de software existente o pre-desarrollado.
5. ISO/IEC 14598-5 Proceso para avaladores: provee los requisitos y guías para la evaluación del producto software cuando la evaluación es llevada a cabo por evaluadores independientes.
6. ISO/IEC 14598-6 Documentación de Módulos: provee las guías para la documentación del módulo de evaluación.

❖ **ISO/IEC 15288 [24]**

El estándar ISO/IEC 15288 Ingeniería de Sistemas y Software: procesos del ciclo de vida del sistema establece un marco común de procesos para describir el ciclo de vida de la Ingeniería de Sistemas. Define un conjunto de procesos y la terminología asociada desde un punto de vista de la ingeniería. Estos procesos se pueden aplicar en cualquier nivel de la estructura jerárquica de un sistema. Esto se logra a través de la participación de todas las partes interesadas, con el objetivo de lograr la satisfacción del cliente.

El proceso que sigue la ISO/IEC 15288 de definición de requisitos del sistema tiene como propósito transformar la vista de las partes interesadas y orientada al usuario de las capacidades deseadas en una vista técnica de una solución que satisfaga las necesidades operativas del usuario.

Este proceso crea un conjunto de requisitos del sistema medibles que especifican, desde la perspectiva del proveedor, qué características, atributos y requisitos funcionales y de rendimiento debe poseer el sistema para satisfacer los requisitos de las partes interesadas, este proceso tiene como resultado:

- a. La descripción del sistema, incluidas las interfaces del sistema, las funciones y los límites, para una solución del sistema.
- b. Los requisitos del sistema (funcionales, de rendimiento, de proceso, no funcionales y de interfaz) y las restricciones de diseño.
- c. medidas críticas de desempeño.
- d. Se analizan los requisitos del sistema.
- e. Están disponibles todos los sistemas o servicios de habilitación necesarios para la definición de los requisitos del sistema.
- f. Se desarrolla la trazabilidad de los requisitos del sistema a los requisitos de las partes interesadas.

❖ **ISO/IEC 15504 (SPICE) [25]**

SPICE “es un estándar internacional que es aplicable a cualquier organización que quiera conocer y mejorar la capacidad de sus procesos”. Además, SPICE “No pretende fijar la manera de realizar los procesos dentro de la organización, sino que valora su capacidad y ayuda a proponer mejoras que aumenten esta capacidad”.

Este estándar se divide en cinco partes las cuales comprenden diferentes aspectos para la evaluación de procesos.

- ❖ Parte 1: Conceptos y vocabulario
- ❖ Parte 2: Realización de una evaluación
- ❖ Parte 3: Orientación para realizar una evaluación
- ❖ Parte 4: Orientación sobre el uso para la mejora del proceso y la determinación de la capacidad del proceso
- ❖ Parte 5: Un modelo ejemplar de evaluación de procesos

El parte que va a ser utilizada en la presenten investigación es la cláusula 6.3: Modelo de evaluación de procesos correspondiente a la parte 2 de este estándar, la cual trata sobre la evaluación de procesos y la aplicación de evaluaciones de procesos con el fin de mejorar los procesos y determinar la capacidad de los procesos. Allí se definen los requisitos mínimos para la implementación de la evaluación que aseguren la objetividad, imparcialidad, consistencia, repetibilidad y representatividad de los resultados de la evaluación para los procesos evaluados. Si el alcance de la evaluación es similar, los resultados de las evaluaciones de procesos conformes son comparables. Se puede encontrar orientación sobre este tema en ISO/IEC 15504-4.

Para garantizar que los resultados de la evaluación se puedan traducir en un perfil de proceso que cumpla con ISO/IEC 15504 de manera repetible y confiable, los modelos de evaluación de procesos deben cumplir con ciertos requisitos. Cada modelo de evaluación de procesos debe incluir una definición de su propósito, alcance y elementos; además, debe fijarse su correspondencia con el marco para las mediciones y con los modelos de referencia del proceso especificado, y debe incluir un mecanismo para la presentación coherente de los resultados. Un modelo de evaluación de procesos se considera adecuado para fines de evaluación de la capacidad de procesos si cumple con lo siguiente:

- ❖ **Alcance del Modelo de Evaluación de Procesos**

Un modelo de evaluación de procesos debe hacer referencia a al menos un proceso de los modelos de referencia de procesos definidos. Cualquier modelo de evaluación de procesos deberá, para un proceso dado, abordar todos los niveles o un subconjunto continuo de niveles (comenzando en el nivel 1) del marco de medición de la capacidad del proceso para cada uno de los procesos dentro de su alcance. Para cada modelo de evaluación de procesos, se debe declarar el alcance apropiado en forma de la siguiente información:

- el(los) modelo(s) de referencia del proceso seleccionado(s)
- los procesos seleccionados de los modelos de referencia de procesos;
- los niveles de habilidad seleccionados del marco para las mediciones.

✧ **Indicadores del Modelo de Evaluación de Procesos**

Cada modelo de evaluación de procesos se basará en un conjunto de indicadores que aborden explícitamente los propósitos y los resultados de todos los procesos dentro del alcance del modelo de evaluación de procesos especificado en el modelo de referencia de procesos seleccionado y utilizado para demostrar el logro de los atributos del proceso dentro de la capacidad. El nivel de alcance del proceso sirve como modelo de evaluación. Los indicadores sirven para centrar la atención en la implementación de los procesos relevantes que están dentro del alcance del modelo.

✧ **Mapeo de modelos de evaluación de procesos a modelos de referencia de procesos**

Cada modelo de evaluación de procesos deberá proporcionar un mapeo explícito de los elementos relevantes del modelo a los procesos del modelo de referencia de proceso seleccionado y a los atributos de proceso relevantes del marco de medición. La ilustración debe ser completa, clara e inequívoca. El mapeo de los indicadores dentro del modelo de evaluación de procesos debe incluir lo siguiente:

- los propósitos y resultados de los procesos en el modelo de referencia de proceso definido;
- los atributos del proceso en el marco de las mediciones (incluidos todos los resultados enumerados para el logro de cada atributo del proceso individual)

Esto hace posible relacionar modelos de evaluación de procesos que difieren en su estructura con el mismo modelo de referencia de proceso

1.3.1.2 Requerimiento

Requerimiento, según el glosario de la IEEE [26], es “una condición o necesidad de un usuario para resolver un problema, o alcanzar un objetivo” o “una condición o capacidad que debe estar presente en un sistema o componentes de sistema para satisfacer un contrato, estándar, especificación u otro documento formal”. En la norma ISO/IEC/IEEE 29148 [15],

se señala que un requerimiento es “una declaración que traduce o expresa una necesidad y sus limitaciones y condiciones asociadas”. Según la Guía del SWEBOK (2004) [27] , un requisito de software se define como “ la propiedad que un software desarrollado o adaptado debe tener para resolver un problema concreto”.

La clasificación más general establece dos tipos de requisitos: Los requisitos funcionales y los requisitos no funcionales [11].

- ❖ Los requisitos funcionales especifican las funciones que un sistema o algún componente de software debe ser capaz de ejecutar.
- ❖ Los requisitos no funcionales se refieren a aspectos técnicos que debe incluir el sistema y se relacionan con características de restricciones y de calidad que debe cumplir el software.

1.3.2 Características de los requerimientos de acuerdo al estudio realizado

Las propiedades que debe cumplir un requerimiento valido de acuerdo con las normas y estándares estudiados son listadas en la tabla 1.1. Para mejor visualización ver Anexo 1.

Normas/Características	Año	NECESARIO	APROPIADO	SIN AMBIGÜEDAD	COMPLETO	SINGULAR	FACTIBLE	VERIFICABLE	CORRECTO	CONFORME	TRAZABLE	CONSISTENTE	MODIFICABLE
IEEE 1233	1998	X	X	X	X	X	X	X	X	-	X	X	X
IEEE 830	1993	X	X	X	X	X	X	X	X	-	X	X	X
ISO 9000	2015	X	X	X	X	X	X	X	X	-	-	-	-
ISO/IEC 9126	2001	X	X	X	X	-	X	X	X	X	X	-	X
ISO/IEC 25000	2014	X	X	X	X	X	X	X	X	X	X	-	-
ISO/IEC 14598	1999	X	X	X	X	X	X	X	X	-	-	-	-
ISO 12207	2017	X	X	-	X	-	X	X	X	-	-	-	-
ISO 15288	2015	X	X	X	X	X	X	X	X	-	-	-	-
ISO 15504	2011	X	X	X	X	X	X	X	X	-	-	-	-
ISO/IEC TR 24766	2009	X	X	X	X	X	X	X	X	-	X	X	-
ISO/IEC/IEEE 29148	2018	X	X	X	X	X	X	X	X	X	-	-	-

Tabla 1.1: Características de Requerimientos

El estudio realizado dio como resultado seis propiedades esenciales (necesario, apropiado, completo, factible, verificable y correcto) que debe cumplir un requerimiento bien formado, dos propiedades esenciales pero no suficientes (sin ambigüedad y singular), y cuatro propiedades no esenciales y no suficientes (trazable, conforme, consistente, modificable).

De acuerdo a la aplicación web desarrollada por Nieto [28] que utiliza frecuencia de términos clave, obtuvimos el resultado que se muestra en la figura 1.7. Ver Anexo 2.



Figura 1.7: Resultado del proceso de análisis de las propiedades de requerimientos

Las propiedades de los requerimientos de acuerdo a la metodología usada son clasificadas como se muestra en la tabla 1.2.

Condición necesaria y no suficiente	No necesaria y no suficiente	Ninguno
Apropiado	Sin ambigüedad	Necesario
Completo	Singular	
Factible	Conforme	
Verificable	Trazable	
Correcto	Consistente	
	Modificable	

Tabla 1.2: Clasificación de las propiedades de requerimientos de acuerdo a frecuencia de términos clave

Haciendo una comparativa con el estudio conceptual realizado, se puede obtener la diferencia en una propiedad. La metodología de frecuencia de términos claves excluye la propiedad "necesario", ya que en varias normas y estándares que no son de requerimientos de software aparece esta propiedad y tiene su misma definición. Las demás propiedades obtenidas por el estudio conceptual concuerdan y son corroboradas por el método de frecuencia

de términos claves, sin embargo, consideramos que la propiedad "necesario.^{es} fundamental para el desarrollo por lo cual se decide utilizar la propiedad "necesario como indispensable para la implementación. Por ende, la clasificación final es la siguiente: Propiedades indispensables (apropiado, completo, factible, verificable y correcto), propiedades deseables (sin ambigüedad, singular, trazable, conforme, consistente y modificable), y propiedad de implementación (necesario).

1.4 PROPUESTA DE INSTRUMENTO PARA EL REFINAMIENTO DE REQUERIMIENTOS

En esta etapa proponemos un instrumento para el refinamiento de requerimientos a partir del documento F07_Cuestionario_de_Refinamiento_de_Requerimientos de la metodología iPlus [2] y las características obtenidas del estudio conceptual (sección 1.3.2) para obtener un requerimiento bien formado.

Por un lado la metodología iPlus [2] para la obtención de requerimientos para juegos serios consta de cinco fases (fase 1: Identificación, fase 2: Objetivos Pedagógicos, fase 3: Guión del Juego Lúdico, fase 4: Juego, fase 5: Refinar), de las cuales nos centramos en la quinta fase de refinamiento. En esta fase se utiliza el cuestionario F07_Cuestionario_de_Refinamiento_de_Requerimientos, basado en las propiedades de requerimientos de ISO/IEC/IEEE 29148 para la evaluación de requerimientos.

Para esta fase se debe tener en cuenta lo siguiente:

- ❖ Los requerimientos deben indicar “qué” se necesita, no “cómo”.
- ❖ Se debe evitar los términos vagos y generales que dan como resultado requerimientos imposible de verificar o de múltiple interpretación.

Por otro lado, es necesario tomar en cuenta las características (necesario, apropiado, completo, factible, verificable, correcto, sin ambigüedad, singular, trazable, conforme, consistente y modificable) que deben cumplirse de acuerdo al estudio realizado para que un requerimiento sea bien formado. Además, de acuerdo a la clasificación final que se obtuvo en la sección 1.3.2, se determinó las plantillas que se muestran en las figuras 1.8 y 1.9 que serán utilizadas en el aplicativo web.

		DELEGATORIOS ACEPTABLES										
Preguntas	¿Los requerimientos están claros, no existe ambigüedad?	¿El requerimiento es factible, es decir es realizable, a pesar de las limitaciones del sistema (tiempo por costo, horario, y por parte técnica) con riesgo aceptable? ¿El requerimiento tienen alguna estimación técnica?	El requerimiento es correcto representa la necesidad real que el cliente necesita. ¿?	¿El requerimiento es apropiado está dentro del alcance del proyecto refleja la necesidad real?	¿El requerimiento es verificable mediante un caso de prueba?	¿El requerimiento esta completo describe la necesidad del cliente sin necesidad de ampliarlo?	¿El requerimiento es necesario, sino se incluye como requisito existirá alguna deficiencia, para otros requerimientos?	¿El requerimiento establece una sola característica, es singular, o puede descomponerse en varios?	El requerimiento esta conforme al estándar de la organización?	¿El requerimiento es consistente no contradice a otros requerimientos o no se encuentra repetido?	¿El requerimiento puede ser modificable sin afectar a otros requerimientos o alcance del producto?	¿El requerimiento tiene una trazabilidad original que mantiene la necesidad del cliente?
Propósitos												

Figura 1.8: Plantilla para propósitos - cliente

		Revisión de acciones funcionalidades que serán implementadas en el juego seno o aplicación informática										
Preguntas	La tarjeta gameplay incluye BLOQUES completa respetando el formato?	Existe sinergia entre la tarjeta gameplay con la historia/narrativa/ género. Existe consistencia es apropiado?	La tarjeta gameplay esta relacionado a la funcionalidad que el usuario necesita, de acuerdo a su alcance. Es correcto?	La tarjeta gameplay es necesario que sea implementada?	El gameplay es verificable?	Es factible realizar esta funcionalidad en la plataforma tecnológica que se va a desarrollar?	La tarjeta gameplay requiere aclaración para ser implementado, no esta con ambigüedad?	La tarjeta gameplay puede ser mejorada o modificada?	La tarjeta Gameplay no se puede dividir en otras funcionalidades es singular.	La tarjeta gameplay mantiene su trazabilidad respetando el principio del diseño del juego?	La tarjeta Gameplay no se contradice con el diseño inicial propuesto del juego?	¿El diseño funcional del juego que contiene la tarjeta gameplay es conforme con todas las opiniones de los participantes?

Figura 1.9: Plantilla para ideas GamePlay

Para usar las plantillas de las figuras 1.8 y 1.9, se debe tener en cuenta lo siguiente:

- ❖ Criterios de lenguaje de requerimientos, y
- ❖ Atributos de requerimientos.

1.4.1 Criterios de lenguaje de requerimientos.

Al expresar requisitos textuales se debe considerar los dos puntos mencionados de la ISO/IEC/IEEE 29148 para la evaluación de requerimientos [15], dando como resultado requisitos bien formados. Además, se debe considerar el estilo de escritura que según Wiegers y Beatty [29] debe tener primero la declaración de necesidad o funcionalidad, seguido de los detalles de apoyo (justificación, origen, prioridad y otros atributos de requisitos), proporcionando facilidad de lectura y comprensión del requerimiento.

1.4.2 Atributos de requerimientos.

Los requerimientos bien formados deben incluir atributos descriptivos definidos para ayudar a identificar los requisitos relevantes y ayudar a comprender y gestionar los requerimientos [15].

- ❖ **Identificación.** Cada requisito debe identificarse de forma única (es decir, número, etiqueta de nombre, mnemónico). La identificación puede reflejar vínculos y relaciones, si es necesario, o pueden ser independientes de la identificación. Los identificadores únicos ayudan en el seguimiento de los requisitos. Una vez asignada, la identificación es única: nunca se cambia (incluso si cambia el requisito identificado) ni se reutiliza (incluso si se elimina el requisito identificado).

- ❖ **Justificación.** La justificación proporciona la razón por la cual el requisito es necesario y apunta a cualquier análisis de respaldo, estudio comercial, modelado, simulación u otra evidencia objetiva sustantiva.

Según Mike Cohn referenciado en Historias de Usuario-Ingeniería de Requisitos Ágil [30] recomienda seguir el siguiente patrón que responde tres preguntas: ¿quién se beneficia? ¿qué se quiere? y ¿cuál es el beneficio?, para garantizar que la funcionalidad este descrita en un alto nivel y de forma rápida.

Como [rol del usuario], **quiero** [objetivo], **para poder** [beneficio].

Para el desarrollo del proyecto integrador se utilizará el formato presentado en tabla 1.4.

Identificador	Descripción
H(número)	Como [rol de usuario], quiero[objetivo], para poder [beneficio].
H1	<p style="text-align: center;">○</p> El/La [rol de usuario], requiero [funcionalidad], para [razón].

Tabla 1.4: Formato de ingreso de requerimiento.

1.5 RESUMEN DEL CAPÍTULO

En el capítulo 1 se describe la problemática que presenta el desarrollo de software debido a la mala elicitación de requerimientos y su respectivo refinamiento. Como respuesta a esta problemática se propone el desarrollo de una herramienta web para la validación de requerimientos de software.

Además, el capítulo presenta el estudio conceptual y propuesta de instrumento que se utilizará para el desarrollo de la herramienta web en respuesta a la problemática planteada. El estudio conceptual se llevó a cabo mediante la metodología propuesta por Morales [12], obteniendo como resultado las características que debe tener un requerimiento bien formado. Además, mediante el uso de la aplicación web desarrollada por M.Nieto [28] se logra corroborar los resultados obtenidos y modificar los resultados obtenidos por el estudio conceptual.

Por otro parte, para la elaboración del instrumento se usó como base la quinta fase de la metodología iPlus, que corresponde al refinamiento de requerimientos. El instrumento nos

permite evaluar las características estudiadas anteriormente, mediante atributos de requerimientos tanto del cliente como de un GamePlay.

2 METODOLOGÍA E IMPLEMENTACIÓN

El presente proyecto para la fase de implementación utilizará la metodología Scrum [31], que permite obtener los requerimientos para el diseño del aplicativo web. Se tendrá como entrada el Product Backlog correspondiente a las historias de usuario obtenidas, y estas serán realizadas en Sprints. Finalmente, para la fase de lanzamiento se realizará el despliegue del proyecto en un ambiente de producción.

2.1 METODOLOGÍA

Scrum, co-creado por Ken Schwaber y Jeff Sutherland, es un marco de trabajo ligero dentro del cual se pueden abordar problemas adaptivos completos mientras se entregan productos de mayor valor posible [31].

A través de un Scrum Master se fomenta un entorno en el que:

1. Un Product Owner ordena el trabajo de un problema complejo en un Product Backlog.
2. El Equipo Scrum convierte una selección del trabajo en un incremento de valor durante un Sprint.
3. El Equipo Scrum y las partes interesadas inspeccionan los resultados y se ajustan para el próximo Sprint.
4. Repite

Scrum se basa en la inteligencia colectiva de las personas que lo utilizan, emplea un enfoque iterativo e incremental para optimizar la previsibilidad y controlar el riesgo basado en el empirismo, afirmando que el conocimiento proviene de la experiencia y el pensamiento lean, que reduce el desperdicio y se enfoca en lo esencial [31].

SCRUM FRAMEWORK

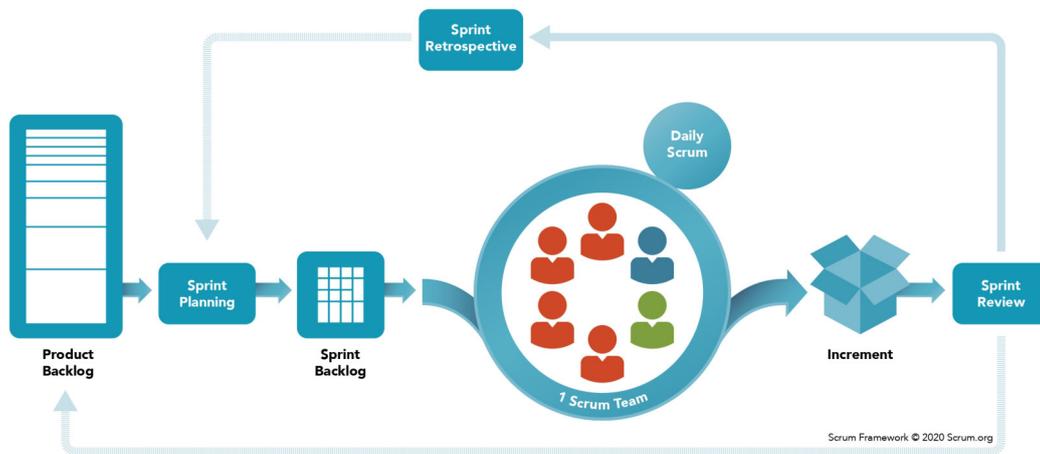


Figura 2.1: Esquema de trabajo de SCRUM [31]

2.1.1 Equipo de Scrum

La unidad fundamental de Scrum es un grupo multifuncional pequeño de personas denominado Equipo Scrum en el cual no existen subgrupos o jerarquías. En otros términos, es una unidad cohesiva de profesionales enfocados en un objetivo a la vez [31].

Los miembros de este equipo tienen las habilidades necesarias para crear valor en cada Sprint, se auto-gestionan y es lo suficientemente pequeño para seguir siendo ágil, comunicarse mejor y ser más productivos, y lo suficientemente grande para completar un trabajo significativo dentro de un Sprint [32].

El Equipo Scrum es el responsable de las actividades relacionadas al producto, crear un incremento valioso y útil en cada Sprint, están estructurados y organizados para gestionar su propio trabajo a un ritmo sostenible que mejore el enfoque y consistencia del equipo en sí [32].

Scrum define tres responsabilidades específicas dentro del Equipo Scrum [32]: los desarrolladores, el Product Owner y el Scrum Master.

2.1.1.1 Desarrolladores

Los desarrolladores son las personas del Equipo Scrum que se comprometen a crear cualquier aspecto de un incremento utilizable en cada Sprint. Según el dominio de trabajo, las habilidades específicas que son necesarias en los desarrolladores son amplias y variadas, estos son responsables de:

- ❖ Crear un plan para el Sprint, el Sprint Backlog
- ❖ Inculcar calidad al adherirse a una definición de DONE
- ❖ Adaptar el plan cada día hacia el objetivo del Sprint
- ❖ Ser responsables profesionalmente de unos a otros

2.1.1.2 Product Owner

El Product Owner es el responsable de maximizar el valor del producto resultante del trabajo del Equipo Scrum, esto varía entre organizaciones, equipos e individuos. También es responsable de la gestión eficaz de la cartera de productos, que incluye: [32]

- ❖ Desarrollar y comunicar explícitamente el Objetivo del Producto
- ❖ Crear y comunicar claramente los elementos del Product Backlog
- ❖ Ordenar elementos de la Lista de Producto; y,
- ❖ Asegurar que el Product Backlog sea transparente, visible y comprensible.

2.1.1.3 Scrum Master

El Scrum Master es un verdadero líder que sirve y ayuda a todos a comprender la teoría y práctica de Scrum, tanto en el equipo como en la organización, es el responsable de la efectividad del Equipo Scrum al permitir que este mejore sus prácticas dentro del marco de trabajo Scrum [32].

2.1.2 Valores de Scrum

Los valores que las personas dentro de un equipo deben poseer para el uso exitoso de Scrum son:

1. **Compromiso.**- Las personas se comprometen personalmente a lograr los objetivos del Equipo Scrum.
2. **Enfoque.**- Todos se enfocan en el trabajo del Sprint y las metas del Equipo Scrum.
3. **Apertura.**- El Equipo Scrum y sus partes interesadas (stakeholders) acuerdan ser abiertos sobre todo el trabajo y los desafíos para realizar el trabajo.
4. **Respeto.**- Los miembros del Equipo Scrum se respetan entre sí para ser personas capaces e independientes.
5. **Coraje.**- Los miembros del Equipo Scrum tienen valor para hacer lo correcto y trabajar en problemas difíciles.

Estos valores dirigen al Equipo Scrum con respecto a su trabajo, acciones y comportamiento. Las decisiones, pasos y la forma en que se utiliza Scrum deben reforzar estos valores, no disminuir o socavar, a medida que se trabaja con los eventos o artefactos de Scrum. Cuando estos valores son incorporados por el Equipo Scrum y las personas con las que trabajan los pilares empíricos de Scrum cobran vida y generan confianza [32].

2.1.3 Artefactos y Eventos de Scrum

Los eventos de Scrum se utilizan para crear regularidad y minimizar la necesidad de reuniones no definidas en Scrum. Estos tienen un límite de tiempo, su duración es fija, no se puede acortar ni alargar, y se pueden finalizar siempre que se logre el propósito del evento, asegurando que se gaste una cantidad adecuada de tiempo sin permitir el desperdicio en el proceso [31].

Los artefactos de Scrum están diseñados específicamente para maximizar la transparencia de la información clave para que todos tengan la misma comprensión del artefacto [31].

2.1.3.1 Product Backlog

El Product Backlog es una lista ordenada de lo necesario para mejorar el producto, esta es realizada por el Equipo Scrum. Los elementos de esta lista se consideran listos para la selección durante la Planificación del Sprint y adquieren transparencia después de las actividades de refinación, el cual es el acto de desglosar y definir aún mas los elementos en elementos más pequeños y precisos. Esta es una actividad continua para agregar detalles y los atributos a menudo varían con el dominio de trabajo [33].

Los desarrolladores son responsables del dimensionamiento, el Product Owner puede influir en los desarrolladores ayudando a comprender y seleccionar compensaciones. En general, se utiliza el Product Backlog para describir el próximo trabajo en el producto [33].

2.1.3.2 Incremento

El incremento es un peldaño hacia el objetivo del producto, en cada incremento se suman los anteriores a este, se verifica y se garantiza que todos los incrementos funcionen juntos siendo este utilizable a medida que proporciona valor [34].

El trabajo no puede considerarse parte de un incremento a menos que cumpla con la definición de DONE, esto significa que debe cumplir con las medidas de calidad requeridas por el producto [34].

2.1.3.3 Sprint

Son eventos de duración fija de un mes o menos para crear consistencia, estos comienzan inmediatamente después de la conclusión de anterior. Todo el trabajo necesario para lograr el objetivo del producto incluida la planificación de Sprint, Scrum diarios, revisión de Sprint y retrospectiva de Sprint, se realiza dentro de Sprints [35].

Durante el Sprint:

- ❖ No se realizan cambios que puedan poner en peligro el Objetivo del Sprint;
- ❖ La calidad no disminuye;
- ❖ El Product Backlog se refina según sea necesario; y,

- ❖ El alcance puede aclararse y renegociarse con el Product Owner a medida que se aprende más.

2.1.3.4 Sprint Planning

El Sprint Planning inicia el Sprint al diseñar el trabajo que se realizará en este, el resultado es creado por el trabajo colaborativo de todo el Equipo Scrum. El Product Owner se asegura que los asistentes estén preparados para analizar los elementos más importantes de la cartera de productos y como se relacionan con el objetivo del producto [36]. El Sprint Planning aborda estos temas:

- ❖ ¿Por qué es valioso este Sprint?
- ❖ ¿Qué se puede hacer en este Sprint?
- ❖ ¿Cómo se realizará el trabajo elegido?

La planificación de Sprint tiene un límite de tiempo de un máximo de ocho horas para un Sprint de un mes, y disminuye si el Sprint es más corto [36].

2.1.3.5 Daily Scrum

El propósito del Daily Scrum es inspeccionar el progreso hacia el objetivo del Sprint y adaptar el Backlog según sea necesario, ajustando el próximo trabajo planificado. Este es un evento de 15 minutos realizado a la misma hora y en el mismo lugar los días hábiles de Sprint, en esta participan los desarrolladores del Equipo Scrum [37].

Los Daily Scrum mejora las comunicaciones, identifica impedimentos, optimiza la probabilidad de que se alcance el objetivo del Sprint, promueve la toma de decisiones rápidas y, en consecuencia, elimina la necesidad de otras reuniones [37].

2.1.3.6 Sprint Review

El propósito de la Sprint Review es inspeccionar el resultado del Sprint, el Equipo Scrum presenta a las partes interesadas clave los resultados de su trabajo y se analiza el progreso

hacia el objetivo de producto. Durante el evento, el Equipo Scrum y las partes interesadas revisan lo logrado en el Sprint, el Product Backlog puede ajustarse para nuevas oportunidades. Al ser una sesión de trabajo, el Equipo Scrum debe limitarla a una presentación con un límite de tiempo de un máximo de 4 horas para un Sprint de un mes [38].

El resultado de la Revisión del Sprint es un Product Backlog revisado que define los elementos probables del Product Backlog para el próximo Sprint [38].

2.1.3.7 Sprint Retrospective

El propósito del Sprint Retrospective es planificar formas de aumentar calidad y eficacia. El equipo Scrum inspecciona como fue el último Sprint con respecto a personas, interacciones, procesos, herramientas y su definición de DONE, analiza que salió bien durante el Sprint, que problemas se encontraron y como se resolvieron (o no) dichos problemas [39].

Sprint Retrospective concluye el Sprint en un límite de tiempo de un máximo de tres horas para un Sprint de un mes y durante la misma el equipo analiza:

- ❖ Lo que salió bien en el Sprint
- ❖ Que se puede mejorar
- ❖ ¿Qué nos comprometemos a mejorar en el próximo Sprint?

2.2 FASE INICIAL DE SCRUM

En esta sección, se describe la fase inicial del proceso de scrum, el que incluye el levantamiento de requerimientos, la identificación de roles, y la creación de historias de usuario tanto épicas como normales, y finalmente el Product Backlog.

2.2.1 Definición de requerimientos

El objetivo de la aplicación es automatizar la quinta fase de la metodología iPlus, mediante la opinión del experto y con ayuda de plantillas elaboradas para este proyecto. Para este fin se hizo uso de historias de usuario épicas, las cuales son historias muy amplias y se

desglosan en historias de usuario más pequeñas [30]. En esta sección se describen las tres historias épicas planteadas.

Historias de Usuario Épica	HUE01
Título: Desarrollar el módulo de carga de requerimientos sin refinar.	
Descripción: Como usuario, requiero poder cargar una plantilla en formato .xlsx que tenga 100 requerimientos sin refinar.	

Tabla 2.1: Historia de usuario épica 1.

Historias de Usuario Épica	HUE02
Título: Desarrollar el módulo de refinamiento de requerimientos.	
Descripción: Como usuario, requiero poder efectuar el refinamiento de los requerimientos con las propiedades que definen un requerimiento bien formado.	

Tabla 2.2: Historia de usuario épica 2.

Historias de Usuario Épica	HUE03
Título: Desarrollar el módulo de administración de usuario.	
Descripción: Como usuario, requiero que se lleve un registro de los procesos realizados en la aplicación, es decir requiero guardar, modificar y borrar los procesos que se efectúen.	

Tabla 2.3: Historia de usuario épica 3.

2.2.2 Identificación de roles

En la tabla 2.4 se muestra los roles de Scrum para este proyecto.

Rol	Encargado	Descripción
Propietario del producto	Mayra Carrión	Es la encargada de encaminar la visión del cliente hacia el equipo Scrum. Es la responsable de que la pila del producto (Product Backlog) se gestione de forma adecuada.

Scrum máster	Marco Santorum	Permite la interacción entre el equipo de desarrollo y el dueño del producto con el fin asegurar que los objetivos planteados se alcancen alineados a los políticas y reglas de Scrum.
Equipo de desarrollo	Leslie Díaz	Encargados de diseñar, desarrollar y realizar las pruebas correspondientes, al producto.
Equipo de desarrollo	Pamela Pachacama	Encargados de diseñar, desarrollar y realizar las pruebas correspondientes, al producto.

Tabla 2.4: Roles de Scrum

2.2.3 Historias de usuario

De acuerdo con los requerimientos planteados y las historias de usuario épicas, se definieron las siguientes historias de usuario.

Historia de Usuario	HU01
Número: 1	Usuario: Administrador del sistema
Nombre historia: Registro de los Usuarios para el ingreso en el sistema	
Prioridad: Media	Riesgo en desarrollo: Alto
Iteración asignada: 1	Responsables: Leslie Díaz Pamela Pachacama
<p>Descripción: Yo como administrador del sistema requiero que los usuarios se registren para poder utilizar la aplicación, con el fin de poder llevar un registro del uso de la aplicación.</p> <p>Descripción técnica: - Se creará un módulo de inicio de sesión que permita al usuario registrarse ingresando un correo electrónico, un nombre de usuario y una contraseña.</p>	
Observaciones:	

Tabla 2.5: Historia de Usuario HU01.

Historia de Usuario	HU02
Número: 2	Usuario: Usuario del sistema
Nombre historia: Definir la estructura de la base de datos	
Prioridad: Alta	Riesgo en desarrollo: Alto
Iteración asignada: 1	Responsables: Leslie Díaz Pamela Pachacama
<p>Descripción: Como usuario requiero poder mantener mi proceso actual (documento y refinamiento) ingresado, cuando cambie de ordenador.</p> <p>Descripción técnica: Se almacenará la información de los requerimientos ingresados, así como el refinamiento realizado previamente o durante el proceso en la base de datos lo que permitirá al usuario acceder al mismo desde otro ordenador y continuar su trabajo si así lo quisiese.</p>	
<p>Observaciones: El proceso se almacenará siempre y cuando se hayan cargado más de un requerimiento.</p>	

Tabla 2.6: Historia de Usuario HU02.

Historia de Usuario	HU03
Número: 1	Usuario: Usuario del sistema
Nombre historia: Iniciar sesión como usuario	
Prioridad: Media	Riesgo en desarrollo: Alto
Iteración asignada: 1	Responsables: Leslie Díaz Pamela Pachacama
<p>Descripción: Como usuario requiero ingresar al sistema con mis credenciales, Nombre de usuario y contraseña.</p> <p>Descripción técnica: Se buscarán las credenciales del usuario almacenadas en la base de datos y se almacenará la información de la sesión actual tanto en la base de datos como en una cookie.</p>	
<p>Observaciones: El usuario podrá acceder únicamente a los proyectos que haya creado.</p>	

Tabla 2.7: Historia de Usuario HU03.

Historia de Usuario	HU04
Número: 1	Usuario: Usuario del sistema
Nombre historia: Administrar proyectos (crear, editar y eliminar).	
Prioridad: Media	Riesgo en desarrollo: Alto
Iteración asignada: 1	Responsables: Leslie Díaz Pamela Pachacama
<p>Descripción: Como usuario requiero crear, editar, duplicar y eliminar los proyectos efectuados en la aplicación</p> <p>Descripción técnica: - Se creará un módulo que le permita al usuario llevar un registro de los proyectos efectuados, así como su edición, duplicación y eliminación. - El usuario podrá descargar de los resultados de la validación realizada de un proyecto una vez que esté finalizado.</p>	
<p>Observaciones: Al eliminar un proyecto se eliminará toda su información relacionada (requerimientos, refinamiento de estos y resultados) siendo esta opción irreversible.</p>	

Tabla 2.8: Historia de Usuario HU04.

Historia de Usuario	HU05
Número: 1	Usuario: Usuario del sistema
Nombre historia: Ingresar requerimientos por medio de cargar archivo .xlsx	
Prioridad: Media	Riesgo en desarrollo: Alto
Iteración asignada: 1	Responsables: Leslie Díaz Pamela Pachacama
<p>Descripción: Como usuario requiero poder cargar un archivo .xlsx que contengan requerimientos sin refinar.</p> <p>Descripción técnica: Procesar el archivo .xlsx, con el fin de que se pueda extraer su contenido.</p>	

Observaciones:
 Si el archivo .xlsx no coincide con la plantilla, se emitirá un mensaje de error de carga.
 El máximo de requerimientos permitidos será de 100, en caso de que existan más, se emitirá un mensaje de error de carga.

Tabla 2.9: Historia de Usuario HU05.

Historia de Usuario	HU06
Número: 1	Usuario: Usuario del sistema
Nombre historia: Ingresar requerimientos de forma gráfica.	
Prioridad: Alta	Riesgo en desarrollo: Alto
Iteración asignada: 1	Responsables: Leslie Díaz Pamela Pachacama
<p>Descripción: Como usuario requiero poder ingresar requerimientos de forma gráfica, donde me pida la descripción del requerimiento, la prioridad, el padre de este e ideas relacionadas.</p> <p>Descripción técnica: Se creará un módulo que permita al usuario ingresar requerimientos de forma gráfica.</p> <p>Observaciones:</p>	

Tabla 2.10: Historia de Usuario HU06.

Historia de Usuario	HU07
Número: 1	Usuario: Usuario del sistema
Nombre historia: Administrar requerimientos (crear, editar y eliminar).	
Prioridad: Alta	Riesgo en desarrollo: Alto
Iteración asignada: 1	Responsables: Leslie Díaz Pamela Pachacama
<p>Descripción: Como usuario requiero crear, editar y eliminar requerimientos, cargados o ingresados de forma gráfica.</p> <p>Descripción técnica: Se creará un módulo que le permita al usuario llevar un registro de los requerimientos ingresados, así como su edición y eliminación.</p>	

Observaciones:

Tabla 2.11: Historia de Usuario HU07.

Historia de Usuario	HU08
Número: 1	Usuario: Administrador del sistema
Nombre historia: Administrar participantes (crear, editar y eliminar).	
Prioridad: Baja	Riesgo en desarrollo: Medio
Iteración asignada: 1	Responsables: Leslie Díaz Pamela Pachacama
<p>Descripción: Como usuario requiero crear, editar y eliminar participantes.</p> <p>Descripción técnica: Se creará un módulo que le permita al usuario llevar un registro de los participantes creados, así como su edición y eliminación.</p>	
<p>Observaciones: Si se elimina un participante este será desvinculado de todos los proyectos en los que haya participado.</p>	

Tabla 2.12: Historia de Usuario HU08.

Historia de Usuario	HU09
Número: 1	Usuario: Usuario del sistema
Nombre historia: Exportar resultados	
Prioridad: Baja	Riesgo en desarrollo: Medio
Iteración asignada: 1	Responsables: Leslie Díaz Pamela Pachacama
<p>Descripción: Como usuario requiero poder exportar los resultados obtenidos en un archivo.</p> <p>Descripción técnica: Se mostrará una tabla que contenga los resultados del refinamiento.</p>	

<p>Observaciones:</p> <p>El usuario podrá descargar un archivo con formato .xlsx que contenga el resultado del refinamiento, así como observaciones de los requerimientos</p>
--

Tabla 2.13: Historia de Usuario HU09.

Historia de Usuario	HU10
Número: 1	Usuario: Usuario del sistema
Nombre historia: Seleccionar las propiedades de requerimiento bien formados.	
Prioridad: Media	Riesgo en desarrollo: Alto
Iteración asignada: 1	Responsables: Leslie Díaz Pamela Pachacama
<p>Descripción:</p> <p>Yo como usuario, requiero poder seleccionar propiedades que cumplan los requerimientos ingresados.</p> <p>Descripción técnica:</p> <p>Se creará un módulo que le permita al usuario seleccionar y deseleccionar las propiedades de un requerimiento bien formado.</p>	
<p>Observaciones:</p> <p>Si los requerimientos ingresados fueron mediante la plantilla, se leerá los datos ingresados y llenando las propiedades si son necesarias.</p>	

Tabla 2.14: Historia de Usuario HU10.

2.2.4 Product Backlog

La construcción del Product Backlog se basa en las historias de usuario tomando en cuenta el riesgo de desarrollo y la prioridad de negocio de cada una de las historias de usuario como se muestra en las siguientes tablas.

El Product Backlog inicial se muestra en la tabla 2.15

Product Backlog			
Historia de Usuario	Enunciado	Prioridad	Riesgo
HU02	Definir la estructura de la base de datos.	Alta	Alto

HU06	Ingresar requerimientos de forma gráfica.	Alta	Alto
HU07	Administrar requerimientos (crear, editar y eliminar).	Alta	Alto
HU01	Registro de los Usuarios para el ingreso en el sistema.	Medio	Alto
HU03	Iniciar sesión como usuario.	Medio	Alto
HU04	Administrar proyectos (crear, editar y eliminar).	Medio	Alto
HU05	Ingresar requerimientos por medio de cargar archivo .xlsx	Medio	Alto

Tabla 2.15: Product Backlog inicial

El Product Backlog final se muestra en la tabla 2.16

Product Backlog			
Historia de Usuario	Enunciado	Prioridad	Riesgo
HU02	Definir la estructura de la base de datos.	Alta	Alto
HU06	Ingresar requerimientos de forma gráfica.	Alta	Alto
HU07	Administrar requerimientos (crear, editar y eliminar).	Alta	Alto
HU01	Registro de los Usuarios para el ingreso en el sistema.	Medio	Alto
HU03	Iniciar sesión como usuario.	Medio	Alto
HU04	Administrar proyectos (crear, editar y eliminar).	Medio	Alto
HU05	Ingresar requerimientos por medio de cargar archivo .xlsx	Medio	Alto
HU10	Seleccionar las propiedades de requerimiento bien formados.	Medio	Alto
HU08	Administrar participantes (crear, editar y eliminar).	Baja	Medio
HU09	Exportar resultados.	Baja	Medio

Tabla 2.16: Product Backlog final

2.3 IMPLEMENTACIÓN DE SCRUM

En esta sección se describe la implementación y ejecución de la metodología Scrum, para los sprints 0, 1, 2 y 3. Cada sprint consta de: objetivos, Product Backlog, requisitos completados, Sprint Review y Sprint Retrospective.

2.3.1 Sprint 0

❖ Objetivos del Sprint

Definir la estructura de la base de datos y administrar e ingresar requerimientos de forma gráfica.

❖ Product Backlog

Las historias de usuario y las tareas que se ejecutaron durante este sprint se muestran en la tabla 2.17.

Código	Historia de Usuario	Tarea	Criterios de Aceptación
HU02	Definir la estructura de la base de datos	Crear la estructura de la base de datos para almacenar la información del usuario y del proyecto.	Registro de creación de tablas en el servidor de la base de datos
HU06	Ingresar requerimientos de forma gráfica	Crear un módulo para el ingreso de requerimientos en forma gráfica.	Registro de un requerimiento en el sistema
		Desarrollar la interfaz que permita visualizar la estructura de un requerimiento	Mostrar en la interfaz la información del requerimiento (rol, título, prioridad, descripción y propósitos).
HU07	Administrar requerimientos (crear, editar y eliminar)	Crear las funciones que permitan crear, editar y eliminar un registro (requerimiento).	Crear, editar y eliminar un requerimiento.

Tabla 2.17: Sprint 0 Backlog

❖ Requisitos completados del Sprint

Definir la estructura de la base de datos y administrar e ingresar requerimientos de forma gráfica.

El objetivo del sprint 0 consiste en la creación de: la estructura de la base de datos que será utilizada en el proyecto, el módulo para el ingreso de requerimientos en forma gráfica; y las funciones que permitan crear, editar y eliminar un requerimiento. A

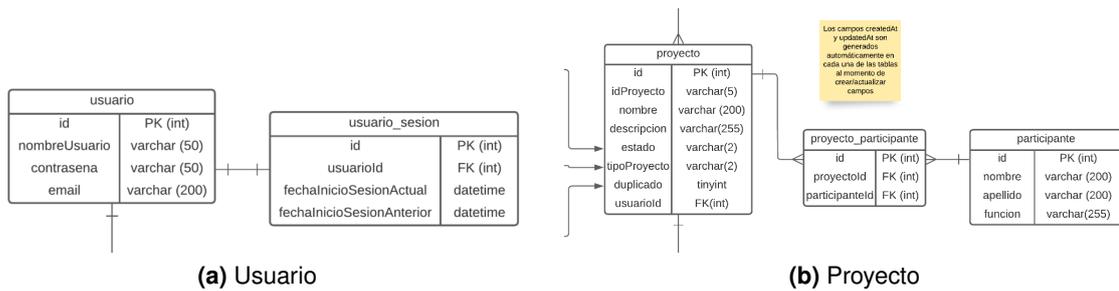


Figura 2.4: Estructura de usuario y proyecto

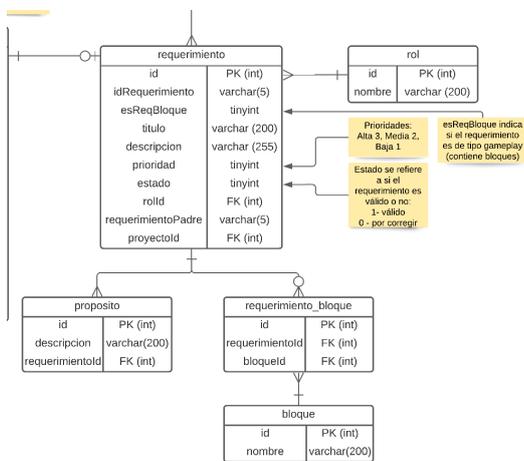


Figura 2.3: Estructura de requerimientos.

El proyecto se estructuró para que sea utilizado por un usuario, por lo cual se creó su respectiva estructura como muestra la figura 2.4a.

El proyecto debe ser capaz de diferenciar requerimientos por proyecto, por ende, también se desarrolló una estructura para proyectos como se muestra en la figura 2.4b.

Finalmente se creó una estructura capaz de guardar los resultados que queremos enviar al usuario para que sea entendido. Se realizó la estructura de resultados como se muestra en la figura 2.5.



Figura 2.5: Estructura de resultado.

- ❖ **Ingresar requerimientos de forma gráfica:** El ingreso de requerimientos de forma gráfica constituye una de las tareas críticas del desarrollo de la aplicación, ya que los datos ingresados serán utilizados para el refinamiento individual de cada requerimiento. Para determinar la forma gráfica que se emplea en el proyecto, se toma como referencia la segunda fase de la metodología iPlus, de la obtención de objetivos específicos de la aplicación.

Los datos que se recolectan de forma gráfica son; identificador, rol, título, prioridad, descripción y propósito. Como se muestra en la figura 2.6.

APLICATIVO PARA LA VALIDACIÓN DE REQUERIMIENTOS DE SOFTWARE

Refinamiento

Agregar requerimiento Visualización de requerimiento Refinamiento Resultados

Regresar Continuar

Cliente GamePlay

Identificador Rol Padre

Título

Descripción

Ideas relacionadas

Identificador Descripción Información Relevante Ideas

No existe resultados

© 2022 Escuela Politécnica Nacional. Todos los derechos reservados

Figura 2.6: Método gráfico - requerimiento cliente.

Para la obtención de requerimientos de GamePlay se toma como referencia la cuarta fase de la metodología iPlus, de cómo se Juega (GamePlay). Los datos que se recolectan de forma gráfica son; identificador, descripción y bloques GamePlay. Como se muestra en la figura 2.7.

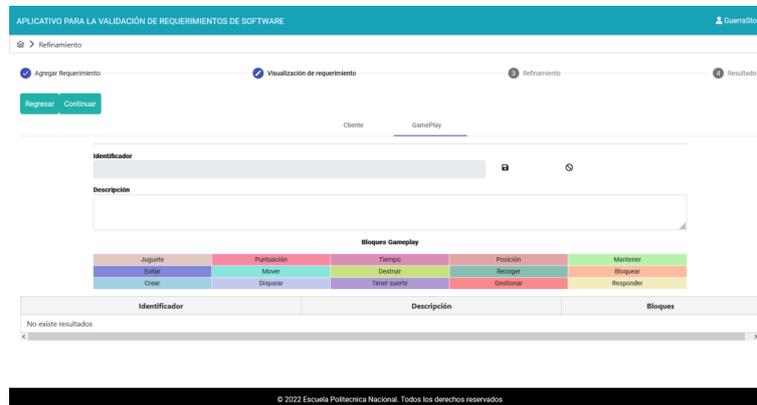


Figura 2.7: Método gráfico - requerimiento GamePlay.

- ❖ **Administrar requerimientos (crear, editar y eliminar):** Con la definición de la estructura de base de datos realizado en las tareas previas, se procedió a crear las funciones que le permitan al usuario crear, editar y eliminar los requerimientos realizados en la aplicación.
- ❖ **Sprint Review**

Se cumplió con el objetivo del sprint que consistía en definir la estructura de la base de datos y administrar e ingresar requerimientos de forma gráfica. Como se muestra en la tabla 2.18.

Código	Historia de Usuario	Tarea	Criterios de Aceptación	Cumplido
HU02	Definir la estructura de la base de datos	Crear la estructura de la base de datos para almacenar la información del usuario y del proyecto.	Registro de creación de tablas en el servidor de la base de datos	Si
HU06	Ingresar requerimientos de forma gráfica	Crear un módulo para el ingreso de requerimientos en forma gráfica.	Registro de un requerimiento en el sistema	Si

		Desarrollar la interfaz que permita visualizar la estructura de un requerimiento	Mostrar en la interfaz la información del requerimiento (rol, título, prioridad, descripción y propósitos).	Si
HU07	Administrar requerimientos (crear, editar y eliminar)	Crear las funciones que permitan crear, editar y eliminar un registro (requerimiento).	Crear, editar y eliminar un requerimiento.	Si

Tabla 2.18: Sprint 0 Review

❖ **Sprint Retrospective**

En este apartado se muestra el avance del trabajo durante el sprint, la figura 2.8 muestra la ejecución de las tareas correspondientes a la definición de la estructura de la base de datos y la administración e ingreso de requerimientos en forma gráfica. Existe un incremento en el tiempo que tomó ejecutar dichas tareas, debido a cambios en la estructura de base de datos.

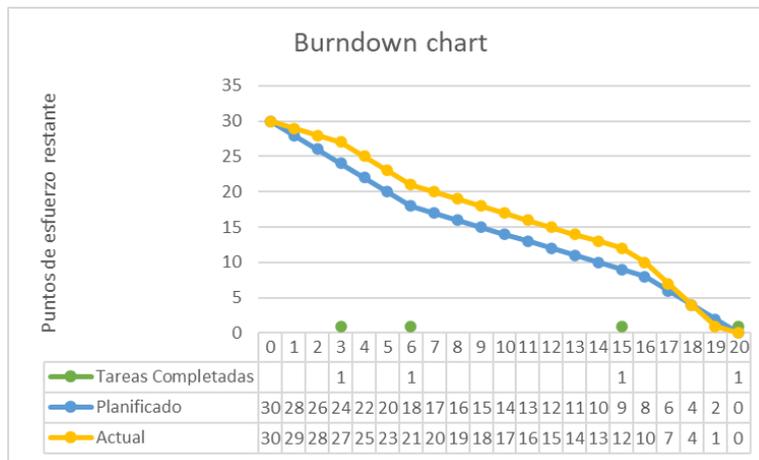


Figura 2.8: Burndown chart sprint 0

2.3.2 Sprint 1

❖ **Objetivos del Sprint**

Registrar e iniciar sesión como usuario en el sistema y administrar proyectos.

❖ Product Backlog

Las historias de usuario y las tareas que se ejecutaron durante este sprint se muestran en la tabla 2.19.

Código	Historia de Usuario	Tarea	Criterios de Aceptación
HU01	Registro de usuario para el ingreso en el sistema	Crear un módulo para el ingreso de usuarios en el sistema	Registro de un usuario en el sistema
		Desarrollar la interfaz que permita visualizar el registro de un usuario	Mostrar en la interfaz la información del usuario (nombre, correo electrónico, contraseña).
HU03	Iniciar sesión como usuario	Desarrollar la interfaz que permita visualizar el ingreso de un usuario al sistema.	Mostrar en la interfaz la información de usuario para ingresar al sistema (correo electrónico, contraseña).
HU04	Administrar Proyectos (crear, editar y eliminar)	Crear un módulo para la administración de proyectos	Registro de un proyecto en el sistema.
		Crear las funciones que permitan crear, editar y eliminar un registro (proyecto).	Crear, editar y eliminar un proyecto.

Tabla 2.19: Sprint 1 Backlog

❖ Requisitos completados del Sprint

El objetivo del sprint 1 consiste en desarrollar dos módulos, uno para el registro de un usuario en el sistema y el otro para administrar proyectos. A continuación, se describe la ejecución de las tareas correspondientes a las siguientes historias de usuario:

- ❖ **Registro de usuario para el ingreso en el sistema:** Con la definición de la estructura de base de datos realizada en el sprint0, se procedió a crear un módulo para el ingreso de usuarios en el sistema. Además, se desarrolló una interfaz que permita visualizar el registro de un usuario. La interfaz debe tener la siguiente información: nombre de usuario, correo electrónico y contraseña.

Figura 2.9: Pantalla de registro de usuario

- ✧ **Iniciar sesión como usuario:** Esta historia de usuario consta de solo una tarea, que es el desarrollo de una interfaz que permita visualizar el ingreso de un usuario registrado al sistema. Para llevar a cabo esta tarea, es necesario concluir con las tareas anteriores mencionadas. La interfaz muestra la información del usuario que se necesita para ingresar al sistema que son: nombre de usuario y contraseña.

Figura 2.10: Pantalla de inicio de sesión

- ✧ **Administrar proyectos (crear, editar y eliminar):** La administración de proyectos consiste en dos tareas. La primer tarea es crear un módulo para la administración de proyectos, es decir, que debe registrar un proyecto en el sistema. La segunda tarea consiste en crear funciones que permitan al usuario crear, editar y eliminar un proyecto guardado en el sistema.

APLICATIVO PARA LA VALIDACIÓN DE REQUERIMIENTOS DE SOFTWARE lesliediaz

Proyectos

Gestión de Proyectos

Crear

 Buscar:

Identificador	Nombre	Descripción	Información Relevante	Opciones
PJ77	Prueba - Copia	Prueba	Estado: En Proceso Tipo de Proyecto: Juego Serio Es un proyecto duplicado	
PJ63	Prueba	Prueba	Estado: Finalizado Tipo de Proyecto: Juego Serio	
PJ55	Requerimiento	Requerimiento	Estado: Finalizado Tipo de Proyecto: Juego Serio	

<< < 1 > >>

© 2022 Escuela Politécnica Nacional. Todos los derechos reservados

Figura 2.11: Pantalla de administración de proyectos

❖ Sprint Review

Se cumplió el objetivo del sprint que consiste en registrar e iniciar sesión como usuario en el sistema y en la administración de proyectos. Los criterios de aceptación establecidos se cumplieron satisfactoriamente como se muestra en la tabla 2.20.

Código	Historia de Usuario	Tarea	Criterios de Aceptación	Cumplido
HU01	Registro de usuario para el ingreso en el sistema	Crear un módulo para el ingreso de usuarios en el sistema	Registro de un usuario en el sistema	Si
		Desarrollar la interfaz que permita visualizar el registro de un usuario	Mostrar en la interfaz la información del usuario (nombre, correo electrónico, contraseña).	Si
HU03	Iniciar sesión como usuario	Desarrollar la interfaz que permita visualizar el ingreso de un usuario al sistema	Mostrar en la interfaz la información de usuario para ingresar al sistema (correo electrónico, contraseña).	Si
HU04	Administrar Proyectos (crear, editar y eliminar)	Crear un módulo para la administración de proyectos	Registro de un proyecto en el sistema.	Si

		Crear las funciones que permitan crear, editar y eliminar un registro (proyecto).	Crear, editar y eliminar un proyecto.	Si
--	--	---	---------------------------------------	----

Tabla 2.20: Sprint 1 Review

❖ **Sprint Retrospective**

La figura 2.12 se muestra el Burndown Chart del avance del sprint.

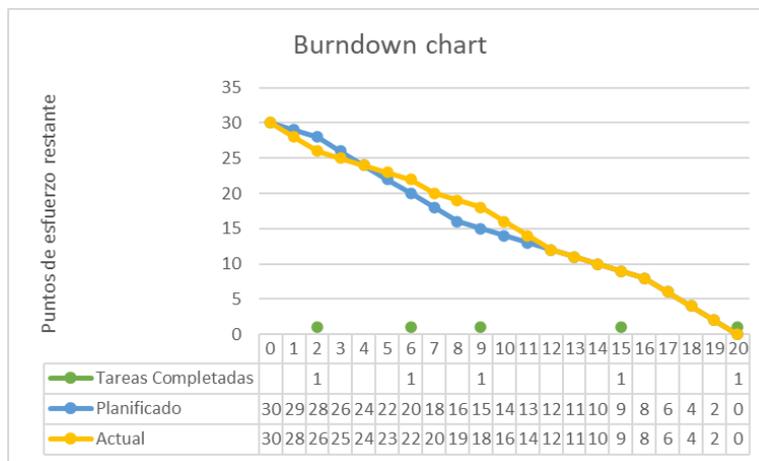


Figura 2.12: Burndown chart sprint 1

2.3.3 Sprint 2

❖ **Objetivos del Sprint**

Ingresar requerimientos por medio de un archivos xlsx y seleccionar las propiedades de un requerimiento bien formado.

❖ **Product Backlog**

Las historias de usuario y las tareas que se ejecutaron durante este sprint se muestran en la tabla 2.21.

Código	Historia de Usuario	Tarea	Criterios de Aceptación
HU05	Ingresar requerimientos por medio de cargar archivos formato .xlsx	Desarrollar la interfaz que permita seleccionar el archivo xlsx que se va a almacenar.	Mostrar en la interfaz el nombre del archivo xlsx seleccionado para cargar.
		Extraer la información del archivo xlsx, en una cadena de caracteres.	Registro de la cadena de caracteres correspondiente al archivo seleccionado.
HU10	Seleccionar las propiedades de requerimientos bien formados.	Desarrollar la interfaz que permita seleccionar las propiedades de un requerimiento bien formado.	Mostrar en la interfaz una tabla con las propiedades de un requerimiento bien formado.
		Registrar las propiedades seleccionadas.	Registro de las propiedades seleccionadas.

Tabla 2.21: Sprint 2 Backlog

❖ **Requisitos completados del Sprint**

El objetivo del sprint 2 consiste en ingresar requerimientos por medio de un archivos xlsx y seleccionar las propiedades de un requerimiento bien formado. A continuación, se describe la ejecución de las tareas correspondientes a las siguientes historias de usuario:

- ❖ **Ingresar requerimientos por medio de cargar archivos formato .xlsx:** La carga de archivo xlsx, está constituido por dos tareas fundamentales. La primera tarea es desarrollar la interfaz, esta debe permitir seleccionar el archivo en formato xlsx y debe mostrar en la interfaz el nombre del archivo. La segunda tarea consistes en extraer la información del archivo cargado en una cadena de caracteres. Esta tarea debe tener como resultado un registro de la cadena de caracteres correspondiente al archivo cargado.

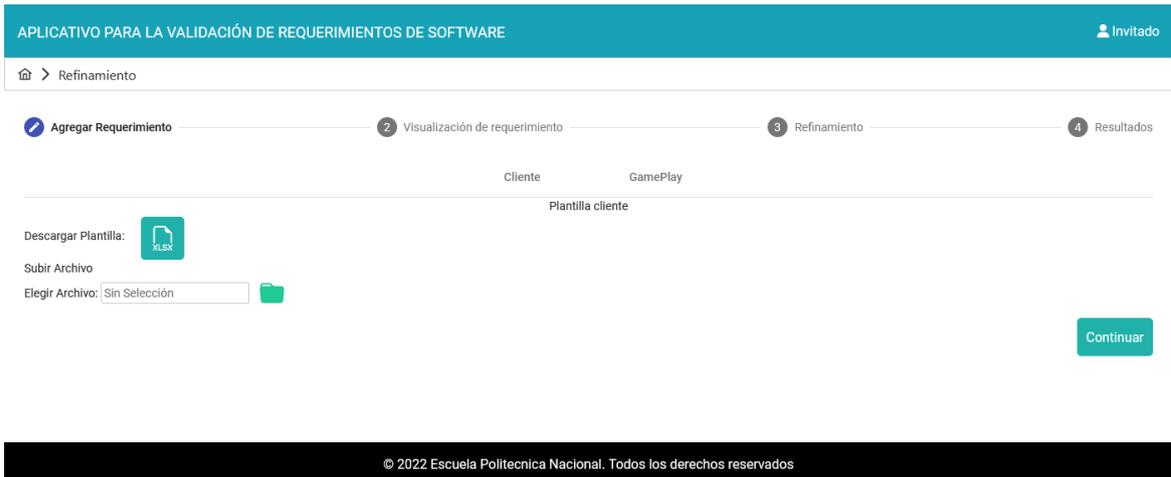


Figura 2.13: Pantalla de subida de archivo plantilla

- ❖ **Seleccionar las propiedades de requerimientos bien formados:** Para cumplir con esta historia de usuario se debe desarrollar una interfaz que permita seleccionar las propiedades de un requerimiento bien formado, para ello la interfaz debe tener una tabla que contenga dichas propiedades. Además, se debe registrar las propiedades que se seleccionaron, por ende, debe tener un registro de esta selección.

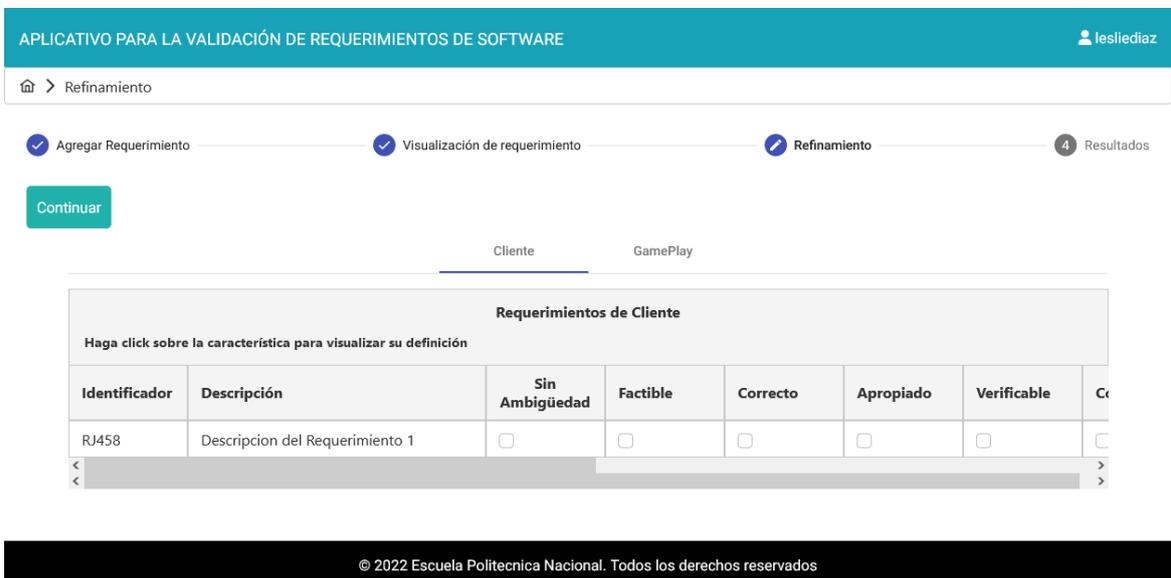


Figura 2.14: Pantalla de refinamiento de requerimientos

❖ Sprint Review

Los criterios de aceptación establecidos se cumplieron satisfactoriamente como se muestra en la tabla 2.22.

Código	Historia de Usuario	Tarea	Criterios de Aceptación	Cumplido
HU05	Ingresar requerimientos por medio de cargar archivos formato .xlsx	Desarrollar la interfaz que permita seleccionar el archivo xlsx que se va a almacenar.	Mostrar en la interfaz el nombre del archivo xlsx seleccionado para cargar.	Si
		Extraer la información del archivo xlsx, en una cadena de caracteres.	Registro de la cadena de caracteres correspondiente al archivo seleccionado.	Si
HU10	Seleccionar las propiedades de requerimientos bien formados.	Desarrollar la interfaz que permita seleccionar las propiedades de un requerimiento bien formado.	Mostrar en la interfaz una tabla con las propiedades de un requerimiento bien formado.	Si
		Registrar las propiedades seleccionadas.	Registro de las propiedades seleccionadas.	Si

Tabla 2.22: Sprint 2 Review

❖ Sprint Retrospective

La figura 2.15 se muestra el Burndown Chart del avance del sprint.

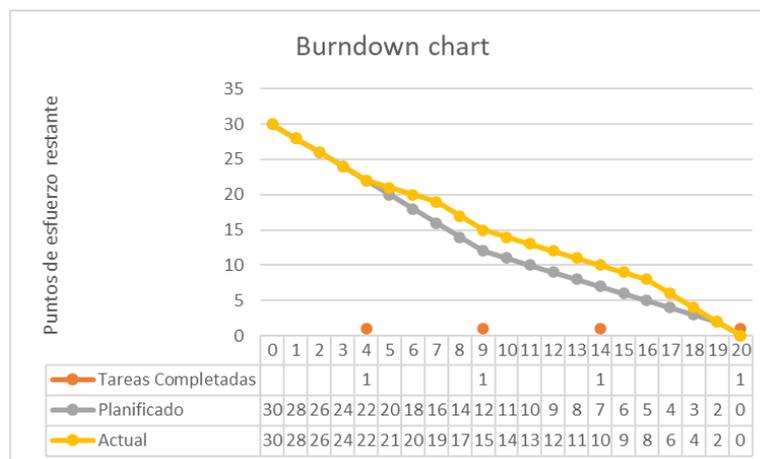


Figura 2.15: Burndown chart sprint 2

2.3.4 Sprint 3

❖ Objetivos del Sprint

Administrar participantes y exportar resultados.

❖ **Product Backlog**

Las historias de usuario y las tareas que se ejecutaron durante este sprint se muestran en la tabla 2.23.

Código	Historia de Usuario	Tarea	Criterios de Aceptación
HU08	Administrar participantes (crear, editar y eliminar)	Desarrollar la interfaz que permita administrar participantes ingresados.	Mostrar en la interfaz los datos de participantes y botones de acceso para administrar los participantes.
		Crear las funciones que permitan crear, editar y eliminar participantes.	Crear, editar y eliminar participantes.
HU09	Exportar resultados.	Obtener una matriz resultante booleana.	Mostrar en la interfaz una tabla con los resultados.
		Crear una función que permita exportar las tablas resultantes del proceso.	Descarga del archivo con los resultados obtenidos.

Tabla 2.23: Sprint 3 Backlog

❖ **Requisitos completados del Sprint**

El objetivo del sprint 3 consiste en administrar participantes y exportar resultados obtenidos. A continuación, se describe la ejecución de las tareas correspondientes a las siguientes historias de usuario:

- ❖ **Administrar participantes (crear, editar y eliminar):** Esta historia de usuario tiene una prioridad baja, sin embargo, se la ha dividido en dos tareas que son esenciales para todo el sistema. La primera tarea es desarrollar una interfaz que permita administrar participantes, es decir que se debe mostrar los datos del participante, además esta interfaz debe tener botones de acceso para realizar la respectiva administración.

La segunda tarea consiste en crear las funciones que permitan crear, editar y eliminar participantes, estas funciones deben estar conectados con la interfaz

realizada con anterioridad.

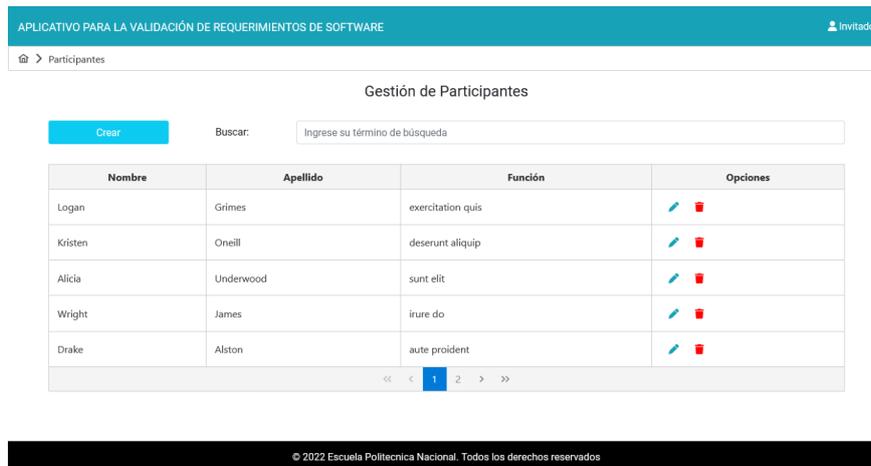


Figura 2.16: Pantalla de administración de participantes

❖ **Exportar resultados:** Para realizar esta historia de usuario, primero se dividió en tareas pequeñas las cuales constan de la obtención de resultados en una matriz y la creación de una función que exporte estos resultados. La matriz que se obtenga será booleana y se mostrará mediante una interfaz que tenga una tabla con estos resultados.

Por otra parte, la función creada anteriormente para exportar los resultados debe descargar el archivo con los resultados obtenidos y mostrados en la interfaz desarrollada en la tarea anterior.

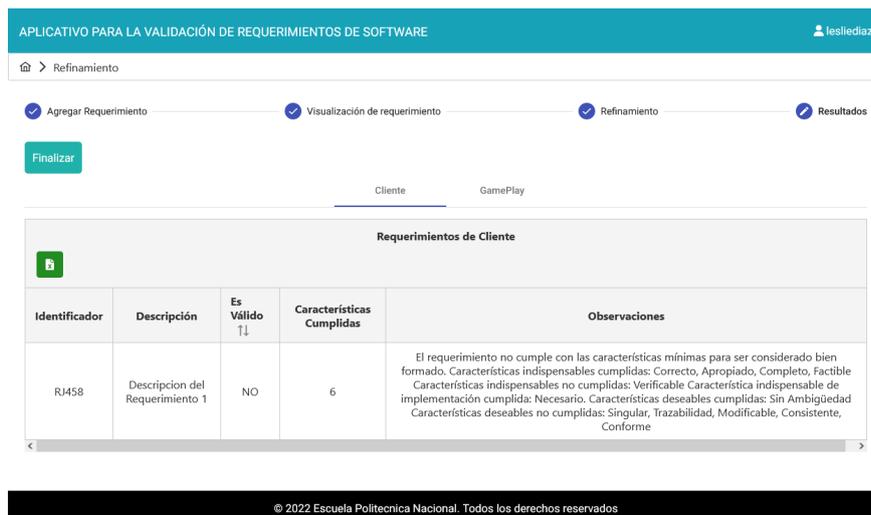


Figura 2.17: Pantalla de resultados del refinamiento de requerimientos

❖ Sprint Review

Los criterios de aceptación establecidos se cumplieron satisfactoriamente como se muestra en la tabla 2.24.

Código	Historia de Usuario	Tarea	Criterios de Aceptación	Cumplido
HU08	Administrar participantes (crear, editar y eliminar)	Desarrollar la interfaz que permita administrar participantes ingresados.	Mostrar en la interfaz los datos de participantes y botones de acceso para administrar los participantes.	Si
		Crear las funciones que permitan crear, editar y eliminar participantes.	Crear, editar y eliminar participantes.	Si
HU09	Exportar resultados.	Obtener una matriz resultante booleana.	Mostrar en la interfaz una tabla con los resultados.	Si
		Crear una función que permita exportar las tablas resultantes del proceso.	Descarga del archivo con los resultados obtenidos.	Si

Tabla 2.24: Sprint 3 Review

❖ Sprint Retrospective

La figura 2.18 se muestra el Burndown Chart del avance del sprint.

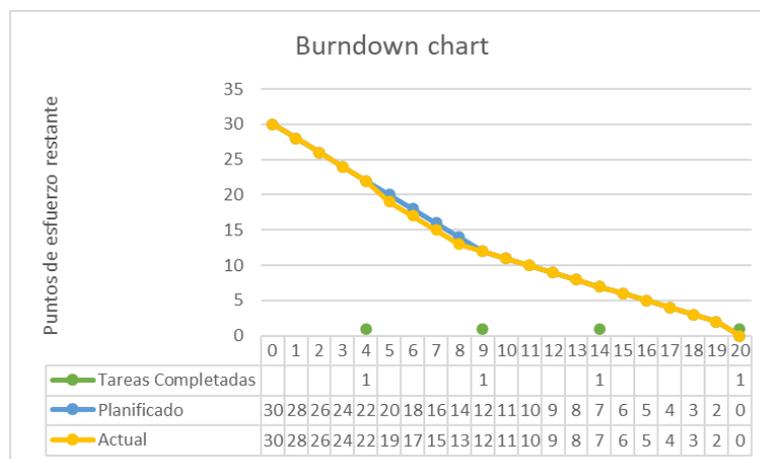


Figura 2.18: Burndown chart sprint 3

2.4 FASE DE LANZAMIENTO

En esta sección se describirá el proceso realizado para el despliegue del presente proyecto en un ambiente de producción, así como las lecciones aprendidas a raíz del desarrollo de este.

2.4.1 Implementación en un ambiente de producción.

Amazon Web Service (AWS) incluye varios servicios dedicados a un área de funcionalidad, uno de ellos es Amazon Elastic Compute Cloud (Amazon EC2) el cual proporciona recursos de computación escalables basados en web [41]. En este contexto, Amazon Elastic Beanstalk es un servicio utilizado para implementar aplicaciones web desarrolladas en diferentes entornos de ejecución, en este caso, Node.js [42]. Es necesaria la configuración de una aplicación y un entorno, definido por AWS como una colección de recursos necesarios para ejecutar el código fuente de la aplicación [43], para su despliegue en producción siendo Elastic Beanstalk el encargado de administrar la misma de manera automática.

De la misma manera, se utilizó el servicio de bases de datos relacionales administrados de AWS denominado Amazon RDS, que facilita las tareas de configuración, operación y escalado de una base de datos en la nube [44]. Se definió como acceso público la instancia de base de datos teniendo credenciales de usuario y un punto de enlace para que esta sea accesible desde la aplicación web. Para la creación del esquema de la base de datos, se conectó previamente al punto de enlace a través de la herramienta MySQL Workbench 8 CE.

Para este proyecto se configuró un entorno en una plataforma que utiliza Node.js versión 16 con una instancia de Amazon Linux 2 de 64bits. En lo que respecta a la configuración de inicio de la aplicación web se agregó el archivo Procfile especificando el comando necesario para su inicio. Asimismo, se definieron como variables de entorno al puerto y la URL del servidor que aloja a la base de datos. Dentro de la aplicación se acceden a dichas variables mediante `process.env.NOMBRE_VARIABLE`.

Si el levantamiento de la aplicación se realiza correctamente, el estado del entorno pasará a establecerse como OK y la aplicación estará lista para usarse mediante el enlace generado por el entorno.

2.4.2 Retrospectiva del proyecto.

Pregunta	Respuesta
¿Qué salió bien?	<ul style="list-style-type: none">❖ El uso de Node.js para el desarrollo de servidor web facilitó la integración de este y la parte de cliente al utilizar en ambos proyectos el lenguaje de programación JavaScript.❖ El uso de marcos de trabajo conocidos como Angular.js y Nest.js permitió agregar paquetes de forma fácil gracias al administrador de paquetes de Node utilizado en ambos. Dichos paquetes facilitaron la administración de la base de datos y la presentación de la información al usuario final.
¿Qué salió mal?	<ul style="list-style-type: none">❖ El diseño de la interfaz tuvo que ser modificado a medida que se desarrollaba el proyecto debido a cambios en la presentación de la información al usuario final. De la misma manera, algunas rutas y funcionalidades de la aplicación fueron desechadas.❖ La presentación de la información, en algunos casos, tuvo inconvenientes de tipo estético por lo que se limitó el número de datos que el usuario podía visualizar.❖ Algunos métodos construidos no se encuentran debidamente optimizados por lo que la aplicación puede tener problemas de fluidez y afectar a la experiencia del usuario.

<p>¿Qué obstáculos se presentaron?</p>	<ul style="list-style-type: none"> ❖ La situación sanitaria global que se vivió en el transcurso del desarrollo de este proyecto. ❖ Impedimento de una adecuada ejecución de pruebas de usabilidad debido a la situación sanitaria y económica del país. ❖ Dificultad en la recolección normas, estándares y documentos relacionados para la investigación realizada. El acceso a estos fue limitado o en algunos casos, no se encontraron. Además, se realizaron traducciones al español de algunas normas y/o estándares encontrados en idiomas no conocidos.
<p>Lecciones Aprendidas</p>	<ul style="list-style-type: none"> ❖ Realizar un prototipo funcional de las interfaces a implementar, así como un mapa de sitio aprobado por el dueño del producto antes de empezar el desarrollo de la aplicación. ❖ El uso de frameworks basados en JavaScript facilita la implementación de aplicaciones escalables y permiten el ahorro de recursos. ❖ El uso de librerías tales como Angular Material o PrimeNG facilitan la presentación de la información al usuario a fin de evitar la saturación de elementos en la pantalla y mejorar su experiencia.

Tabla 2.25: Retrospectiva del proyecto

2.5 RESUMEN DEL CAPÍTULO

En el capítulo 2 "Metodología" se presentan cuatro secciones. La primera sección describe la metodología utilizada llamada Scrum, la que nos permite obtener requerimientos para el aplicativo web.

La segunda sección del capítulo presenta la fase inicial de Scrum, donde se definen los requerimientos de la aplicación, que dio como resultado 3 historias de usuario épicas las cuales se desglosan en 10 historias de usuario. Además, en esta sección se identifican los

roles que son: Mayra Carrión como la propietaria del producto, Marco Santorum como el scrum máster, Leslie Díaz y Pamela Pachacama como el equipo de desarrollo.

La tercera sección presenta la implementación de la aplicación en cuatro Sprints. Cada Sprint contiene objetivos, Product Backlog, requisitos completados, Sprint Review y Sprint Retrospective. Por ende, cada Sprint muestra el avance alcanzado y el cumplimiento de los criterios de aceptación.

Finalmente, la cuarta sección presenta la fase de lanzamiento de la aplicación. En esta sección se describe la implementación del aplicativo en un ambiente de producción usando los servicios de AWS. A su vez, se presenta la retrospectiva de lo que salió bien, mal, los obstáculos que se presentaron y las lecciones aprendidas en el desarrollo del proyecto.

3 APLICATIVO PARA LA VALIDACIÓN DE REQUERIMIENTOS DE SOFTWARE

En esta sección se describirá la funcionalidad de la aplicación web y su implicación en el propósito del presente proyecto, la arquitectura general de la aplicación, así como las herramientas y los recursos utilizados para el desarrollo de esta.

3.1 APLICATIVO PARA LA VALIDACIÓN DE REQUERIMIENTOS DE SOFTWARE

El presente proyecto contempla el desarrollo de una aplicación web para la validación de requerimientos de software de proyectos enfocados o no en el desarrollo de juegos serios. Para ello se emplea una matriz de refinamiento, herramienta utilizada en la fase 5 de la metodología iPlus propuesta en [2] la cual fue rediseñada a partir de la investigación realizada y previa a la construcción de este aplicativo. Asimismo, para la definición de un requerimiento bien formado se utilizan las métricas resultantes obtenidas de la aplicación web desarrollada por Ronnie Nieto [28] que está basada en la metodología para la inferencia de criterios clave y que efectúa la inferencia de criterios que se consideran necesarios y/o suficientes.

El propósito del desarrollo de la plataforma web es automatizar la fase 5 de la metodología iPlus lo que implica la validación en la escritura de los requerimientos a través de plantillas, su posterior evaluación según el cumplimiento de características previamente definidas y en base a la opinión del evaluador y la generación de observaciones que contemplen las razones del porqué dicho requerimiento se considere bien formado o no. Todo este proceso busca que el refinamiento de requerimientos sea más fluido y sea mucho más fácil identificar los requerimientos que necesitan ser revisados, mejorados o que simplemente cumplen con las características necesarias o deseables.

Como resultado de la ejecución de Scrum, se obtiene la aplicación web nombrada “Aplicativo para la Validación de Requerimientos de Software - AVaRS”. La aplicación web contiene 4 módulos que son: módulo de usuario, módulo de proyectos, módulo de participantes y módulo de refinamiento, siendo este último el núcleo de toda la aplicación.

❖ **Módulo de Usuario**

El módulo de usuario contempla las funciones de inicio de sesión, registro en la aplicación y visualización del perfil del usuario.

❖ **Módulo de Proyectos**

El módulo de proyectos contienen la administración de los proyectos, estos pueden ser de tipo cliente, para el cual no se mostrarán ciertas pantallas del aplicativo, y juego serio. Dentro de su gestión se podrá crear un nuevo proyecto, editar la información de este, duplicar el proyecto en el caso de que se encuentre en estado finalizado, eliminar el proyecto, así como descargar los resultados de la validación.

❖ **Módulo de Participantes.**

El módulo de gestión de participantes permite administrar a los participantes de un proyecto de software, se puede añadir un nuevo participante, editar su información y eliminarlo.

❖ **Módulo de Refinamiento**

El módulo de refinamiento es el núcleo de la aplicación y el en cual se contempla todo el proceso de recolección, refinamiento y validación de requerimientos de software. La primera parte permite escoger como se va a ingresar los requerimientos a ser refinados, para ello se definen 2 métodos: plantilla o gráficamente. Si el usuario escoge la opción plantilla se mostrará la pantalla para la descarga de la plantilla y la opción para subir la plantilla con los requerimientos que desee refinar. En el caso de escoger el método gráfico, se mostrará la pantalla respectiva para ingresar los requerimientos de forma gráfica, así como su gestión respectiva (editar y eliminar).

Para el ingreso de requerimientos en un proyecto de juego serio, se tiene 2 tipos de métodos: El ingreso normal, el cual cuenta con la opción de añadir propósitos y el ingreso de

relatos GamePlay, con una estructura similar a los requerimientos normales además de la opción de registrar los bloques GamePlay ligados al relato.

Una vez agregados los requerimientos se continua al apartado de refinamiento que, dependiendo del tipo de proyecto, el orden de las características a evaluar cambia, así como la descripción de cada una.

Para finalizar, la sección de resultados que muestra en una tabla los requerimientos que han pasado la validación, así como lo que no lo hicieron, además de las observaciones respectivas en cada caso.

3.2 ARQUITECTURA, HERRAMIENTAS Y RECURSOS UTILIZADOS

En esta sección se presenta de manera detallada el patrón modelo vista controlador, la arquitectura general de la aplicación, y las herramientas y recursos utilizados.

3.2.1 Patrón Modelo-Vista-Controlador (MVC)

MVC es un patrón de diseño que considera dividir una aplicación en tres módulos identificables y con funcionalidad definida: El modelo, las vistas y el controlador.

- ❖ El **modelo** es un conjunto de clases que representan la información del mundo real que el sistema debe procesar, sin tomar en cuenta ni la forma en la que esa información va a ser mostrada ni los mecanismos que hacen que esos datos estén dentro del modelo, es decir, sin tener relación con ninguna otra entidad dentro de la aplicación. El modelo desconoce la existencia de las vistas y del controlador.
- ❖ Las **vistas** son el conjunto de clases que se encargan de mostrar al usuario la información contenida en el modelo. Una vista está asociada a un modelo, pudiendo existir varias vistas asociadas al mismo modelo.
- ❖ El **controlador** es un objeto que se encarga de dirigir el flujo del control de la aplicación debido a mensajes externos, como datos introducidos por el usuario u opciones del menú seleccionadas por él. A partir de estos mensajes, el controlador se encarga de modificar el modelo o de abrir y cerrar vistas. El controlador tiene acceso al modelo

y a las vistas, pero las vistas y el modelo no conocen de la existencia del controlador [45].

En la figura 3.1 se puede observar la interacción de los módulos existentes en el patrón MVC.

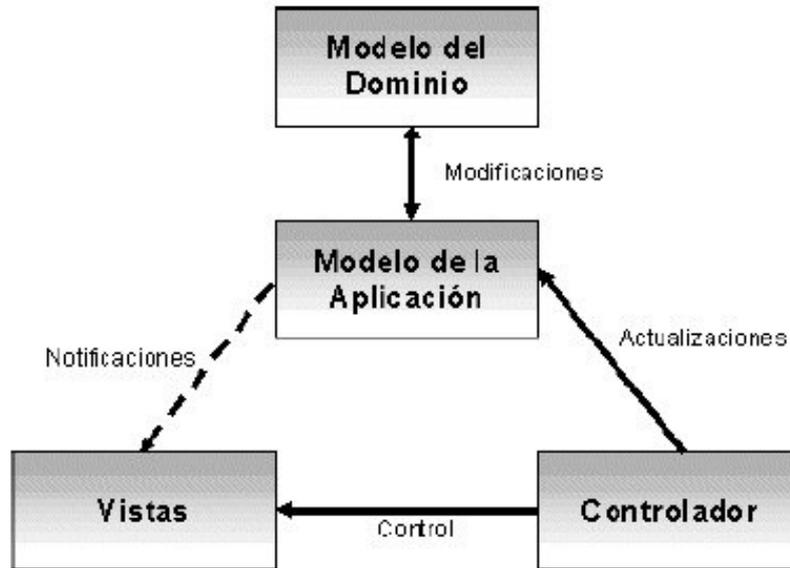


Figura 3.1: Relación entre los módulos del patrón MVC

3.2.2 Arquitectura general de la aplicación

En la figura 3.2 se muestra la arquitectura de la aplicación basada en una arquitectura de una aplicación web tradicional y un modelo de programación organizado en tres capas: la capa de presentación o frontera, la capa de lógica de negocio o control y la capa de datos; módulos que pueden ser tratados de manera independiente y hasta paralela asegurando avanzar de manera ordenada en la programación.

- ❖ La capa de **presentación** debe manejar interfaces que faciliten al usuario la interacción con la aplicación.
- ❖ En la capa de **lógica de negocio** se definen las reglas que deben cumplirse para una correcta ejecución del programa, se encuentra la lógica del programa, estructuras de datos, procesamiento de la información ingresada o solicitada desde la capa de presentación.

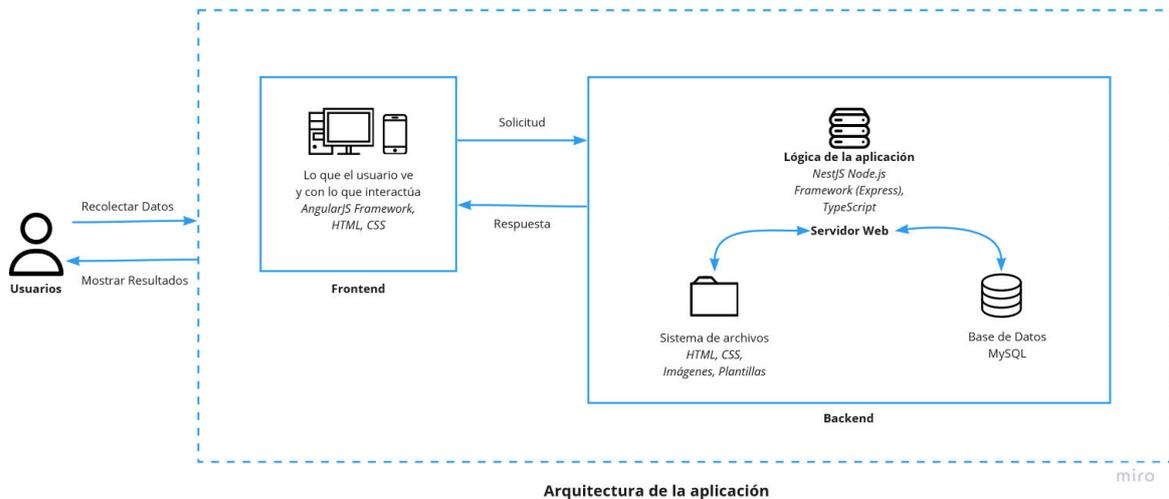


Figura 3.2: Arquitectura general de la aplicación

- ❖ La capa de **datos** es la encargada de realizar transacciones con bases de datos y otros sistemas para obtener o ingresar información al sistema. [46]

3.2.3 Herramientas y recursos utilizados

Para el desarrollo del presente proyecto se utilizaron las herramientas y recursos listados en la tabla 3.1 mostrada a continuación.

Logo	Nombre	Descripción
Diseño		
	Figma	Figma es una herramienta de prototipo web y editor de gráficos vectoriales, que se aloja en la web [47].
Desarrollo y Base de datos		
	JavaScript	JavaScript (JS) es un lenguaje de programación ligera, interpretado, o compilado justo-a-tiempo (just-in-time) con funciones de primera clase [48].

	Angular	Angular es una plataforma de desarrollo, construido sobre TypeScript incluye un marco basado en componentes, una colección de bibliotecas y un conjunto de herramientas de desarrollo [49].
	Node.js	Node.js es un entorno de tiempo de ejecución de JavaScript, fue creado por los desarrolladores originales de JavaScript. Está diseñado para crear aplicaciones de red escalables [50].
	Nest	Nest (NestJS) es un marco para crear aplicaciones del lado del servidor Node.js eficientes y escalables. Utiliza JavaScript progresivo, está construido con TypeScript [51].
	MySQL	MySQL es un sistema de gestión de bases de datos relacionales (RDBMS) de código abierto respaldado por Oracle y basado en el lenguaje de consulta estructurado (SQL) [52].
	Webstorm	WebStorm es un entorno de desarrollo integrado para JavaScript y las tecnologías relacionadas [53].

	<p>Docker</p>	<p>Docker es una plataforma de software que le permite crear, probar e implementar aplicaciones rápidamente. Empaqueta software en unidades estandarizadas llamadas contenedores que incluyen todo lo necesario para que el software se ejecute [54].</p>
	<p>DBeeer</p>	<p>DBeeer es una aplicación de software cliente de SQL y una herramienta de administración de bases de datos [55].</p>
<p>Control de versiones</p>		
	<p>Git</p>	<p>Git es un sistema de control de versiones distribuido gratuito y de código abierto [56].</p>
	<p>GitHub Desktop</p>	<p>GitHub Desktop es una aplicación que te habilita para interactuar con GitHub utilizando una GUI en vez de la línea de comandos o de un buscador web [57].</p>
	<p>GitHub</p>	<p>GitHub es un repositorio online gratuito que permite gestionar proyectos y controlar versiones de código [58].</p>

	<p>GitKraken</p>	<p>GitKraken es una herramienta multi plataforma (Mac, Windows y Linux) que nos ayuda a manejar Git de manera sencilla [59].</p>
---	------------------	--

Tabla 3.1: Herramientas y Recursos

3.3 CASO DE ESTUDIO

Los requerimientos en la elaboración de software influyen de manera primordial, como se detalló con anterioridad en el punto 1.1 de este documento. Por ende, se consideró un proyecto desarrollado con la metodología iPlus, del cual la segunda y cuarta fase serán los puntos indispensables de este caso de estudio.

Aplicación de la herramienta para el contexto de estudio.

En esta sección se describe el uso del aplicativo web en el caso de estudio mencionado. Los pasos iniciales son los siguientes:

- ❖ **Ingresar a la aplicación.** Al ingresar a la aplicación, primero se debe crear una cuenta, es decir que debe ingresar un nombre de usuario, un correo electrónico y una contraseña.



Figura 3.3: Registro de usuario.

- ❖ **Interfaz inicial.** Una vez creada la cuenta, se muestra la interfaz inicial, donde se encuentra la metodología, el manual de usuario y la aplicación desarrollada como se muestra en la figura 3.4.

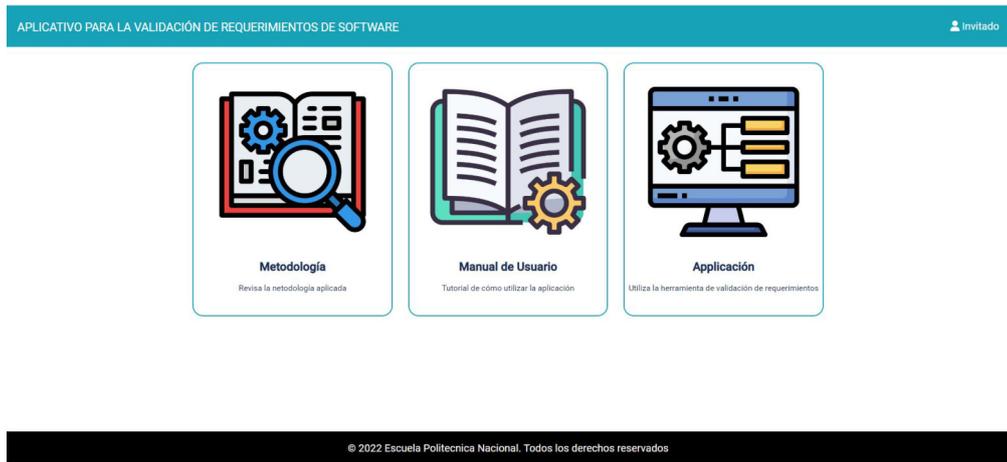


Figura 3.4: Interfaz inicial.

- ❖ **Aplicación para la validación de requerimientos de software.** Primero se debe elegir el módulo de la aplicación web, después se continua con el método por el cual se quiere ingresar los datos. Para este caso de estudio se seleccionó el método de plantillas, por ende, se descargó las plantillas y se llenó con los resultados de la segunda y tercera fase de la metodología iPlus desarrollada en el proyecto de titulación de Betancourt C. [60].

Las plantillas fueron llenadas para ser matrices booleanas donde 0 es el valor de NO y 1 es el valor de Sí. Estos valores fueron proporcionados por el desarrollador del proyecto "LAS JOYAS DE MI TIERRA"[60], como se muestran en la figura 3.5 y la figura 3.6.

Identificador	Descripción	SIN AMBIGÜEDAD	FACTIBLE	CORRECTO	APROPIADO	VERIFICABLE	COMPLETO	NECESARIO	SINGULAR	CONFORME	CONSISTENTE	MODIFICABLE	TRAZABILIDAD
H1-01	Se registrará la interacción con la aplicación, progreso, características, tiempo.(3)	1	1	1	1	1	1	1	1	1	1	1	1
H1-02	La aplicación deberá registrar el puntaje obtenido por el estudiante al realizar las actividades de cada cultura.	1	1	1	1	1	1	1	1	1	1	1	1
H2-01	Rol estudiante y rol tutor que enseña y tiene actividades de evaluación.(9)	1	0	1	0	1	1	0	0	0	0	0	0
H2-02	El tutor deberá evidenciar el progreso de todos los participantes.	1	1	1	1	1	1	1	1	1	1	1	1
H3-01	Aprender sobre cultura y conciencia sobre la naturaleza accesible a todos.(1)	0	0	1	0	0	0	0	0	0	0	0	0
H4-01	Aprender sobre vestimenta.(4)	0	0	1	0	0	0	0	0	0	0	0	0
H4-02	La aplicación deberá permitir vestir personajes por cultura o nacionalidad del Ecuador	1	1	1	1	1	1	1	1	1	1	1	1
H5-01	Aprender dialectos. (7)	1	0	1	0	0	0	0	0	0	0	0	0
H6-01	Aprender las diferencias entre pueblos y nacionalidades.(5)	0	0	0	0	0	0	0	0	0	0	0	0
H7-01	Interactuar con un avatar real que se mueve y se viste en un entorno real de acuerdo con su entorno.(6)	0	0	0	0	0	0	0	0	0	0	0	0
H7-02	El participante podrá personalizar su avatar agregando o quitando prendas de vestir	1	1	1	1	1	1	1	1	1	1	1	1
H7-03	El participante podrá personalizar su avatar agregando o quitando habilidades correspondientes a las	1	1	1	1	1	1	1	1	1	1	1	1
H8-01	Rol administrador.(5)	0	0	0	0	0	0	0	0	0	0	0	0
H8-02	El administrador podrá agregar una cultura al juego para que sea visiblemente en las actividades de juego y evaluación.	1	1	1	1	1	1	1	1	1	1	1	1

Figura 3.5: Plantilla requerimientos cliente.

Identificador	Descripción	COMPLETO	APROPIADO	CORRECTO	NECESARIO	VERIFICABLE	FACTIBLE	SIN AMBIGÜEDAD	MODIFICABLE	SINGULAR	TRAZABILIDAD	CONSISTENTE	CONFORME
RI 01	El jugador podrá vestir a la cuca y obtener mayor puntaje en caso de cometer menos errores en la	1	1	1	1	1	1	1	1	1	0	1	1
RI 02	El participante podrá hacer bailar la cuca una vez acabada de vestir	0	0	1	0	0	0	1	0	1	0	0	0
RI 03	Las distintas cucas vestidas aparecerán y el jugador las podrá emparejar con su localización.	1	0	1	0	0	1	1	1	0	1	1	1
RI 04	El jugador tendrá una guía de lo que va a hacer en el juego e información de las culturas.	0	1	1	0	1	1	1	1	0	1	1	1
RI 05	El jugador podrá realizar un test de cada cultura, lo cual, al responder bien le permitirá obtener puntos.	1	1	1	1	1	1	1	1	0	1	1	1
RI 06	En el juego, el jugador se someterá a una evaluación al final de cada nivel, lo cual, si obtiene la base del puntaje le permitirá pasar al siguiente nivel.	0	0	1	0	1	1	1	0	1	1	1	0
RI 07	El jugador tendrá desafíos durante la visita de cada región, y al completarlos obtendrá una insignia.	1	1	1	1	1	1	1	1	0	1	1	1
RI 08	Evita colocar la vestimenta en el orden incorrecto.	0	0	0	0	0	0	0	0	1	0	0	0
RI 08 01	El jugador deberá evitar colocar vestimentas incorrectas a las culturas, ya que, reducirá su experiencia y su tabla de puntuación.	1	1	1	0	1	1	1	1	0	1	1	1
RI 09	El asistente de inicio indicará el rendimiento su experiencia y su tabla de puntuación.	0	0	1	0	0	0	1	1	0	1	1	0
RI 10	El jugador podrá asignar habilidades al personaje al colocarle insignias obtenidas.	1	1	1	1	1	1	1	1	0	0	1	1
RI 11	Retos más difíciles al avanzar niveles superiores.	0	0	1	0	0	0	1	0	1	0	0	0
RI 12	El jugador podrá vestir al avatar con las vestimentas disponibles para el jugador.	1	1	1	1	1	1	1	1	0	1	1	1
RI 13	Arrastrar las piezas de vestir sobre el avatar.	0	0	1	0	0	0	1	0	1	0	0	0
RI 14	El estudiante responde las preguntas acerca de un tema mientras el avatar lo ayuda	1	1	1	1	1	1	1	1	0	1	1	1
RI 15	El personaje se mueve en el país.	0	0	1	1	1	1	1	1	0	1	1	1
RI 16	El personaje podrá desbloquear provincias del Ecuador al terminar las actividades de las culturas de la provincia actual.	1	1	1	1	1	1	1	1	0	1	1	1
RI 17	El jugador podrá jugar al ahorcado, respondiendo a una palabra representativa de la cultura para obtener	1	1	1	1	1	1	1	1	0	1	1	1
RI 18	El jugador podrá obtener vestimentas de cada cultura, al terminar todas las actividades de la misma.	1	1	1	1	1	1	1	1	0	1	1	1
RI 19	El jugador podrá realizar trueques con los puntos obtenidos al completar las actividades.	1	1	1	1	1	1	1	1	0	1	1	1

Figura 3.6: Plantilla GamePlay.

- ❖ **Refinamiento.** Antes del paso de refinamiento, se tiene la posibilidad de agregar más requerimientos, pero en este caso pasamos directo al refinamiento. En el refinamiento existe la posibilidad de cambiar los datos ingresados en la matriz booleana, es decir, que podemos cambiar los valores ingresados en la plantilla.

Figura 3.7: Refinamiento de requerimientos

- ❖ **Resultados.** Los resultados obtenidos se muestran en el módulo de resultados. Estos pueden ser descargados y son mostrados en una tabla en la cual se especifican

cuantas propiedades cumplen con ser: necesarias y no suficientes, no necesarias y no suficientes, y ninguna. La figura 3.8 muestra los resultados de refinamiento.

APLICATIVO PARA LA VALIDACIÓN DE REQUERIMIENTOS DE SOFTWARE Instituto

Refinamiento

Agregar Requerimiento Visualización de requerimiento Refinamiento Resultados

Finalizar

Cliente GamePlay

Requerimientos de Cliente				
Identificador	Descripción	Es Válido	Características Cumplidas	Observaciones
R473	El administrador podrá agregar una cultura al juego para que sea visible en las actividades de juego y evaluación.	SI	11	El requerimiento cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto, Apropiado, Completo, Verificable, Factible Característica indispensable de implementación cumplida: Necesario. Características deseables cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme Características deseadas no cumplidas: Singular
R471	Rol administrador.(5)	NO	1	El requerimiento no cumple con las características mínimas para ser considerado bien formado. Características indispensables no cumplidas: Correcto, Apropiado, Completo, Verificable, Factible Característica indispensable de implementación no cumplida: Necesario. Características deseables cumplidas: Singular Características deseadas no cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme
R470	Interactuar con un avatar real que se mueva y se vea en un entorno real de acuerdo con su entorno.(6)	NO	1	El requerimiento no cumple con las características mínimas para ser considerado bien formado. Características indispensables no cumplidas: Correcto, Apropiado, Completo, Verificable, Factible Característica indispensable de implementación no cumplida: Necesario. Características deseables cumplidas: Singular Características deseadas no cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme
R469	Aprender las diferencias entre pueblos y nacionalidades.(5)	NO	1	El requerimiento no cumple con las características mínimas para ser considerado bien formado. Características indispensables no cumplidas: Correcto, Apropiado, Completo, Verificable, Factible Característica indispensable de implementación no cumplida: Necesario. Características deseables cumplidas: Singular Características deseadas no cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme
R468	Aprender dialectos. (7)	NO	3	El requerimiento no cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto Características indispensables no cumplidas: Apropiado, Completo, Verificable, Factible Característica indispensable de implementación no cumplida: Necesario. Características deseables cumplidas: Sin Ambigüedad, Singular Características deseadas no cumplidas: Trazabilidad, Modificable, Consistente, Conforme
R467	La aplicación deberá permitir vestir personajes por cultura o nacionalidad del Ecuador	SI	11	El requerimiento cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto, Apropiado, Completo, Verificable, Factible Característica indispensable de implementación cumplida: Necesario. Características deseables cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme Características deseadas no cumplidas: Singular
R466	Aprender sobre vestimenta.(4)	NO	2	El requerimiento no cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto Características indispensables no cumplidas: Apropiado, Completo, Verificable, Factible Característica indispensable de implementación no cumplida: Necesario. Características deseables cumplidas: Singular Características deseadas no cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme
R465	Aprender sobre cultura y conciencia sobre la naturaleza accesible a todos.(1)	NO	2	El requerimiento no cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto Características indispensables no cumplidas: Apropiado, Completo, Verificable, Factible Característica indispensable de implementación no cumplida: Necesario. Características deseables cumplidas: Singular Características deseadas no cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme
R464	El tutor deberá evidenciar el progreso de todos los participantes.	SI	11	El requerimiento cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto, Apropiado, Completo, Verificable, Factible Característica indispensable de implementación cumplida: Necesario. Características deseables cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme Características deseadas no cumplidas: Singular
R463	Rol estudiante y rol tutor que enseña y tiene actividades de evaluación.(9)	NO	5	El requerimiento no cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto, Completo, Verificable Características indispensables no cumplidas: Apropiado, Factible Característica indispensable de implementación no cumplida: Necesario. Características deseables cumplidas: Sin Ambigüedad, Singular Características deseadas no cumplidas: Trazabilidad, Modificable, Consistente, Conforme
R462	El participante podrá personalizar su avatar agregando o quitando habilidades correspondientes a las insignias.	SI	11	El requerimiento cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto, Apropiado, Completo, Verificable, Factible Característica indispensable de implementación cumplida: Necesario. Características deseables cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme Características deseadas no cumplidas: Singular
R461	El participante podrá personalizar su avatar agregando o quitando prendas de vestir	SI	11	El requerimiento cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto, Apropiado, Completo, Verificable, Factible Característica indispensable de implementación cumplida: Necesario. Características deseables cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme Características deseadas no cumplidas: Singular
R460	La aplicación deberá registrar el puntaje obtenido por el estudiante al realizar las actividades de cada cultura.	SI	11	El requerimiento cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto, Apropiado, Completo, Verificable, Factible Característica indispensable de implementación cumplida: Necesario. Características deseables cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme Características deseadas no cumplidas: Singular
R459	Se registrará la interacción con la aplicación, progreso, características, tiempo.(3)	SI	11	El requerimiento cumple con las características mínimas para ser considerado bien formado. Características indispensables cumplidas: Correcto, Apropiado, Completo, Verificable, Factible Característica indispensable de implementación cumplida: Necesario. Características deseables cumplidas: Sin Ambigüedad, Trazabilidad, Modificable, Consistente, Conforme Características deseadas no cumplidas: Singular

© 2022 Escuela Politécnica Nacional. Todos los derechos reservados

Figura 3.8: Resultados de refinamiento

3.4 RESUMEN DEL CAPÍTULO

En el capítulo 3 se presentan tres secciones. La primera sección explica a fondo el aplicativo web para la validación de requerimientos de software, incluye sus características

importantes y detalla su usabilidad.

La segunda sección del capítulo presenta el patrón modelo vista controlador [45], la arquitectura general del aplicativo web, y las herramientas y recursos utilizados para la implementación de Scrum descritos en la sección 2.3.

Finalmente, la tercera sección del capítulo presenta el caso de estudio que se llevó a cabo, donde se presenta el proyecto seleccionado y el uso de la aplicación desarrollada paso a paso.

4 EVALUACIÓN Y RESULTADOS

En esta etapa del proyecto, tras completar el desarrollo del aplicativo web, se realizó dos evaluaciones de usabilidad. La primera evaluación de usabilidad es de diseño y desarrollo basado en las heurísticas de Nielsen [61]. La segunda evaluación de usabilidad es entorno al usuario final, esta se caracteriza por cumplir con el protocolo propuesto por Abhay Rautela [62], permitiendo determinar la conformidad y el grado de usabilidad del aplicativo web.

4.1 EVALUACIÓN DE USABILIDAD CON DISEÑO DE PROTOTIPOS

Se realizaron 4 prototipos de la aplicación para la definición del diseño final de este tomando en cuenta las heurísticas de Nielsen [61] para su diseño. Estos fueron revisados por el dueño del producto a fin de cumplir sus expectativas. El detalle de las interfaces se encontrará en el **ANEXO 11**.

❖ Primer Prototipo

En este prototipo se tenía la idea de automatizar todo el proceso de refinamiento por lo que se optó una interfaz simple y que muestre dicho proceso.

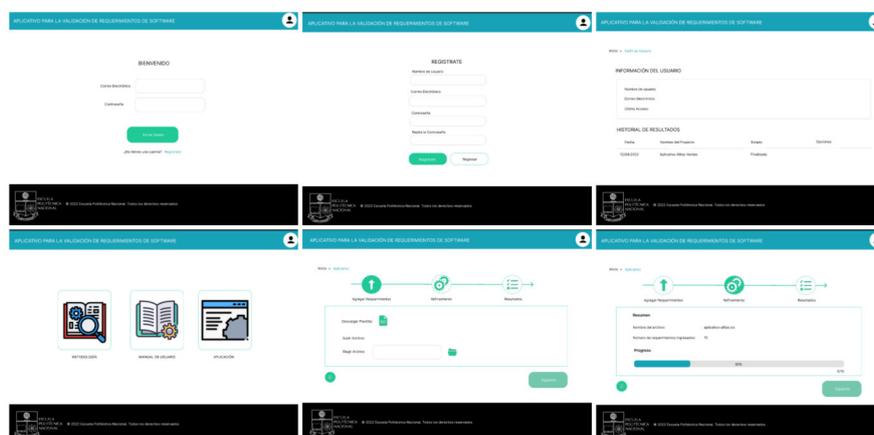


Figura 4.1: Evidencias del primer prototipo de la aplicación

❖ Segundo Prototipo

Para el segundo prototipo se añadieron nuevas interfaces y el proceso de automatización se rediseñó para una forma manual y otra automática. El número de interfaces aumentó significativamente.

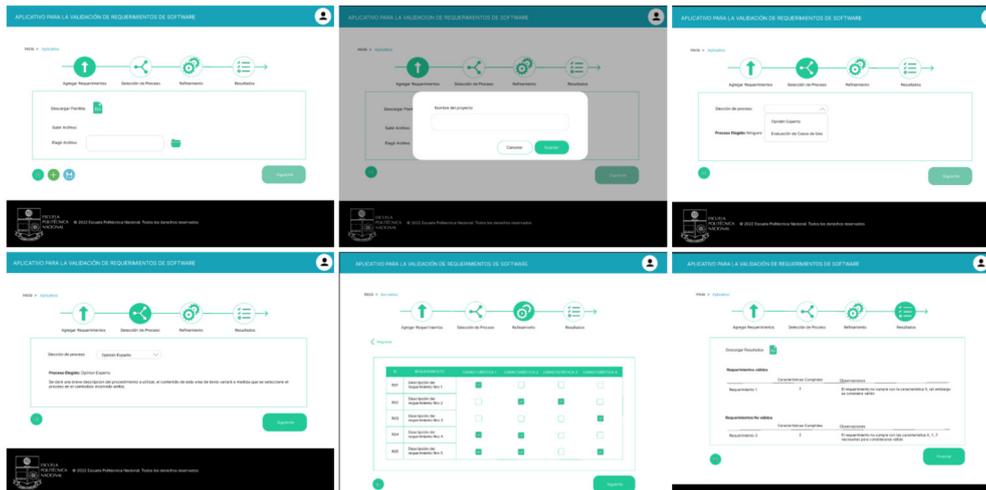


Figura 4.2: Evidencias del segundo prototipo de la aplicación

❖ Tercer Prototipo

El tercer prototipo fue el inicio de lo que sería la interfaz final del proyecto. Se optó por el proceso de refinamiento manual para proyectos de software comunes y se incorporó la automatización de la fase 5 de la metodología iPlus [2] para requerimientos de proyectos de juegos serios.

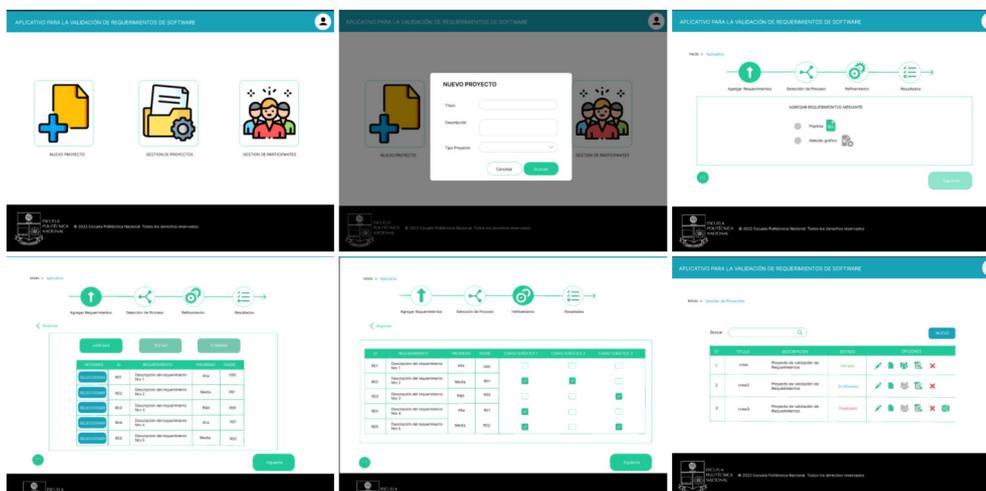


Figura 4.3: Evidencias del tercer prototipo de la aplicación - parte 1

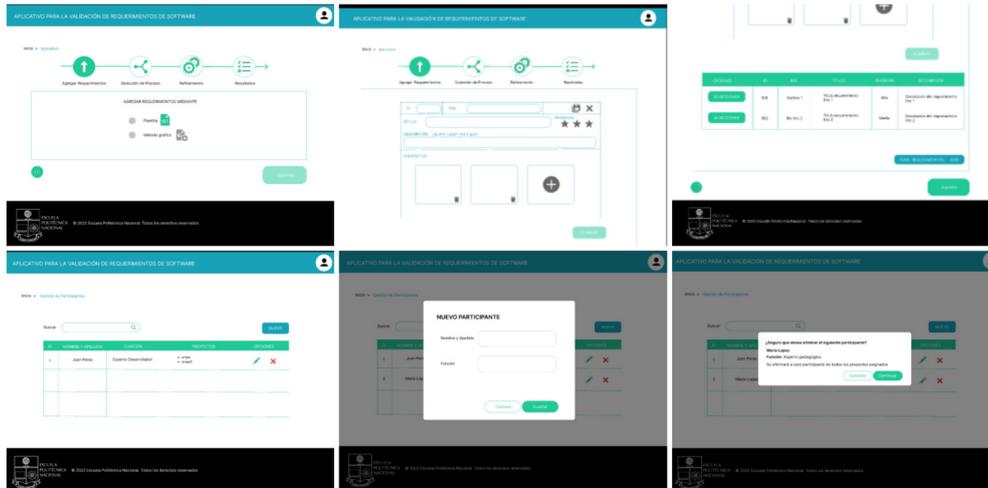


Figura 4.4: Evidencias del tercer prototipo de la aplicación - parte 2

❖ Cuarto Prototipo

En el cuarto y último prototipo se realizaron las correcciones a su predecesor previo a empezar el desarrollo de la aplicación. Se tomaron en consideración las sugerencias del dueño del producto, así como los criterios de usabilidad establecidos en las heurísticas de Nielsen.

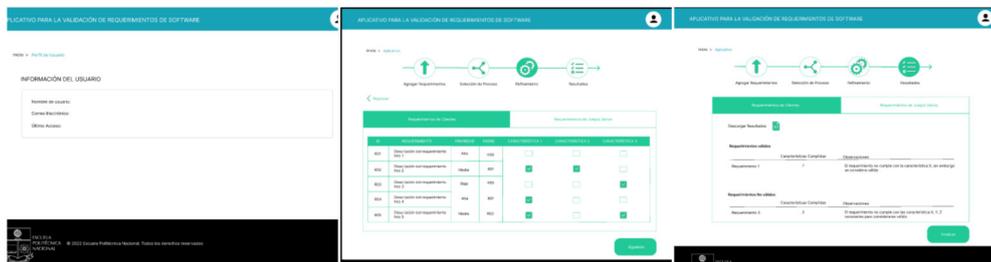


Figura 4.5: Evidencias del cuarto prototipo de la aplicación

4.2 EVALUACIÓN DE USABILIDAD CON USUARIO FINAL

Para la evaluación de usabilidad, se utilizó el protocolo propuesto por Abhay Rautela en [62]. El proceso de la evaluación se muestra en la figura 4.6, para cumplir con el protocolo se dividió en cuatro fases:

- ❖ La primera fase consiste en el paso 1, 2 y 3 del protocolo, es decir, que se realizará la definición de los objetivos de la evaluación, la preparación y la selección de los participantes.

- ❖ La segunda fase consiste en el paso 4 del protocolo, en donde se realizará la ejecución de la evaluación por parte de los participantes.
- ❖ La tercera fase consiste en analizar y presentar los resultados obtenidos en la evaluación, es decir, el paso 5 y 6 del protocolo.
- ❖ La cuarta fase comprende la discusión de los resultados obtenidos en la evaluación, que es el paso 7 del protocolo.

Además del protocolo, se utilizará la encuesta propuesta por Lewis en [63], conocido como Computer System Usability Questionnaire (CSUQ). Dicha encuesta consta de 16 preguntas y cada una de ellas es valorada en una escala de 1 al 7, donde 1 significa estar totalmente en desacuerdo y 7 estar totalmente de acuerdo.

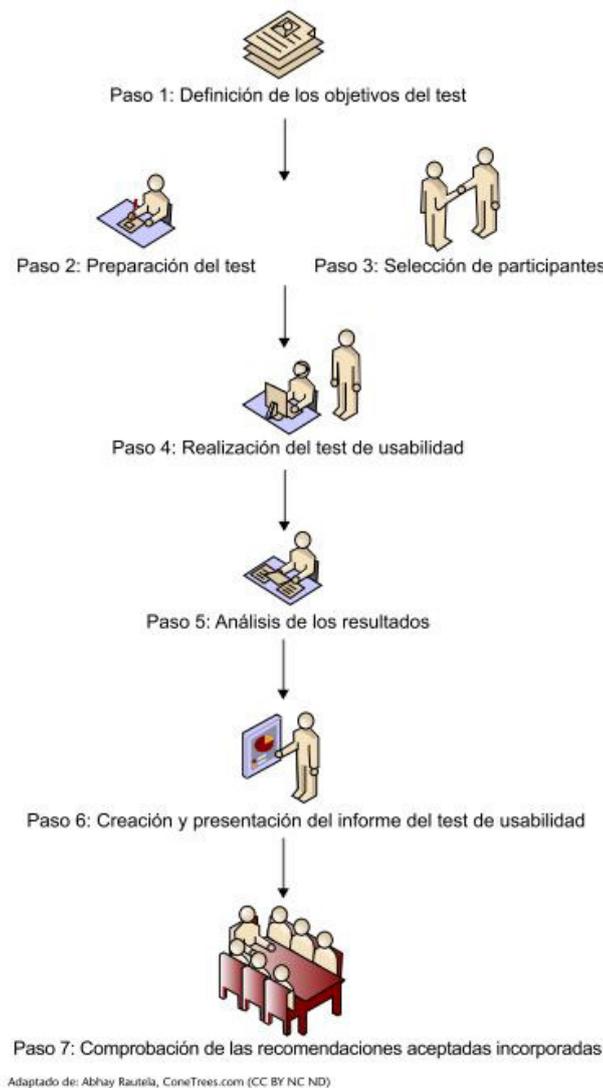


Figura 4.6: Protocolo de evaluación de usabilidad.

4.2.1 Primera fase

Definición de los objetivos de la evaluación.

- ❖ Utilizar la encuesta propuesta, como instrumento de evaluación de usabilidad del aplicativo para la validación de requerimientos de software.
- ❖ Conocer las opiniones de los usuarios y su grado de satisfacción respecto al aplicativo.

Preparación de la evaluación.

En este paso, se preparó las actividades y documentos correspondientes que serán entregadas a cada usuario. Estas actividades y documentos se describen a continuación:

- ❖ Consentimiento para efectuar el estudio y uso de información: Esta es una actividad donde se debe exponer a los usuarios el objetivo del estudio. Por ende, se preparó un documento de consentimiento que debe ser llenado y firmado por los participantes. Anexo 7.
- ❖ Presentación del funcionamiento de la aplicación: En esta actividad se da una exposición de las interfaces y las funciones más importantes de la aplicación.
- ❖ Experimento: En esta actividad se entrega el documento de las tareas y subtareas que van a ser evaluadas dentro de la aplicación. Los participantes son los encargados de ejecutar cada una de las tareas propuestas.

Selección de participantes

Para esta evaluación se convocó a 6 participantes, entre ingenieros y tesisistas pertenecientes a la Escuela Politécnica Nacional y otras universidades, quienes han trabajado previamente en desarrollo de software. De acuerdo con Nielsen [64] para realizar la evaluación de un producto de software se debe contar únicamente con 5 personas, con ello se puede identificar alrededor del 85 % de todos los problemas de usabilidad que pueda presentar el producto.

El aplicativo está enfocado a un único rol, por ende, no requiere el uso de varios grupos de participantes para evaluar las mismas tareas. Además, por ser un aplicativo web, la evaluación se realizó de forma remota.

4.2.2 Segunda fase

Realización del test de usabilidad

❖ Reunión planificada

La evaluación comenzó con una reunión introductoria entre los participantes como se muestra de la figura 4.7 a la figura 4.8, donde se trataron los siguientes puntos:

- ❖ Indicaciones de cómo se llevará a cabo la evaluación.
- ❖ Envío de la documentación correspondiente.
- ❖ Establecer el canal de comunicación para resolver dudas o preguntas que surjan durante la evaluación.

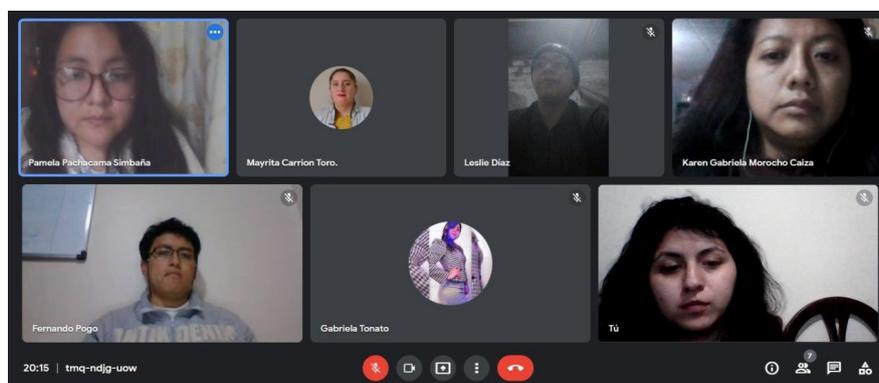


Figura 4.7: Reunión de planificación-indicaciones generales.

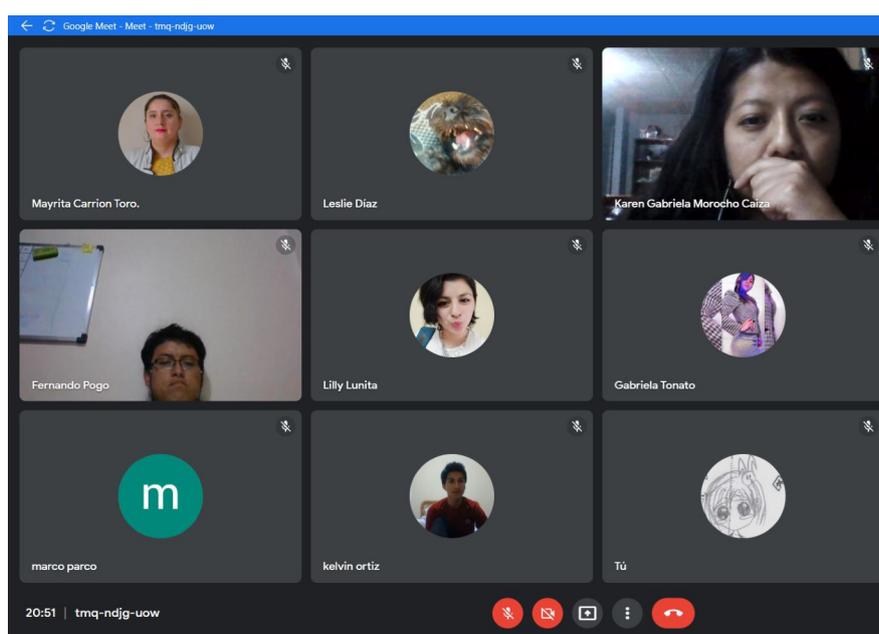


Figura 4.8: Reunión de planificación-envío de documentación.

❖ **Documento de consentimiento.**

Cada participante aceptó el documento de consentimiento, para poder comenzar con la evaluación de usabilidad. Ver Anexo 7.

❖ **Ingreso al aplicativo.**

Los participantes ingresaron de manera individual al aplicativo para la validación de requerimientos de software. En el siguiente enlace: <http://tesis-env.eba-fvibnpm3.us-east-1.elasticbeanstalk.com/login>

❖ **Tareas para realizar.**

Se estableció una lista de tareas, en base al caso de estudio como se muestra en la tabla 4.1

Tareas	Subtareas
T1. Ingresar a la aplicación	1.1. Registrarse en la aplicación, llenando los datos solicitados.
T2. Crear un nuevo proyecto	2.1. Seleccionar crear nuevo proyecto, y llenar los datos solicitados. Escoger tipo de proyecto: requerimientos de cliente.
	2.2. Seleccionar crear nuevo proyecto, y llenar los datos solicitados. Escoger tipo de proyecto: requerimientos de juego serio.
Paso 1: Agregar requerimientos	
T3. Seleccionar opción modo plantilla	3.1.1. Seleccionar método plantilla.
	3.1.2. Descargar formato de la plantilla.
	3.1.3. Buscar archivo plantilla01_historias_usuario.xlsx
	3.2.1. Seleccionar método plantilla.
	3.2.2. Descargar formato de la plantilla.
	3.2.3. Buscar archivo plantilla01_historias_usuario.xlsx
	3.2.4. Descargar plantilla GamePlay
	3.2.5. Buscar archivo plantilla01_gameplay.xlsx

T4. Seleccionar la opción modo gráfico	4.1.1. Llenar datos solicitados (5 veces) en pestaña Cliente.
	4.1.2. Seleccionar primer requerimiento editar descripción y título, después guardar los cambios.
	4.1.3. Seleccionar el segundo requerimiento y elimine.
T4. Seleccionar la opción modo gráfico	4.2.1. Llenar datos solicitados (5 veces) en pestaña Cliente.
	4.2.2. Seleccionar primer requerimiento editar descripción y título, después guardar los cambios.
	4.2.3. Seleccionar el segundo requerimiento y elimine.
	4.2.4. Seleccionar el tercer requerimiento y editar las ideas relacionadas, después guarde los cambios.
T4. Seleccionar la opción modo gráfico	4.3.1. Llenar datos solicitados (5 veces) en pestaña GamePlay.
	4.3.2. Seleccionar primer requerimiento editar descripción, después guardar los cambios.
	4.3.3. Seleccionar el segundo requerimiento y elimine.
	4.3.4. Seleccionar el tercer requerimiento y editar seleccione otros bloques GamePlays, después guarde los cambios.
Paso 2: Visualización de requerimientos	
T5. Gestionar requerimientos	5.1. Agregar nuevo requerimiento, llenar los datos solicitados.
	5.2. Editar descripción de requerimiento ingresado.
	5.3. Eliminar requerimiento.
Paso 3: Refinamiento	
T6. Llenar la matriz seleccionando propiedades.	6.1.1. Archivo seleccionado anterior lleno con matriz de propiedades.
	6.1.2. Cambiar selección de algunas propiedades.
	6.2.1. Archivo seleccionado anterior con la matriz de propiedades vacía.
	6.2.2. Seleccionar propiedades que crea necesario.
	6.3.1. Requerimientos ingresados de forma gráfica.
	6.3.2. Seleccionar propiedades que crea necesario.

Paso 4: Resultado	
T7. Desplegar los resultados	7.1. Desplegar la tabla de resultantes del proceso.
T8. Exportar Resultados	8.1. Descargar la tabla resultante en un archivo Excel.

Tabla 4.1: Tareas y subtareas

❖ **Evaluación con la encuesta propuesta.**

Una vez finalizadas las tareas de evaluación anteriormente detalladas, los participantes proceden a contestar la encuesta de usabilidad. La encuesta utilizada se muestra en la tabla 4.2.

Nº	Pregunta	Escala						
		1	2	3	4	5	6	7
1	En general, estoy satisfecho con lo fácil que es usar este sistema							
2	Es simple usar este sistema.							
3	Soy capaz de completar mi trabajo rápidamente usando este sistema.							
4	Me siento cómodo usando este sistema.							
5	Fue fácil aprender a usar este sistema.							
6	Creo que me volví productivo rápidamente usando este sistema.							
7	El sistema mostró mensajes de error que me indicaron claramente cómo solucionar problemas.							
8	Cada vez que cometía un error al utilizar el sistema, podía recuperarme fácil y rápidamente.							
9	La información (como ayuda en línea, mensajes en pantalla y otra documentación) proporcionada con este sistema es clara.							
10	Es fácil encontrar la información que necesitaba							
11	La información proporcionada con el sistema es eficaz para ayudarme a completar mi trabajo.							
12	La organización de la información en las pantallas del sistema es clara.							

13	La interfaz de este sistema es muy agradable (incluye ratón, teclado, mouse y pantallas, además los gráficos y lenguaje que se usan)								
14	Me gustó usar la interfaz de este sistema.								
15	Este sistema tiene todas las funciones y capacidades que espero tener.								
16	En general, estoy satisfecho con este sistema								

Tabla 4.2: Cuestionario para evaluar la usabilidad

4.3 RESULTADOS

En esta sección se describe la tercera fase del protocolo propuesto por Abhay Rautela [62], donde se presenta los resultados obtenidos de la evaluación de usabilidad en la sección 4.2.2.

4.3.1 Tercera fase

Esta fase consiste en presentar los resultados obtenidos en la evaluación de usabilidad. La encuesta fue realizada por los participantes convocados. La información obtenida sin procesar se encuentra en el Anexo 8.

Resultados

La figura 4.9 muestra los porcentajes promedio obtenidos en cada una de las preguntas de la encuesta CSUQ.

Como se puede observar en la figura 4.9 la pregunta 4 alcanza el porcentaje más alto con noventa y siete coma dos por ciento (97,2%). Por otro lado, el porcentaje más bajo es de ochenta y cinco coma ochenta y tres por ciento (85,83%) que corresponden a las preguntas: 2, 5, 8 y 10. Además, la media obtenida del porcentaje global de todas las preguntas planteadas es de noventa y uno coma cinco por ciento (91,5%).

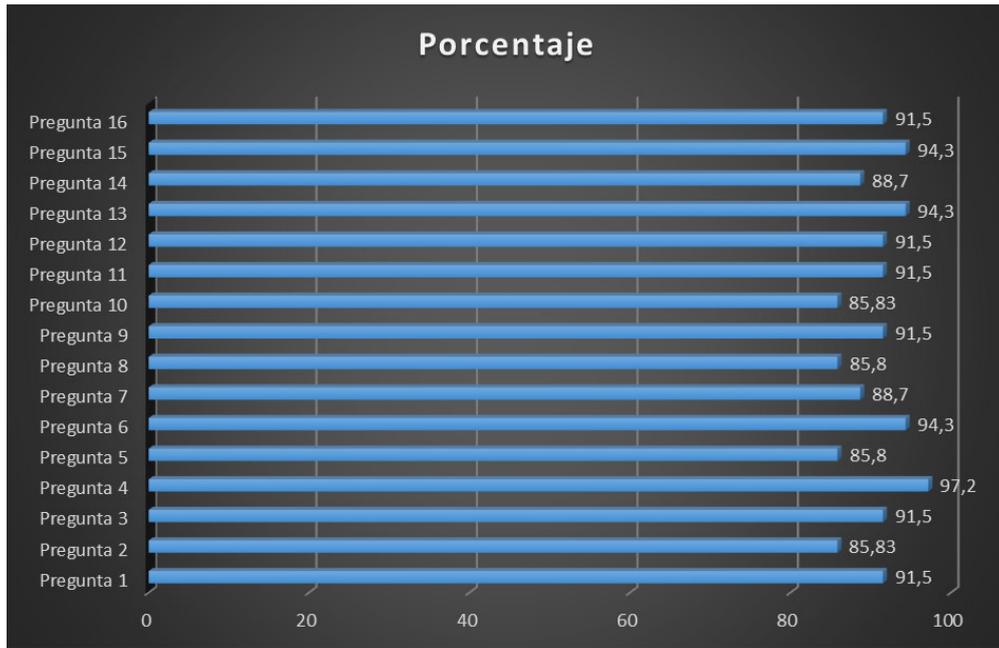


Figura 4.9: Promedio porcentual por pregunta, encuesta CSUQ

El porcentaje de cada pregunta por escala se muestra en la figura 4.10. La figura identifica los valores obtenidos en la encuesta CSUQ mediante la escala de 1 al 7, y representando predominancia mediante la longitud del color. Por lo tanto, la pregunta 4 muestra que más del ochenta y tres coma tres por ciento (83,3%) de los participantes están totalmente de acuerdo usando la aplicación. Por otro lado, en la pregunta 2 y 10 solo el dieciséis coma seis por ciento (16,6%) están totalmente de acuerdo.

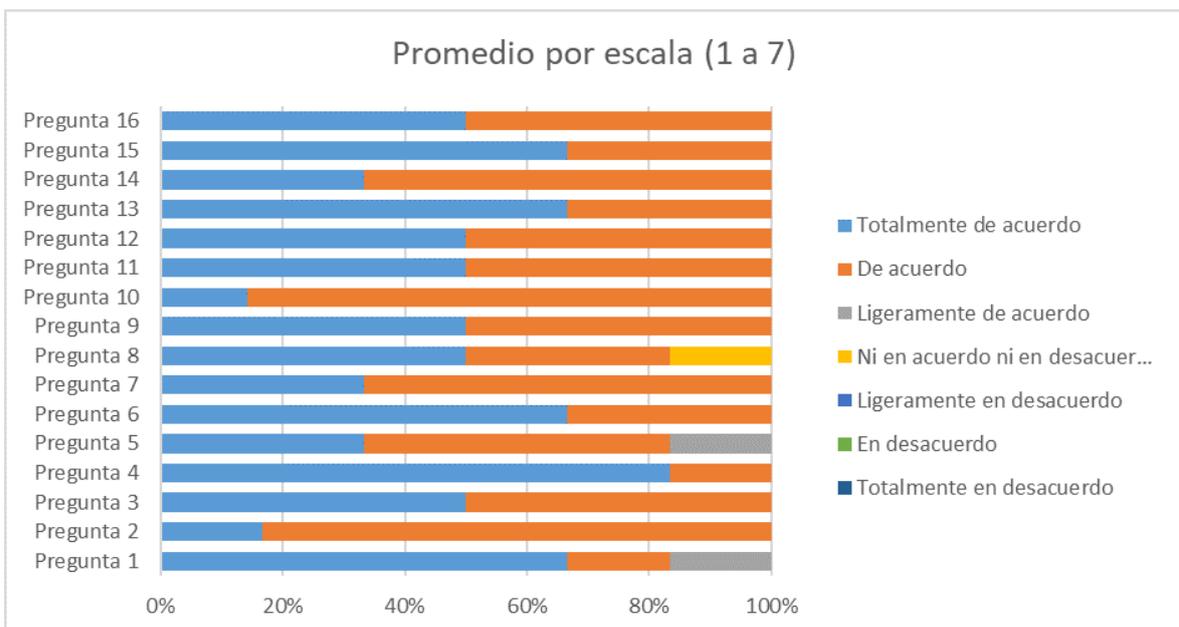


Figura 4.10: Promedio por escala

El Anexo 9, es la evidencia visual y gráfica de usabilidad. Se encuentra el Excel completo con la información procesada junto con las gráficas que se mostraron anteriormente.

4.4 DISCUSIÓN

En esta sección se describe la cuarta fase del protocolo propuesto por Abhay Rautela [62], donde se realiza la discusión de los resultados obtenidos en la sección 4.3.

4.4.1 Cuarta fase

Finalmente, en esta fase se procede a la discusión de los resultados obtenidos en la evaluación de usabilidad.

Con los resultados obtenidos en la encuesta CSUQ, se concluye que la aplicación es usable, ya que las preguntas: “En general, estoy satisfecho con lo fácil que es usar este sistema” y “En general, estoy satisfecho con este sistema”, mostradas en las figuras 4.11 y 4.12, tuvieron una aceptación mayor al ochenta por ciento (80 %). Es decir, que los participantes estuvieron de acuerdo con dichas preguntas, obteniendo mayor porcentaje en la escala de totalmente de acuerdo y en de acuerdo.

1. Overall, I am satisfied with how easy it is to use this system.

En general, estoy satisfecho con lo fácil que es usar este sistema.

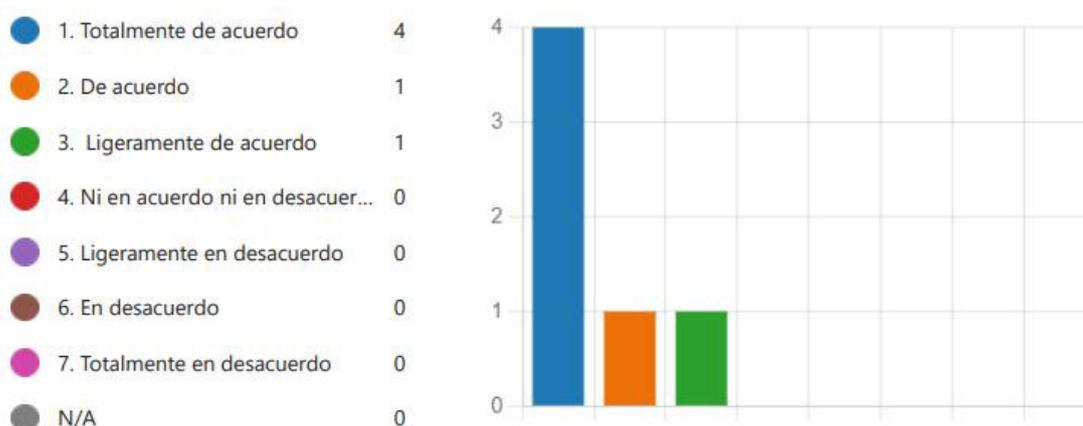


Figura 4.11: Resultados pregunta 1 - CSUQ.

16. Overall, I am satisfied with this system.

En general, estoy satisfecho con este sistema.

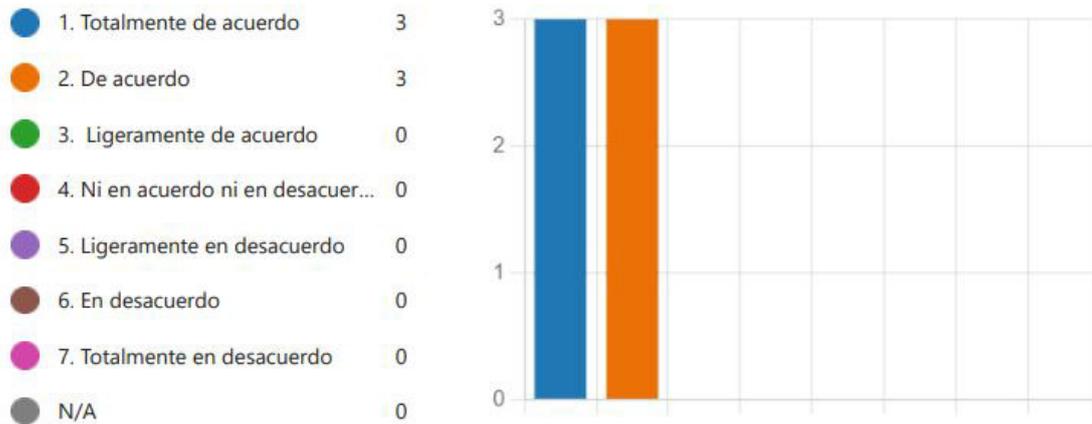


Figura 4.12: Resultados pregunta 16 - CSUQ.

De igual forma en la pregunta 5: "Fue fácil aprender a usar este sistema", se obtuvo un promedio de ochenta y cinco coma ocho por ciento (85,8 %) como se muestra en la figura 4.13, aunque fue el porcentaje más bajo supera el ochenta por ciento (80 %), que evidencia que para los participantes fue fácil aprender a usar la herramienta.

5. It was easy to learn to use this system.

Fue fácil aprender a usar este sistema.

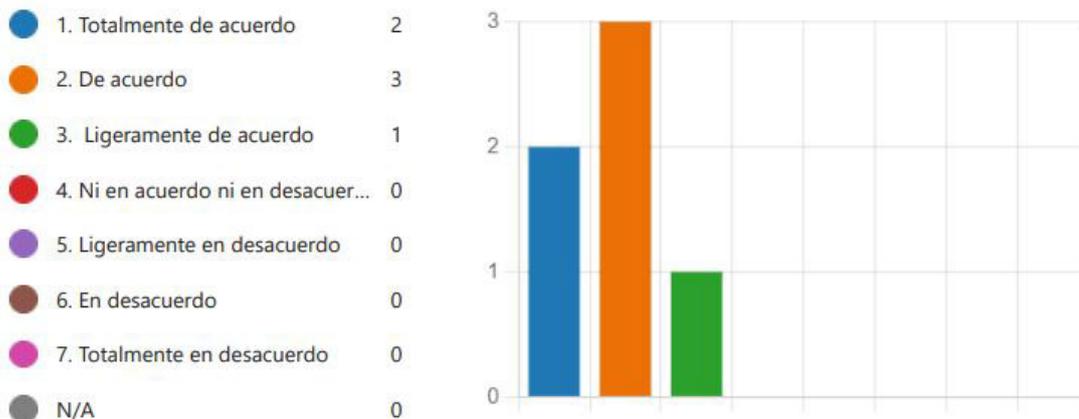


Figura 4.13: Resultados pregunta 5 - CSUQ.

En la pregunta 7: "El sistema mostró mensajes de error que me indicaron claramente cómo

solucionar problemas", se obtuvo ochenta y ocho coma siete por ciento (88,7%), como se muestra en la figura 4.14.

7. The system gives error messages that clearly tell me how to fix problems.

El sistema mostró mensajes de error que me indicaron claramente cómo solucionar problemas.

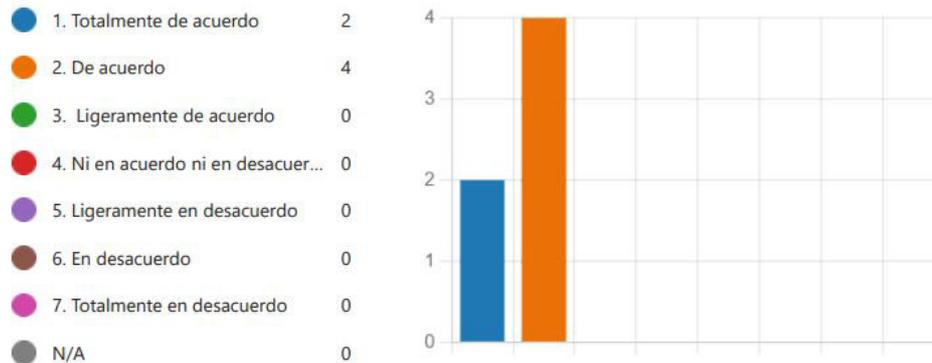


Figura 4.14: Resultados pregunta 7 - CSUQ.

4.5 RESUMEN DEL CAPÍTULO

En el capítulo 3 Evaluación y Resultados se presentan la evolución de las interfaces de la aplicación teniendo presente, para el diseño de estas, las heurísticas de Nielsen [61] y las sugerencias del cliente final. Los prototipos se fueron alterando a medida que surgían propuestas para lograr el objetivo general del proyecto.

Además, el capítulo presenta las pruebas de usabilidad, donde se utilizó el cuestionario CSUQ [63] y el protocolo propuesto por Abhay Rautela [62] que se lo dividió en cuatro fases: La primera fase consiste en definir los objetivos de la evaluación, la preparación y la selección de los participantes, en donde se convocó a 6 usuarios entre ingenieros y tesisistas pertenecientes a la Escuela Politécnica Nacional y otras universidades. La segunda fase consiste en realizar la ejecución de la evaluación por parte de los participantes, donde se entregó a los mismos los documentos necesarios para llevar a cabo la evaluación y recolección de datos al finalizar las tareas propuestas en esta fase. La tercera fase consiste en analizar y presentar los resultados obtenidos en la evaluación, se presenta el porcentaje por pregunta y porcentaje con respecto a la escala de la encuesta. Finalmente, la cuarta fase comprende la discusión de los resultados obtenidos en la evaluación realizada.

5 CONCLUSIONES, PROSPECTIVAS Y RECOMENDACIONES

5.1 CONCLUSIONES

- ❖ En este proyecto se realizó un estudio conceptual con la metodología propuesta por Morales. Este estudio considero 11 normas y estándares ISO/IEC/IEEE que ayudaron a encontrar las propiedades que debe cumplir un requerimiento para considerarse bien formado. Además, se utilizó el método para la determinación de criterios de relevancia donde se pudo corroborar el estudio realizado.
- ❖ En base al análisis de resultados del estudio realizado, se obtuvo la siguiente clasificación: Propiedades indispensables (apropiado, completo, factible, verificable y correcto), propiedades deseables (sin ambigüedad, singular, trazabilidad, conforme, consistente y modificable), y propiedad de implementación (necesario). Esta clasificación se implementó en el instrumento propuesto.
- ❖ En el presente trabajo se desarrolló de manera exitosa una aplicación web que automatiza la quinta fase de la metodología iPlus que es la de refinamiento, mediante el uso de un enfoque ágil como Scrum. Con el uso de los principios de desarrollo ágil se establecieron 10 historias de usuario (sección 2.3.3) distribuidas en 4 Sprints (sección 2.4).
- ❖ Para el proyecto se efectuó la evaluación de usabilidad mediante la encuesta CSUQ (sección 3.2.2), donde se llegó a la conclusión de que el aplicativo tiene un alto grado de usabilidad obteniendo 91,5% en el porcentaje global de todas las preguntas planteadas.
- ❖ Finalmente, la aplicación se implementó de manera exitosa en un ambiente de producción utilizando los servicios de AWS. La aplicación en ambiente de producción se

puede encontrar en la URL <http://tesis-env.eba-fvibnpm3.us-east-1.elasticbeanstalk.com/inicio>

5.2 PERSPECTIVAS

- ❖ En un proyecto futuro se debería proponer una metodología que utilice Inteligencia Artificial, para evitar que se llene la matriz booleana de manera manual y realizar un estudio directo de las propiedades, como entrada la descripción y al momento de ponderar realizar un estudio con expertos en refinamiento de requerimientos.
- ❖ Mejorar el diseño visual como son: perfil de usuario, historial de proyectos terminados, participantes, y más, mediante la ayuda de expertos en diseño gráfico y artístico.
- ❖ El aplicativo web para la validación de requerimientos es muy útil en el ámbito de desarrollo software, por ende, se podría considerar la adición de soporte para múltiples idiomas.

5.3 RECOMENDACIONES

- ❖ Realizar una correcta recopilación y categorización de todos los documentos relacionados directa o indirectamente con el tema de investigación, si fuese el caso, realizar sus respectivas transcripciones y traducciones para un mejor manejo de la información.
- ❖ Definir una propuesta de solución al objetivo general del proyecto antes de empezar el diseño de interfaces. Los constantes cambios en las soluciones del proyecto provocaron cambios significativos en las interfaces principales y aumento de tiempo de trabajo para realizar modificaciones en prototipos que vayan en línea con la solución propuesta. La retro alimentación que el dueño del producto pueda ofrecer es importante para lograr su satisfacción y evitar correcciones futuras cuando la aplicación esté en proceso de desarrollo.
- ❖ La selección de propiedades corresponde a uno de los pasos críticos dentro del refinamiento, realizar la selección adecuada de las propiedades influyen directamente en los resultados, dando la posibilidad de obtener un resultado no deseado.

6 REFERENCIAS BIBLIOGRÁFICAS

- [1] A. Hussain, E. O. Mkpojiogu y F. M. Kamal, «The role of requirements in the success or failure of software projects,» *International Review of Management and Marketing*, vol. 6, n.º 7Special Issue, págs. 305-310, 2016, ISSN: 21464405.
- [2] M. Carrión, M. Santorum, J. Benavides, J. Aguilar e Y. Ortiz, «Developing a virtual reality serious game to recreational therapy using iPlus Methodology,» *Conference: 2019 International Conference on Virtual Reality and Visualization (ICVRV)*, 2019. dirección: https://www.researchgate.net/publication/347154666_Developing_a_Virtual_Reality_Serious_Game_to_Recreational_Therapy_Using_iPlus_Methodology.
- [3] A. Mera, «Guía para interactuar con stakeholders en el proceso de ingeniería de requerimientos,» inf. téc., 2010. dirección: <https://repository.javeriana.edu.co/bitstream/handle/10554/7513/tesis341.pdf?sequence=1&isAllowed=y>.
- [4] B. Davey y K. R. Parker, «Requirements Elicitation Problems: A Literature Analysis,» *Instituto de Ciencias Informantes*, vol. 12, págs. 071-082, 2015. dirección: <http://iisit.org/Vol12/IISITv12p071-082Davey1929.pdf>.
- [5] P. C. Casanova y P. Consultores, «Factores clave para el éxito o el fracaso en proyectos de implementación de sistemas ERP,» págs. 84-87, 2010.
- [6] H. J. E. Herrada y R. A. N. Macurí, «MODELO DE MEJORA DEL CICLO DE VIDA DEL DESARROLLO DE SOFTWARE CON REFERENCIA A LA ISO / IEC 29110 CASO : MYPE HOLINSYS,» Tesis doct., Universidad de Lima. Escuela Universitaria de Ingeniería. Carrera de Ingeniería de Sistemas, 2016. dirección: https://repositorio.ulima.edu.pe/bitstream/handle/20.500.12724/3139/Egusquiza_Herrada_Hugo.pdf?sequence=1&isAllowed=y.
- [7] L. J. V. Mejía, «Estudio empírico sobre el proceso y la productividad de la ingeniería de requisitos en las empresas antioqueñas de software,» Tesis doct., Universidad

- EAFIT, 2014. dirección: https://repository.eafit.edu.co/bitstream/handle/10784/2844/LuzJanetteV%c3%a9lezMej%c3%ada_2014.pdf?sequence=3&isAllowed=y.
- [8] O. C. Medina, M. Martín Marciszack y M. A. Groppo, «Trazabilidad y validación de requerimientos funcionales de sistemas informáticos mediante la transformación de modelos conceptuales,» *ReCIBE. Revista electrónica de Computación, Informática, Biomédica y Electrónica*, vol. 5, 2016.
- [9] M. Cristiá, «Introducción a la Ingeniería de Requerimientos Software testing from set-based specifications View project,» inf. téc., 2011. dirección: <https://www.researchgate.net/publication/319728760>.
- [10] M. Arias Chaves, «La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software,» *InterSedes: Revista de las Sedes Regionales*, vol. 6, n.º 10, págs. 1-13, 2005, ISSN: 2215-2458. dirección: <https://revistas.ucr.ac.cr/index.php/intersedes/article/view/790/851>.
- [11] J. C. Moreno y M. M. Marciszack, «La Usabilidad Desde La Perspectiva De La Validación de Requerimientos No Funcionales Para Aplicaciones Web,» inf. téc., 2013. dirección: <http://conaiisi.frc.utn.edu.ar/PDFsParaPublicar/1/schedConfs/7/231-655-1-DR.pdf>.
- [12] A. O. Morales, «Fundamentos de la Investigación Documental y la Monografía. En Manual para la elaboración y presentación de la monografía,» *Universidad de Los Andes*, pág. 20, 2003. dirección: http://www.saber.ula.ve/bitstream/123456789/16490/1/fundamentos_investigacion.pdf.
- [13] M. d. C. Carrión Toro, *Cuestionario de Refinamiento de Requerimientos*, 2022.
- [14] IEEE, «IEEE 1233 Guide for Developing System Requirements Specification,» *IEEE Std*, 1998. dirección: <https://ieeexplore.ieee.org/document/741940>.
- [15] ISO/IEC, «ISO/IEC/ IEEE 29148 - Systems and software engineering - Life cycle processes - Requirements engineering,» 2018.
- [16] —, «ISO/IEC 25000:2014 - Systems and software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE,» *BSI Standards Publication*, 2014.
- [17] —, «ISO/IEC 25030:2012 Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — Quality requirements framework,» 2012. dirección: <https://kikakurui.com/x25/X25030-2012-01.html>.

- [18] —, «ISO/IEC 24766:2009 Information technology - Guideline to requirement engineering tool capabilities,» 2009.
- [19] R. M. Agut, «Especificación de Requisitos Software según el estándar de IEEE 830,» *Especificación de Requisitos Software según el estándar de IEEE 830*, pág. 19, 2000. dirección: <http://textos.pucp.edu.pe/pdf/3134.pdf>.
- [20] ISO/IEC, «INGENIERÍA DE SOFTWARE—CALIDAD DEL PRODUCTO—PARTE 1: MODELO DE LA CALIDAD NC-ISO/IEC 9126-1,» *Cuban National Bureau of Standards*, 2005.
- [21] —, «ISO/IEC 9000:2015 Sistemas de gestión de la calidad. Fundamentos y vocabulario,» 2015.
- [22] —, *International Standard ISO/IEC 12207: Systems and software engineering — Software life cycle processes*. 2008, vol. 2, pág. 138. dirección: [http://doi.wiley.com/10.1002/\(SICI\)1099-1670\(199603\)2:1%5C%3C35::AID-SPIP29%5C%3E3.0.CO;2-3](http://doi.wiley.com/10.1002/(SICI)1099-1670(199603)2:1%5C%3C35::AID-SPIP29%5C%3E3.0.CO;2-3).
- [23] —, «ISO/IEC 14598-1 Tecnología de la información. Evaluación del producto software. Parte 1: Visión General,» 2004.
- [24] —, «ISO/IEC/IEEE 15288:2015 Systems and software engineering - System life cycle processes,» vol. 17, 2015.
- [25] —, «ISO/IEC 15504-2:2003 Information technology – Process assessment – Part 2: Performing an assessment,» 2003.
- [26] «IEEE Standard Glossary of Software Engineering Terminology,» *IEEE Std 610.12-1990*, págs. 1-84, 1990. DOI: 10.1109/IEEESTD.1990.101064.
- [27] I. C. Society, *Guide to the Software Engineering Body of Knowledge Version 3.0 (SWE-BOK Guide V3.0)*.
- [28] R. M. Nieto, «Desarrollo de una Aplicación Web para la Determinación de Criterios de Relevancia Aplicados a Revisiones Documentales de Ingeniería,» *Escuela Politécnica Nacional*, 2021. dirección: <https://bibdigital.epn.edu.ec/bitstream/15000/21718/1/CD%2011202.pdf>.
- [29] K. Wiegers y J. Beatty, *Software Requirements*. 2013, pág. 208. dirección: <https://manualzz.com/doc/46725131/software-requirements-3--3rd-edition>.
- [30] M. Cohn, *User Stories Applied: For agile software development*. 2009. dirección: <https://athena.ecs.csus.edu/~buckley/CSc191/User-Stories-Applied-Mike-Cohn.pdf>.

- [31] Scrum.org, *What is Scrum?* Dirección: <http://www.scrum.org/resources/what-is-scrum> (visitado 21-03-2022).
- [32] Scrumguides.org, *Scrum Guide*. dirección: <http://www.scrumguides.org/scrum-guide.html> (visitado 21-03-2022).
- [33] Scrum.org, *What is a Product Backlog?* Dirección: <https://www.scrum.org/resources/what-is-a-product-backlog> (visitado 24-03-2022).
- [34] —, *What is an Increment in Scrum?* Dirección: <https://www.scrum.org/resources/what-is-an-increment> (visitado 24-03-2022).
- [35] —, *What is a Sprint in Scrum?* Dirección: <https://www.scrum.org/resources/what-is-a-sprint-in-scrum> (visitado 24-03-2022).
- [36] —, *What is Sprint Planning?* Dirección: <https://www.scrum.org/resources/what-is-sprint-planning> (visitado 24-03-2022).
- [37] —, *What is a Daily Scrum?* Dirección: <https://www.scrum.org/resources/what-is-a-daily-scrum> (visitado 24-03-2022).
- [38] —, *What is a Sprint Review?* Dirección: <https://www.scrum.org/resources/what-is-a-sprint-review> (visitado 24-03-2022).
- [39] —, *What is a Sprint Retrospective?* Dirección: <https://www.scrum.org/resources/what-is-a-sprint-retrospective> (visitado 24-03-2022).
- [40] N. Tecnológica. «ESTRUCTURA DE UNA BASE DE DATOS: NORMALIZACIÓN Y MÁS.» (), dirección: <https://notatecnologica.com/tecnologia/estructura-de-una-base-de-datos/>.
- [41] I. Amazon Web Services. «Preguntas frecuentes sobre Amazon EC2.» (2001), dirección: <https://aws.amazon.com/es/ec2/faqs/>.
- [42] —, «AWS Elastic Beanstalk.» (), dirección: <https://aws.amazon.com/es/elasticbeanstalk/>.
- [43] —, «AWS Elastic Beanstalk. Guía para desarrolladores: Paso 1.» (), dirección: https://docs.aws.amazon.com/es_es/elasticbeanstalk/latest/dg/GettingStarted.CreateApp.html.
- [44] —, «Amazon RDS.» (), dirección: <https://aws.amazon.com/es/rds/>.
- [45] E. Bascón Pantoja, «El patrón de diseño Modelo-Vista-Controlador (MVC) y su implementación en Java Swing,» *Acta Nova*, vol. 2, n.º 4, págs. 493-507, 2004.

- [46] R. J. Vargas, D. Valle y J. P. M. Granados, «Programación en Capas.»
- [47] Figma, *The modern interface design tool*. dirección: <https://www.figma.com/ui-design-tool/> (visitado 26-03-2022).
- [48] MDN we docs, *JavaScript*. dirección: <https://developer.mozilla.org/es/docs/Web/JavaScript> (visitado 26-03-2022).
- [49] Angular, *Introduction to the Angular Docs*. dirección: <https://angular.io/docs> (visitado 26-03-2022).
- [50] Nodejs, *Acerca de Node.js*. dirección: <https://nodejs.org/es/about/> (visitado 26-03-2022).
- [51] Nestjs, *Nest Documentation - Introduction*. dirección: <https://docs.nestjs.com/> (visitado 26-03-2022).
- [52] Mysql, *What is MySQL?* Dirección: <https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html> (visitado 26-03-2022).
- [53] JetBrains, *Getting started with WebStorm*. dirección: <https://www.jetbrains.com/help/webstorm/getting-started-with-webstorm.html#:~:text=WebStorm%20is%20an%20integrated%20development,%2C%20HTML%2C%20and%20style%20sheets.> (visitado 26-03-2022).
- [54] Docker, *Docker overview*. dirección: <https://docs.docker.com/get-started/overview/> (visitado 26-03-2022).
- [55] DBeaver, *DBeaver Documentation*. dirección: <https://dbeaver.com/docs/wiki/> (visitado 26-03-2022).
- [56] Git, *Getting Started - What is Git?* Dirección: <https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F> (visitado 26-03-2022).
- [57] GitHub, *Comenzar con GitHub Desktop*. dirección: <https://docs.github.com/es/desktop/installing-and-configuring-github-desktop/overview/getting-started-with-github-desktop> (visitado 26-03-2022).
- [58] GitHub Docs, *Hola Mundo - Introducción*. dirección: <https://docs.github.com/es/get-started/quickstart/hello-world> (visitado 26-03-2022).
- [59] GitKraken, *About GitKraken*. dirección: <https://www.gitkraken.com/about> (visitado 26-03-2022).

- [60] C. Betancourt, «Desarrollo de una Aplicación Educativa Gamificada para Preservar el Patrimonio y Material de las Comunidades y Pueblos Indígenas,» 2021. dirección: <https://bibdigital.epn.edu.ec/bitstream/15000/21655/1/CD%2011134.pdf>.
- [61] J. Nielsen, «Ten Usability Heuristics,» dirección: <http://ingenieriasimple.com/usabilidad/10Heuristics.pdf>.
- [62] A. Rautela, «The Usability Testing Process,» 2018. dirección: https://codecamp.com.br/artigos_cientificos/usability-testing-process-a4.pdf.
- [63] J. Lewis, «Measuring Perceived Usability: The CSUQ, SUS, and UMUX,» *International Journal of Human-Computer Interaction*, 2018. dirección: https://www.researchgate.net/publication/322348620_Measuring_Perceived_Usability_The_CSUQ_SUS_and_UMUX.
- [64] J. Nielsen, *Why You Only Need to Test with 5 Users*, 2000. DOI: <http://www.useit.com/alertbox/20000319.html>. dirección: <https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/%20http://www.useit.com/alertbox/20000319.html> (visitado 01-07-2022).

7 ANEXOS

- ❖ Anexo 1: Matriz de normas y estándares.
- ❖ Anexo 2: Resultado de KEY TERMS AND FREQUENCY ANALYSIS PROCESS
- ❖ Anexo 3: Matriz de refinamiento.
- ❖ Anexo 4: Ejecución de Sprints.
- ❖ Anexo 5: Caso de estudio - Matriz historia de usuario
- ❖ Anexo 6: Caso de estudio - Matriz GamePlay
- ❖ Anexo 7: Documento de consentimiento.
- ❖ Anexo 8: Excel completo - respuestas de la encuesta de usabilidad.
- ❖ Anexo 9: Evidencia visual - graficas de la encuesta de usabilidad.
- ❖ Anexo 10: Respuestas encuesta de usabilidad.
- ❖ Anexo 11: Prototipo Final de la Aplicación.