Especialización en Gestión de la Innovación y la Vinculación Tecnológica

Centro de Estudios Interdisciplinarios

Universidad Nacional de Rosario

Implementación de una Metodología Centrada en el Usuario en el Proceso de Verificación y Validación de Software

Autora: *Ing. Valeria Raquel Filoniuk*

Directora: *Ing. Ma. Soledad Martínez*

Córdoba, 21 de noviembre de 2023

AGRADECIMIENTOS

Tengo varias personas a quien agradecer, pero, antes que nada, quiero dar un agradecimiento a mi hermano, que ya no está entre nosotros. Su pasión y tenacidad fueron el motor que me impulsaron a concretar esta propuesta.

A Ma. Soledad Martínez, mi tutora, porque desde el primer momento en que aceptó asesorar el Trabajo Final hasta el último día ha sido una persona incondicional. De ella he aprendido mucho y sigo aprendiendo.

A mi esposo e hijos, que son mis ganas de seguir, son los que estimulan mis logros personales y quienes me acompañan en cada uno de mis desafíos.

A mis amigos, la familia que yo elegí, quienes sin esperar nada a cambio están siempre buscando alentar mis ánimos.

RESUMEN

El presente Trabajo Final tiene como finalidad aplicar la Metodología Design Thinking en el Ciclo de Vida de Verificación y Validación de todos los proyectos asociados a Software Operativo de la Fuerza Aérea Argentina, para dar una solución creativa e innovadora y optimizar su funcionalidad. La utilización de este método en las primeras etapas de dicho proceso permite una interacción activa y temprana con el usuario, aumentando la generación de opciones de solución y la identificación de necesidades, consiguiendo contribuir a la calidad del producto final. Es importante asegurar la calidad en el desarrollo de este tipo de software, debido a su funcionamiento crucial y a la esencialidad que aportan en la defensa aérea de nuestro país.

Para mejorar el proceso mencionado se trabaja principalmente con tres escenarios de Design Thinking: Ideación, Inspiración e Implementación. El objetivo principal de este Proyecto, es integrar los escenarios mencionados en el Ciclo de Vida de Verificación y Validación de manera que ayuden a gestionar una nueva forma de testear software, incluyendo al usuario como fuente estratégica. Se espera que los resultados contribuyan a crear productos de calidad, ya que los mismos representan un activo esencial para quienes asumen el desafío de custodiar el cielo argentino.

Palabras clave: Calidad, Verificación y Validación de Software, Design Thinking, Usuario.

ABSTRACT

This article exposes the application of the “Design Thinking” Methodology during the Verification and Validation process in the Flight Simulator Software´s Life Cycle in order to ensure the Flight Simulator´s functionality. This will lead to provide more creative and innovative solutions during the Verification and Validation process. The application of this method in the first stages of the process allows an active and early interaction with user´s needs, increasing the building of solutions and the identification of needs, heighten the final product´s standards. It is important to ensure quality in the software development of flight´s simulators, due to its application is crucial and essential to the air defense of our country. So as to improve the aforementioned process, Design Thinking works with three main scenarios: Ideation, Inspiration and Implementation. The main objective of this article is to integrate these scenarios with the Verification and Validation of the Software´s Life Cycle and develop a new way to test software, including the user as a strategic source. It is expected that the results will contribute to create high quality products being these vital assets for the pilot´s training process who daily assume the challenge of guarding the Argentine sky.

Keywords: Design Thinking, Quality, User, Verification and Validation of Software.

CONTENIDO

INTRODUCCION 6

I DESCRIPCION GENERAL 7

1. ANTECEDENTES, PROBLEMÁTICA, JUSTIFICACION, SOLUCIÓN AL PROBLEMA 7

1.1 ANTECEDENTES 7

1.2 PLANTEO DEL PROBLEMA 8

1.3 JUSTIFICACIÓN 9

1.4 SOLUCIÓN AL PROBLEMA 10

1.5 DESCRIPCION DEL PROYECTO 10

II MARCO TEÓRICO 13

2. ASPECTO TEÓRICO 13

2.1 RELEVAMIENTO DE ANTECEDENTES 13

2.2 FUNDAMENTACIÓN TEÓRICA 14

III ANÁLISIS DE LA PROBLEMÁTICA Y SOLUCIÓN PROPUESTA 21

3. SITUACIÓN ACTUAL DE LOS PROYECTOS DE DESARROLLO DE SOFTWARE EN EL CICLO DE VIDA DE V&V 21

3.1 PROBLEMAS EN LA FASE DE ANÁLISIS DE REQUERIMIENTOS 21

3.2 PROBLEMAS EN LA FASE DE PLANIFICACIÓN DE PRUEBAS 22

3.3 PROBLEMAS EN LA FASE DE DISEÑO Y DESARROLLO DE PRUEBAS 23

3.4 PROBLEMAS EN LA FASE DE EJECUCIÓN DE PRUEBAS 25

3.5 PROBLEMAS EN LA FASE DE EVALUACIÓN DE RESULTADOS 26

3.6 PROBLEMAS EN LA FASE DE CIERRE DEL PROCESO 27

3.7 SOLUCIÓN A LA PROBLEMÁTICA ACTUAL 27

IV IMPLEMENTACIÓN DE LA SOLUCIÓN 30

4. DEFINICION DE SOFTWARE Y EJECUCIÓN DE MARCO DE TRABAJO 30

4.1 DEFINICIÓN DEL SOFTWARE 30

4.2 EJECUCIÓN DEL MARCO DE TRABAJO 31

V EVALUAR LOS RESULTADOS 59

5. EVALUACION DE RESULTADOS 59

5.1 RESULTADOS Y MEDIDAS 59

VI CONCLUSIONES 65

VII REFERENCIAS 66

INTRODUCCION

Este apartado muestra un resumen sobre los capítulos que conforman el Trabajo Final "***Implementación de una Metodología Centrada en el Usuario en el Proceso de Verificación y Validación de Software (V&V).***

El documento consta de seis secciones que describen el proceso del Trabajo Final. La primera sección presenta la descripción general del mismo, donde se contextualiza al lector sobre la problemática que se busca solucionar y la estrategia para llevarla a cabo, también encontrará la información sobre el objetivo general, los objetivos específicos y la metodología de trabajo empleada.

En la segunda sección, se indican los estudios anteriores encontrados con relación al proyecto, así como un análisis de toda la información técnica y los lineamientos teóricos para el desarrollo del mismo.

La tercera sección detalla la situación actual de las actividades que se realizan en el Ciclo de Vida de Verificación y Validación de Software y posteriormente plantea la solución a la problemática derivada de dichas actividades. El resultado es el diseño de un marco de trabajo que integra los conceptos de Design Thinking (DT) con el proceso de Verificación y Validación de Software.

La cuarta sección expone la forma mediante la cual el marco de trabajo propuesto evidencia la factibilidad de llevarlo adelante en un caso real, especificando las técnicas implementadas como estrategia de solución.

La quinta sección, demuestra cómo la aplicación de Design Thinking permite obtener mejores resultados en el Ciclo de Vida de V&V, además de medir el impacto que tiene el uso de la misma en tal proceso. La última sección, finaliza con las conclusiones y los beneficios obtenidos.

I DESCRIPCION GENERAL

El presente capítulo presenta la descripción general del Trabajo Final, donde se contextualiza al lector sobre la problemática que busca solucionar y la estrategia para llevarla a cabo, también encontrará la información sobre el objetivo general, los objetivos específicos y la metodología empleada para llevar adelante el mismo.

1. ANTECEDENTES, PROBLEMÁTICA, JUSTIFICACION, SOLUCIÓN AL PROBLEMA

1.1 ANTECEDENTES

La Verificación y Validación de Software, conocida comúnmente como “Testing” es un conjunto de procesos de comprobación y análisis que aseguran que el software que se desarrolla está acorde a su especificación y cumple con las necesidades de los usuarios.

Con la verificación se comprueba que el software cumple con los Requisitos Funcionales y No Funcionales de su especificación, y con la validación se comprueba que cumpla con las expectativas del usuario [1].

El Testing es considerado una de las actividades más importantes y fundamentales aplicable a todo proyecto que contenga desarrollo de Software, ya que posibilita procesos, métodos de trabajo, técnicas de pruebas y estrategias necesarias para garantizar la calidad del mismo [2].

Sin embargo, en los proyectos de desarrollo de software, es común que surjan situaciones que comprometan el éxito o la continuidad de los mismos debido a diferentes factores. Usualmente los problemas surgen debido a requerimientos incompletos, incapacidad para empatizar e incorporar al usuario final, como así también, diseñar soluciones que no responden a sus necesidades reales, lo cual repercute en la calidad del producto final.

Es responsabilidad de los actores involucrados en cada proyecto anticiparse a estas situaciones a fin de reducir la probabilidad de ocurrencia, y minimizar el impacto [3].

La Dirección de Análisis Operativo (DAO) dependiente de la Dirección General de Investigación y Desarrollo (DGID) de la Fuerza Aérea Argentina (FAA), tiene la

responsabilidad de realizar la Verificación y Validación de todos los Proyectos asociados a Software Operativo de la FAA. Por esta razón, los especialistas en testing que planifican y ejecutan pruebas, necesitan saber lo que el usuario requiere, debido al funcionamiento crucial y a la esencialidad que este tipo de software aporta en materia de defensa de una nación. Deben escuchar al usuario final y llevar a cabo actividades de acuerdo a los requerimientos que se solicitan.

Ante este escenario se observa que la metodología tradicional utilizada actualmente concibe proyectos con una estructura rígida, siguiendo un proceso secuencial en una sola dirección y sin posibilidad de dar marcha atrás; los requerimientos son acordados en una única instancia y luego son utilizados para todo el proyecto, contando con poca comunicación con el usuario. La planificación, diseño y ejecución de las pruebas de software son llevadas a cabo perdiendo el enfoque en el valor al usuario, siendo difícil definir lo que ellos reconocen como valioso.

Esta situación dificulta el proceso de Verificación y Validación de Software (V&V), provocando que no haya una interacción activa y temprana con los usuarios. Solo se tiene contacto con estos actores al finalizar las pruebas lo cual obstaculiza el feedback; disminuyendo la generación de opciones de solución, la identificación de necesidades del usuario y aumentando los niveles de riesgos ya que no se cuenta con su opinión en el momento apropiado. Como consecuencia, la calidad del software no será la que ellos esperan.

1.2 PLANTEO DEL PROBLEMA

Las dificultades asociadas a un proceso de V&V poco efectivo, ponen de manifiesto la necesidad de comenzar a plantear un cambio en las estrategias existentes.

Los proyectos de desarrollo de software se ven afectados debido a factores que impactan en el resultado de los mismos. Estos factores incluyen requerimientos incompletos, falta de involucramiento con usuarios y diseño de soluciones que no satisfacen las expectativas de los mismos.

La metodología usada actualmente no contempla estrategias para atacar estos factores y consecuentemente repercute en la calidad del producto obtenido.

Entonces, ¿Cómo lograr un proceso de Verificación y Validación de Software basado en una metodología centrada en el usuario para satisfacer sus necesidades/expectativas y asegurar la calidad del software?

1.3 JUSTIFICACIÓN

Este proyecto pretende interactuar con usuarios desde etapas tempranas del proceso de Verificación y Validación de Software, lo cual ayudará al equipo de testing a entender sus necesidades, priorizando sus problemas. Actualmente, debido a que los usuarios no tienen participación desde el comienzo, el área de testing se encuentra en determinadas ocasiones con que la solución planteada no es la deseada. En consecuencia, se llega al final del Ciclo de Vida de V&V trabajando en una propuesta que no cumple las expectativas del usuario y por lo tanto genera insatisfacción, lo cual obliga a retroceder a etapas anteriores. Tal situación conduce a los desarrolladores a cambiar soluciones y a los especialistas en testing a planificar y ejecutar pruebas nuevamente.

Muchas veces los desarrolladores tienen poco entendimiento de las necesidades reales de los usuarios, ya que no empatizan con ellos y no las consideran como factor de calidad, corriendo el riesgo de obtener software poco usable. Como resultado, se desarrollan productos que al ser testeados y probados para la aceptación del usuario final no responden a sus expectativas. Hecho que conduce a pérdida de tiempo y a un incremento de los costos del proyecto.

1.4 SOLUCIÓN AL PROBLEMA

Desarrollar un marco de trabajo basado en la integración de Design Thinking (DT) con el proceso de Verificación y Validación de Software, para atenuar los principales factores que interfieren en la calidad del software, utilizando técnicas y herramientas centradas en entender y solucionar las necesidades reales de los usuarios.

El marco de trabajo se implementará dentro de un proyecto real de software operativo de la FAA para validar su aplicación. Para ello, la DAO cuenta con capital humano, habilidades, capacidades, conocimientos técnicos sobre metodologías y gran interés para optimizar el proceso mencionado. Asimismo, el tiempo requerido para llevarlo a cabo es acorde al que se necesita y no demanda gran inversión monetaria.

1.5 DESCRIPCION DEL PROYECTO

A continuación, se presenta la descripción del proyecto estableciendo objetivo general, objetivos específicos y la metodología de trabajo:

1.5.1 OBJETIVO GENERAL

Implementar un marco de trabajo que sirva de base para comenzar a gestionar una nueva forma de verificar y validar software, incluyendo al usuario como fuente estratégica.

1.5.2 OBJETIVOS ESPECÍFICOS

- Analizar las etapas del proceso de Verificación y Validación de Software y las de la metodología Design Thinking, para luego unificarlas y ejecutar un marco de trabajo.
- Validar la aplicación del marco de trabajo en un proyecto real asociado a software operativo de la FAA.

- Dar a conocer el marco teórico de la metodología basada en pensamiento de diseño (Design Thinking), puesto que todos los actores involucrados en el proceso deben incorporarla para que sea efectiva.
- Garantizar la continuidad del marco de trabajo propuesto, para que permita gestionar futuros proyectos de Verificación y Validación de Software, optimizando los resultados.

1.5.3 METODOLOGÍA DE TRABAJO

La metodología de trabajo que se utiliza para el presente proyecto se basa en identificar y analizar la causa del problema que interfiere en el cumplimiento de los objetivos en el proceso de V&V, para luego generar, implementar y evaluar la solución. Consiste en observar cómo se desarrolla el proceso de Verificación y Validación de Software y los pasos propuestos por la metodología Design Thinking para luego integrarlos.

Con el objetivo de aclarar lo anteriormente expuesto, se detallan a continuación las fases de trabajo a ejecutar:

Primera Fase*:** ***Identificar y analizar el problema	Encontrar y comprender el problema que afecta el logro de los objetivos en el proceso de V&V, analizando las fallas que está provocando.
Segunda Fase: ***Generar la Solución***	Proponer la solución que permita resolver el problema. Esta fase se enfocará en el estudio de la metodología Design Thinking y el análisis de los pasos que se desarrollan en el Ciclo de Vida de V&V, con el objetivo de unificarlos. Al final de esta etapa se obtendrá el diseño de un marco de trabajo que combine ambos conceptos.
***Tercer Fase*:** ***Implementar la Solución*:**	Ejecutar la solución mediante la implementación de un caso real, con el propósito de determinar si la misma permite abordar los desafíos planteados.
Cuarta Fase: ***Evaluar Resultados***	Demostrar que la implementación de Design Thinking permite obtener mejores resultados en el proceso de V&V. Se utilizarán métricas, que nos permitirán observar el grado de satisfacción de los usuarios. Del mismo modo, se dará a conocer la metodología a los actores involucrados, de tal manera que sirva de base para gestionar futuros proyectos.

Figura 1. Metodología de Trabajo para el Proyecto

II MARCO TEÓRICO

En este capítulo se indican los estudios anteriores encontrados con relación al presente proyecto, así como un análisis de toda la información técnica y los lineamientos teóricos para el desarrollo del mismo.

2. ASPECTO TEÓRICO

2.1 RELEVAMIENTO DE ANTECEDENTES

El actual proyecto surge de un problema que se centra en las dificultades asociadas a un proceso de Verificación y Validación de Software poco efectivo, por no contar con una metodología centrada en identificar las necesidades prioritarias de los usuarios.

La investigación realizada en el artículo "**Casos de Estudio de Design Thinking en las etapas de Análisis y Diseño del Desarrollo de Software",** sugiere que DT puede tener un impacto positivo en el desarrollo de software, ya que facilita una profunda comprensión de las necesidades del usuario y aumenta la colaboración del equipo. [4]

El artículo señala que Design Thinking es lo suficientemente flexible para ser tomado como marco general de mejora de soluciones, adecuándose para ser utilizado en los procesos de desarrollo de software. [4]

Como se explicó anteriormente, en Ingeniería de Software, los proyectos están afectados debido a factores que impactan en el resultado de los mismos. Estos factores contribuyen a incrementar el costo, tiempo, y a disminuir la calidad del producto software [5]. Según lo enunciado en [6]**,** esta metodología permite dar soluciones a tales circunstancias ya que se centra en las personas, considerando multidisciplinariedad, colaboración y concreción de pensamientos y procesos, los cuales son caminos que llevan a soluciones innovadoras basándose en la resolución de problemas, desde el punto de vista del usuario y con el apoyo de prototipos [6]. Se apoya en la creatividad e innovación, integrando factores humanos y tecnológicos [5].

La utilización de Design Thinking en el del Ciclo de Vida de Verificación y Validación de Software, de acuerdo a lo expresado en el artículo "**Aplicación del Método Design Thinking en el Área de Requerimientos de Software"**, nos posibilita identificar técnicas aplicables a las diferentes etapas de tal proceso. Adicionalmente, dicho artículo garantiza que la implementación de esta estrategia de trabajo, contribuye considerablemente en la identificación de problemas y necesidades de usuarios, recopilación de nuevas perspectivas de solución, generación de opciones de solución y prototipos que permiten representar ideas. [7]

Por consiguiente, considerando que la esencia de este enfoque se orienta en el diseño centrado en el usuario, se ofrece una propuesta cuyo objetivo es hacer énfasis en la obtención de feedback desde etapas tempranas del Ciclo de Vida de V&V, con la aplicación de técnicas a lo largo del proceso y hasta la ejecución de las pruebas finales. Así los resultados de las evaluaciones permitirán asegurar la obtención de un producto usable y con un mayor nivel de calidad.

2.2 FUNDAMENTACIÓN TEÓRICA

Seguidamente, se detallan los lineamientos teóricos a utilizar que orientan el desarrollo de este Trabajo Final, de tal manera de cumplir con lo establecido en los objetivos planteados:

- ***¿Qué es Calidad de Software?***

  Según el Standard IEEE 610-1990 la calidad del Software es el grado con el que el sistema, componente o proceso cumple con los requerimientos especificados y las necesidades o expectativas del cliente o usuario. [8]

  La calidad es importante en el desarrollo de un producto o servicio y, más aún, en la creación de un producto de software, no solo porque busca cumplir con las expectativas del cliente, sino también para mejorar los procesos internos en la

elaboración de un producto, tarea fundamental en el crecimiento de la organización [9].

- ***¿Qué es la Verificación y Validación de Software?***

  El testing de software pertenece a una actividad o etapa del proceso de producción de software denominada Verificación y Validación, usualmente abreviada como V&V. V&V es el nombre genérico dado a las actividades de comprobación que aseguran que el software respeta su especificación y satisface las necesidades de sus usuarios [10].

  Dado que es un proceso complicado, los especialistas lo llevan a cabo en fases que componen el Ciclo de Vida del Testing de Software. En la Figura 2 se sintetiza el proceso mencionado, detallando cada una de las etapas [11]:

***Primer Fase*: Análisis de Requerimientos**	✓ Analizar los requerimientos documentados provistos por los desarrollistas. ✓ Comprender lo que los usuarios esperan que haga el software en base a los requerimientos documentados. ✓ Priorizar los requerimientos a probar
***Segunda Fase* Planificación de Pruebas**	✓ Definir el alcance de las pruebas detallando niveles, tipos, estrategias y técnicas de prueba tomando como referencia los requerimientos a probar.
***Tercer Fase*: Diseño y Desarrollo de Pruebas**	✓ Comprobar que los requerimientos documentados estén completos. ✓ Desarrollar casos de prueba en base a los requerimientos documentados. ✓ Diseñar todas las combinaciones posibles y priorizar cuál de ellas afecta más al producto software.
***Cuarta Fase*: Ejecución de Pruebas**	✓ Ejecutar los casos de prueba y documentar las novedades y desviaciones del comportamiento del software sin participación del usuario. ✓ Transmitir novedades para que se solucionen.
***Quinta Fase*: Evaluación de resultados**	✓ Validar si los requerimientos implementados cubren las expectativas de los usuarios.
***Sexta Fase*: Cierre del Proceso**	✓ Reportar las novedades del proceso y la aceptación/rechazo del software testeado, sin intervención del usuario final.

Figura 2. Fases del Ciclo de Vida de Verificación y Validación de Software

La figura anterior muetra que los requerimientos son comprendidos y analizados una sola vez y para todo el ciclo y que solo se interactúa con los usuarios en la fase de Evaluación de Resultados. Durante el proceso, los usuarios son ajenos al ciclo de vida de V&V, hasta que el producto se prueba y se determina que hace lo que ellos esperan. Por consiguiente, se llega al final del ciclo trabajando en una propuesta que, en determinas circunstancias, no cumple las expectativas de los usuarios, repercutiendo en la calidad del producto final.

Para aclarar el concepto de Software al cual se hace referencia, se da a conocer la siguiente diferenciación:

- ***Software Operativo***: son aquellos sistemas operativos y/o aplicaciones dedicadas a la actividad operativa. Ejemplo de ello son los simuladores de juegos de guerra, simuladores de cuadros de mando, simuladores de vuelo, incluyendo al software dedicado a la actividad de vuelo específicamente.
- ***Software Administrativo***: aquel software dedicado al manejo de información de carácter no operativo como puede ser administración de personal, páginas de internet, ofimática, etc. Este tipo de software no será objeto de análisis para este Trabajo Final [12].

- ***¿Qué son los Requerimientos?***

  Un requerimiento es una característica que debe cumplir el sistema, software o componente para alcanzar un objetivo. El conjunto de éstos crea un prototipo del sistema final, el cual es extraído de las ideas abstractas que representan las necesidades de los usuarios. Los mismos quedan establecidos en un documento denominado Especificación de Requerimientos, el cual se define luego de pasar un previo análisis. Deben ser consistentes, completos, realizables y verificables [13].

  La IEEE (Institute of Electrical and Electronics Engineers) determina la clasificación de dichos requerimientos en funcionales y no funcionales:

Los ***requerimientos funcionales*** describen cómo debe comportarse el sistema, software o componente ante determinado estímulo. Son declaraciones de los servicios que deben proporcionar, es decir, de la manera en que deben reaccionar a entradas particulares y de cómo deben comportarse en situaciones particulares.

Los ***requerimientos no funcionales*** describen una restricción o limitación sobre el sistema, software o componente, es decir, restringen los servicios o funciones ofrecidas por ellos.

Por lo tanto, los ***requerimientos funcionales*** describen lo que el sistema, software o componente debe hacer, mientras que los ***requerimientos no funcionales*** ponen límites y restricciones al sistema, software o componente. [14]

¿Qué es la Metodología de Pensamiento de Diseño (Design Thinking)?

El Pensamiento de Diseño, es una metodología utilizada para crear ideas innovadoras que centra su eficacia en entender y plantear soluciones a las necesidades reales de los usuarios. En términos generales, el pensamiento de diseño primero define un problema y luego implementa soluciones, siempre teniendo en cuenta las necesidades de los usuarios como núcleo del desarrollo de conceptos.

El fundamento del Pensamiento de Diseño es la acción y la creación, es decir, siempre busca crear y probar algo para seguir aprendiendo del usuario y mejorar el proceso de generación de ideas. Se trabaja principalmente con tres escenarios [15][16]:

***Inspiración*:** consiste en identificar la necesidad y el problema para analizar soluciones. Incluye las siguientes etapas:

- **Empatizar:** comprender lo que el usuario necesita, en términos de lo que siente el usuario.
- **Definir:** establecer las necesidades del usuario en términos de un problema que se puede resolver.

***Ideación**:* se basa en proponer una solución que satisfaga las necesidades del usuario.

Implementación: desarrollar la solución final, en la que intervienen las siguientes fases:

- **Prototipar:** crear una solución que cubra una selección de las características requeridas por el usuario, para que pueda experimentarlo.
- **Probar:** utilizar la solución real, de manera que el usuario pueda proporcionar comentarios oportunos y el proceso pueda iterar.

Según Brown y Wyatt, *"el Design Thinking se basa en nuestra capacidad para ser intuitivos, reconocer patrones, construir ideas que tienen significado emocional, además de ser funcionales, y para expresarnos por más medios que las palabras y símbolos. Nadie quiere dirigir una organización basada en el sentimiento, la intuición y la inspiración, pero un exceso de confianza en lo racional y lo analítico puede ser también un riesgo. El Design Thinking, como enfoque integrado en el núcleo del proceso de diseño, proporciona una tercera vía"* [17].

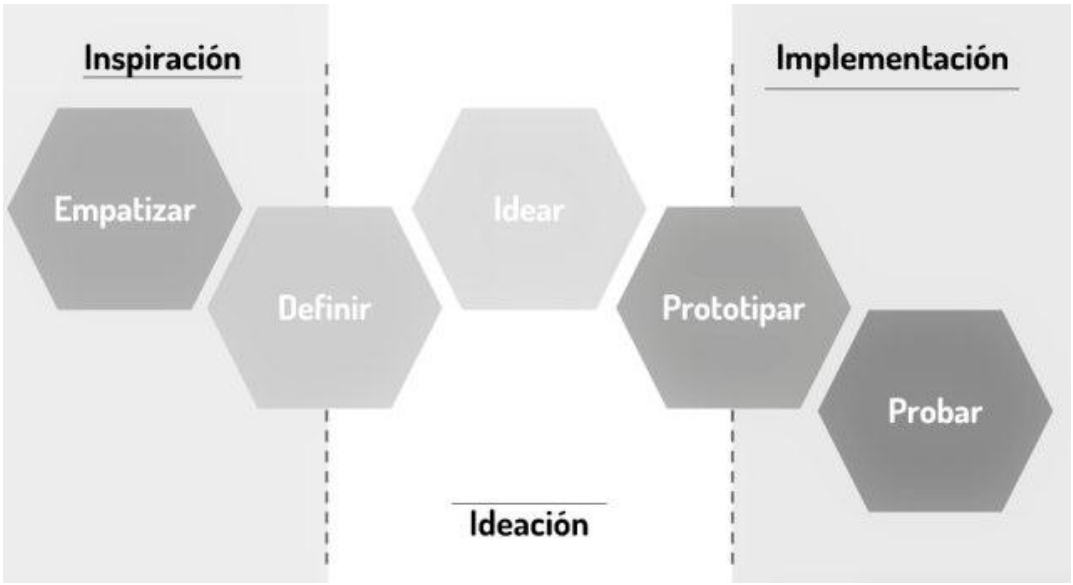

Figura 3 Etapas de la metodología del Pensamiento de Diseño.

- ***¿Qué es la Usabilidad?***

  De acuerdo al estándar ISO 9241-11, la usabilidad es “el grado en el que un producto puede ser utilizado por usuarios específicos para conseguir objetivos específicos con efectividad, eficiencia y satisfacción en un determinado contexto de uso. [18]

  Se la considera un atributo intangible del software, por lo tanto, es difícil de visualizar, medir y reconocer como un factor determinante de su calidad. Esto genera que un gran número de productos software tengan un nivel de usabilidad deficiente, cuando una mayor atención por este aspecto contribuiría a incrementar la calidad del producto percibido por el usuario, sin un aumento excesivo en el costo de desarrollo**.** [19]

  Alcanzar la usabilidad en el desarrollo de software implica emplear un Diseño Centrado en el Usuario. A fin de aplicarlo se debe tomar en cuenta las características del usuario, las actividades que realiza y el escenario donde se desempeña su actividad.

  Todos estos factores permitirán conocer cuáles son los requisitos que debe satisfacer el producto software para facilitar el uso del mismo. [20]

III ANÁLISIS DE LA PROBLEMÁTICA Y SOLUCIÓN PROPUESTA

En el capítulo I se describió la metodología de trabajo para llevar adelante el proyecto en cuatro fases. El presente capítulo detalla la Primer Fase, que consiste en ***Identificar y Analizar el Problema*** y la Segunda Fase, que se fundamenta en ***Generar la Solución***.

Para ello, se toma como base el análisis de la problemática actual en cada paso del proceso de V&V, para luego proponer la solución que permita resolver tales dificultades a través de técnicas y herramientas de Design Thinking. Como resultado, se obtiene el diseño de un marco de trabajo que combina ambos conceptos. Se espera que el mismo sea utilizable en futuros proyectos de Verificación y Validación de Software Operativo de la organización en cuestión.

3. SITUACIÓN ACTUAL DE LOS PROYECTOS DE DESARROLLO DE SOFTWARE EN EL CICLO DE VIDA DE V&V

3.1 PROBLEMAS EN LA FASE DE ANÁLISIS DE REQUERIMIENTOS

En esta fase se estudian y analizan los requerimientos funcionales y no funcionales disponibles y se trata de comprender lo que los usuarios esperan que haga el producto software. Se traducen los requerimientos de los usuarios a requerimientos de software y se los prioriza en función de su criticidad.

Sin embargo, algunos usuarios tienen poca o nula idea de qué es lo que quieren que haga el software a desarrollar. Aquellos que poseen software operando tendrán ideas sobre cómo mejorarlo, por el contrario, cuando el software se encuentra en sus etapas iniciales de diseño, esta fuente de inspiración, en ocasiones, no está disponible.

En Ingeniería de Software, la actividad de captura de requerimientos es la que cubre el entendimiento entre el usuario y el equipo de desarrollo. [21]

Los stakeholders son las personas a quienes se necesita acercarse durante la ejecución del proyecto de manera tal de poder aclarar cualquier aspecto de los requerimientos. Generalmente, son personas que tienen un gran interés en el

producto y poseen requerimientos definidos para el mismo, como el cliente, un usuario y cualquier persona que esté relacionada con el proceso de construcción de este [22]. En la organización objeto de estudio, los principales stakeholders son usuarios que utilizan software operativo como herramienta de entrenamiento, tal es el caso de los simuladores de vuelo, simuladores de ayuda humanitaria, simuladores UAV, entre otros. En consecuencia, es inevitable identificar e integrar a estos usuarios para comprender sus necesidades reales desde las primeras etapas del proceso de Verificación y Validación de Software, lo que permitiría evitar repercusiones negativas durante el progreso del proyecto.

A pesar de ello, y como se citó anteriormente, solo se interactúa con dichos usuarios en la definición de requerimientos con los desarrollistas, y luego en la fase de Evaluación de Resultados del ciclo de vida de V&V, sin participación en el resto del proceso, trayendo como consecuencia, en ocasiones, el retroceso a etapas anteriores. Este escenario se agrava aún más, cuando tales requerimientos son mal interpretados por los desarrollistas y luego son entregados a través de la documentación formal, al equipo de testing, quienes realizarán las pruebas en base a requerimientos mal definidos.

Es debido a estos inconvenientes que muchas veces los requisitos recibidos por el equipo de testing, no son suficientes o no reflejan realmente los deseos de los usuarios. Esto provoca demoras, aumento de costos, y en determinadas circunstancias que el proyecto fracase. Por tal razón, los stakeholders necesitan estar más involucrados en el proceso de V&V para lograr un resultado exitoso del producto.

3.2 PROBLEMAS EN LA FASE DE PLANIFICACIÓN DE PRUEBAS

Luego de comprender lo que se espera que haga el software se define qué se va a probar, cómo se va a probar, quién lo va a probar y cuando.

El objetivo principal de las pruebas de software es asegurar la calidad y funcionalidad del producto. Las mismas se realizan para identificar errores, defectos o fallos en el

software antes de su lanzamiento o implementación. Adicionalmente, ayudan a aumentar la confiabilidad del software, lo que genera confianza en los usuarios. Por otra parte, permiten evaluar la facilidad de uso, la interfaz de usuario y la experiencia del usuario en general. Esto garantiza que el software sea intuitivo.

Un paso crucial para garantizar lo enunciado anteriormente es la Planificación de las Pruebas; la cual consiste en definir el alcance de las mismas detallando niveles, tipos, estrategias y técnicas. Actualmente esta actividad se lleva a cabo tomando como referencia la documentación detallada en la Especificación de Requerimientos. Dicho documento especifica los requerimientos que fueron acordados entre desarrollistas y usuarios. Por lo tanto, el equipo de Testing los analiza determinando su criticidad y en función de esto establece cuáles de ellos serán probados en una primera etapa. En base a estos y a la revisión del software se resuelve qué pruebas se llevarán a cabo y cuáles serán excluidas, como también corresponderá detectar los Casos de Pruebas necesarios que servirán para verificar los resultados esperados. La participación temprana del usuario en estas instancias, ayudaría a identificar con mayor grado de objetividad los aspectos críticos a probar y establecer las prioridades adecuadas como parte de estrategia de verificación.

3.3 PROBLEMAS EN LA FASE DE DISEÑO Y DESARROLLO DE PRUEBAS

Tal como se citó anteriormente, los requerimientos de software cumplen un papel primordial en el proceso de V&V, ya que se enfocan en la definición de lo que se desea producir. Su principal tarea consiste en la generación de especificaciones correctas que describan con claridad, sin ambigüedades, en forma consistente y completa, el comportamiento del software; de esta manera, se pretende minimizar los defectos, fallas y errores que repercutan en la calidad del mismo [7].

No obstante, en la mayoría de los casos, los requisitos de software se especifican utilizando un lenguaje natural debido a su flexibilidad y simplicidad. Dicho lenguaje natural con frecuencia hace que la especificación sea propensa a la ambigüedad e

incompletitud. Por esta razón, antes de comenzar con el Diseño y Desarrollo de las Pruebas se propone seguir los criterios que establece el estándar IEEE 830. El cual determina que un requerimiento debe cumplir con ciertas características para ser considerado de calidad. Sin embargo, no siempre se tiene el éxito esperado, razón por la cual se pretende implementar técnicas de DT que ayuden a idear una adecuada definición de los mismos. A continuación, se presentan las particularidades más importantes que determina dicho estándar:

- ✓ **Necesario**: un requerimiento es necesario si su omisión provoca una deficiencia en el software a construir, y además su capacidad, características físicas o factor de calidad no pueden ser reemplazados por otras capacidades del producto o del proceso.
- ✓ **Conciso:** un requerimiento es conciso si es fácil de leer y entender. Su redacción debe ser simple y clara para aquellos que vayan a consultarlo.
- ✓ **Completo**: un requerimiento está completo si no necesita ampliar detalles en su redacción, es decir, si se proporciona la información suficiente para su comprensión.
- ✓ **Consistente**: un requerimiento es consistente si no es contradictorio con otro requerimiento.
- ✓ **No ambiguo**: Un requerimiento no es ambiguo cuando tiene una sola interpretación. El lenguaje usado en su definición, no debe causar confusiones al lector.
- ✓ **Verificable**: Un requerimiento es verificable cuando puede ser comprobado por medio de demostración o pruebas.
- ✓ **Clasificable**: no todos los requisitos son igual de importantes. Los requisitos pueden clasificarse por diversos criterios, por ejemplo, la importancia. De acuerdo a este criterio pueden ser esenciales, condicionales, opcionales.
- ✓ **Modificable**: un requerimiento es modificable si cualquier cambio puede realizarse de manera fácil, completa y consistente.

También es deseable evitar la redundancia, es decir que no aparezca un mismo requisito en más de un lugar.

- ✓ **Trazable**: los requisitos tienen un origen claro y el seguimiento del requerimiento se puede realizar de forma sencilla a lo largo del proyecto. Cuando un requisito representa una derivación de otro requisito, se debe facilitar tanto las referencias hacia atrás como hacia adelante.

Después de comprobar que los requerimientos cumplen con las especificaciones del estándar, se procede al Diseño y Desarrollo de casos de prueba que permiten la ejecución de las mismas. Los casos de prueba se definen como un conjunto de acciones que se ejecutan para autenticar la funcionalidad del software [23]. Es importante que se documenten incluyendo una clara trazabilidad con los Requerimientos. Asimismo, se considera relevante detallar las entradas (válidas o no válidas), los posibles resultados o salidas y la técnica que se utilizará para probar cada Requerimiento.

En esta fase es preeminente involucrar al usuario, de tal manera que aporte ideas innovadoras en la creación de escenarios de prueba realistas y relevantes. Su experiencia y conocimiento del dominio pueden ayudar a identificar posibles omisiones o casos de prueba incorrectos. Sin embargo, no se cuenta en esta instancia con la participación del mismo.

3.4 PROBLEMAS EN LA FASE DE EJECUCIÓN DE PRUEBAS

Esta fase consiste en identificar la desviación del comportamiento esperado del software a través de la ejecución de los casos de prueba diseñados en la etapa anterior. Se registran los casos fallidos, se documentan las novedades y se las transmite a los desarrolladores para que las solucionen.

Luego se llevan a cabo pruebas de regresión para garantizar que el software funcione adecuadamente después de implementar un cambio, ya que la corrección de las

novedades encontradas puede crear un error en otro sector o módulo del producto. Se considera sustancial en este paso validar con los usuarios la criticidad de las novedades, a fin de que contribuyan a priorizarlas según su gravedad y utilidad. Los datos a probar deben reflejar los datos que el usuario considere prioritarios y las soluciones esperadas deben ser también el reflejo de sus necesidades. No obstante, es indispensable en este paso, considerar la apreciación del usuario mediante el trato directo con los mismos, para evitar de esta manera, que la apreciación del equipo de Testing con respecto a la severidad de la novedad, no se corresponda con el criterio del usuario final.

3.5 PROBLEMAS EN LA FASE DE EVALUACIÓN DE RESULTADOS

En esta fase se determina si se han alcanzado los objetivos de las pruebas, es decir, si la implementación de los requerimientos fue la óptima. De igual modo, se valida que cubran las expectativas de los usuarios. Es en este momento donde el equipo de testing involucra a los usuarios para garantizar que el producto software se corresponda con los requerimientos definidos, a través de las pruebas de aceptación. Las mismas son diseñadas por el propio equipo en base a los requisitos especificados y ejecutadas por el usuario final. Estas pruebas no son ejecutadas por todos, pero sí por una cantidad de usuarios finales significativo que den validez y conformidad al producto que se les entrega en base a lo que se acordó inicialmente. Es aquí también donde se evalúa la facilidad de uso y la experiencia del usuario con el software [24].
Por ser esta etapa de alta importancia es necesario abordar la intervención del usuario desde el inicio del proceso, ofreciendo así mecanismos de detección precoz de inconsistencias.

Es recién en esta etapa del proceso de V&V, donde el usuario se involucra para validar su conformidad con respecto a lo solicitado, motivo por el cual, a menudo, se tiene que retroceder a etapas anteriores, lo que implica pérdida de tiempo y dinero.

3.6 PROBLEMAS EN LA FASE DE CIERRE DEL PROCESO

Esta es la última fase del proceso de pruebas donde se documenta y se prepara el informe final considerando los niveles de aceptación del software testeado. Dicha instancia se lleva a cabo sin la intervención del usuario.

Aquí también se puede evaluar el proceso de testing teniendo en cuenta el aprendizaje obtenido durante las pruebas o fallas en las estrategias y tomar decisiones basadas en los hallazgos alcanzados.

La fase de cierre establece formalmente la calidad del software probado y compara los resultados con los criterios de aceptación y los objetivos establecidos durante la planificación. En esta etapa se documentan las novedades importantes encontradas durante las pruebas, esto incluye errores, problemas de rendimiento, mejoras sugeridas y cualquier otra información relevante. Los resultados, informes y documentación generados se revisan y se obtiene la aprobación de los responsables del proyecto.

Es importante destacar que en el cierre del proceso es indispensable participar a los usuarios para escuchar comentarios y sugerencias que ayuden a definir si es necesario volver a repetir el ciclo. Los resultados obtenidos pueden influir en futuras iteraciones de las pruebas, así como en la mejora continua de los procesos de desarrollo y calidad del software. Por tal motivo, en esta fase, al igual que en todas las fases anteriormente mencionadas, la participación del usuario es crucial.

3.7 SOLUCIÓN A LA PROBLEMÁTICA ACTUAL

Partiendo de la problemática analizada previamente, se puede deducir que existe una potencial oportunidad para aplicar Design Thinking en tal contexto. Es posible incluir técnicas y herramientas de DT en el Ciclo de Vida de V&V y dar solución a los

inconvenientes presentados. Para ello se definirá un marco de trabajo que permita integrar ambos conceptos.

Está claro que, dentro del ciclo mencionado, existen etapas en las que debería involucrarse al usuario. Son ellos a quienes tenemos que tomar en cuenta para aplicar la propuesta sugerida, dándoles lugar a manifestar sus necesidades, con el fin de beneficiarlos aún más con el producto software.

La propuesta de trabajo que se desea implementar, busca que Design Thinking acompañe los pasos y actividades del Ciclo de Vida de V&V, encontrando una sinergia que permita superar fallas desde la identificación de la necesidad inicial del usuario hasta el cierre del proceso. De esta manera, se reducirán las dificultades ocasionadas a lo largo de estas etapas y permitirán mejorar la calidad de los productos desarrollados.

3.7.1 APLICACIÓN DE DESIGN THINKING EN EL CICLO DE VIDA DE VERIFICACIÓN Y VALIDACIÓN DE SOFTWARE

Tal y como se describe anteriormente, se considera trabajar en las seis fases del Ciclo de Vida de V&V: **Análisis de Requerimientos, Planificación de Pruebas, Diseño y Desarrollo de Pruebas, Ejecución de Pruebas, Evaluación de Resultados y Cierre del Proceso**. En dichas fases, se aplicarán las etapas de Design Thinking: **Empatizar, Definir, Idear, Prototipar y Probar**, además de considerar las técnicas y herramientas necesarias que permitirán abordar la problemática en cada paso. La siguiente figura resume detalladamente el marco de trabajo, considerando las fases del Ciclo de Vida de Verificación y Validación de Software y los pasos de la metodología Design Thinking.

Figura 4 Marco de Trabajo

Como se puede apreciar, en las fases de **Análisis de Requerimientos** y **Planificación de Pruebas** se consideran los pasos empatizar y definir, con la finalidad de comprender y establecer lo que el usuario necesita.

Las fases de **Diseño y Desarrollo de Pruebas** y de **Ejecución de Pruebas** aplican los conceptos idear y prototipar para proponer soluciones innovadoras y desarrollarlas.

La fase de **Evaluación de Resultados**, por un lado, comprueba la facilidad de uso y la experiencia del usuario al interactuar con un prototipo de software, lo cual permite identificar si el producto es amigable y satisfactorio. Por otro lado, determina a través de encuestas, si el software prototipado cumple con los requisitos y expectativas del usuario y si está listo para su implementación en un entorno de producción.

La siguiente fase que consiste en el **Cierre de Proceso** se enfoca en evaluar con el usuario las novedades de dicho proceso, para determinar conjuntamente si es necesario llevar a cabo un nuevo ciclo.

IV IMPLEMENTACIÓN DE LA SOLUCIÓN

Continuando con la metodología de trabajo descripta en el capítulo I, en este apartado se detalla la tercera fase, ***implementación del marco de trabajo,*** ejecutándolo de forma directa en un caso real.

Primero se define el software que permite poner en práctica el desafío mencionado y luego se describen los pasos de ejecución del marco de trabajo y su aplicación en el proceso de V&V.

4. DEFINICION DE SOFTWARE Y EJECUCIÓN DE MARCO DE TRABAJO

4.1 DEFINICIÓN DEL SOFTWARE

Tal cual se mencionó, la implementación de dicha metodología será aplicada al software del proyecto FAS D - AV 0180 - “Desarrollo de un Sistema de Debriefing”, desarrollado por el Centro de Investigación y Desarrollo de Tecnologías Aeronáuticas (CITeA)

El Sistema de Debriefing es un software operativo e interactivo de visualización y análisis de ejercicios pos vuelo que ofrece la posibilidad de llevar a cabo la reproducción de los mismos, los cuales se registran mediante un Generador de Vuelos.

Dicho software recibe como entradas los archivos generados por el software Generador de Vuelos a partir de los datos de los distintos sistemas de armas. El Sistema de Debriefing permite en base a estos datos simular el vuelo junto a la reproducción sincronizada de los datos ingresados. Así mismo, ofrece la posibilidad de llevar a cabo un análisis de maniobras y una evaluación técnica del vuelo permitiendo la representación en 2D y 3D.

De esta manera, el usuario dispone de información con la que puede llevar a cabo un estudio más detallado del vuelo y con la posibilidad de solicitar la indicación de la distancia real entre aeronaves.

Del mismo modo, cuenta con la capacidad de reproducir hasta 20 aeronaves de forma simultánea provenientes de diversas fuentes de datos. Estas incluyen sistemas ADS-B,

CITeA CHIP, Sarm IA-63 Pampa II, Sarm A4AR, IA-58 Pucará. Dichos sistemas de armas corresponden a diferentes brigadas distribuidas geográficamente a lo largo del país [25].

4.2 EJECUCIÓN DEL MARCO DE TRABAJO

El primer paso importante para la resolución de Problemas en la Fase de Análisis de Requerimientos, es la definición formal de todos los usuarios involucrados en el proyecto Sistema Debriefing. Para ello se utiliza como herramienta el Mapa de Actores, que permite identificar, de manera eficiente y ordenada, los stakeholders principales, clasificados por la relación que tienen con el mismo.

Como se mencionó en el capítulo III, el análisis de requerimientos es la actividad más crítica, ya que, para dar solución a la problemática explicada, se requiere de la colaboración de diferentes stakeholders. Para nuestro proyecto objeto de estudio, los mismos están distribuidos geográficamente y no necesariamente son de la misma área de conocimiento. Esta es la primera etapa de construcción del entendimiento del problema que se debe resolver. Es fundamentalmente una actividad humana en la cual se identifican los stakeholders y se establecen las relaciones que éstos van a tener a lo largo del proyecto. Por lo cual, es prioritario que en esta tarea tenga intervención el equipo de testers. Por consiguiente, se lleva a cabo un análisis que incluye tres pasos principales, los cuales se detallan a continuación**:** [26] [27]

- ✓ **Identificación de Actores:** se determinan quiénes son los actores en un proyecto. Se utiliza la lluvia de ideas, esta técnica aporta ideas sobre quienes pueden ser los posibles stakeholders.

- ✓ **Priorización de los actores:** Una vez completada la lista de posibles actores se inicia el proceso de priorización donde se detallan las principales categorías en las que pueden situarse los actores:

- **Usuario:** se coloca en el círculo central. El mapa parte de él, y de las relaciones que establece con los actores colocados en los otros círculos.
- **Actores Internos:** Se sitúan en el círculo más cercano al central. En él, se colocan aquellos actores que interactúan de la forma más directa posible con el usuario.
- **Actores Externos:** son aquellos cuya relación con los actores internos hace posible la producción o entrega del producto o servicio al usuario. Se colocan dentro del segundo círculo.
- **Administraciones Públicas:** son aquellas entidades públicas que se relacionan de forma directa o indirecta (a través de otros actores) con el usuario.

✓ **Comprensión de los actores:** Es crucial conocer las opiniones de las personas sobre el proyecto. Esto puede tener un enorme impacto en la dinámica de todas las relaciones que se desarrollen entre los participantes

Como resultado de lo descripto anteriormente, se confeccionó la siguiente plantilla en la cual se detallan los stakeholders intervinientes. Esta herramienta, denominada Mapa de Actores nos ayuda a identificar a los usuarios que participan en el proyecto de software.

MAPA DE ACTORES

Objetivo: Identificar a los actores que participan en el proyecto.

Proyecto:	Versión:
Equipo:	Fecha:
Observaciones:	

Figura 5 Mapa de Actores

Listado de Actores

Usuarios:

- Pilotos de Brigada
- Instructores de Brigada

Actores Internos:

- Jefes de Brigadas

Actores Externos:

- Jefe de Proyecto
- Jefe de Verificación y Validación
- Jefe de Ensayo en Vuelo
- Desarrolladores
- Testers

Instituciones Públicas:

- No intervienen

El segundo paso en la problemática planteada es comprender lo que los usuarios esperan que haga el software. Por consiguiente, luego de identificar a los actores se realizaron entrevistas de empatía, con la finalidad de obtener información sobre los deseos y expectativas anheladas por los stakeholders. Las mismas fueron registradas en el documento detallado a continuación denominado Mapa de Empatía, el cual nos ayuda a entender y analizar sus necesidades y requerimientos. El siguiente ejemplo fue realizado para un instructor de vuelo de brigada. Cabe aclarar que es necesario realizar tal ejercicio para cada usuario a investigar.

MAPA DE EMPATIA

Objetivo: aumentar la empatía con los usuarios investigados

Proyecto:	**Versión:**
Equipo:	**Fecha:**
Observaciones:	

Figura 6 Mapa de Empatía

Uno de los principios claves de Design Thinking es su enfoque orientado hacia los valores humanos en cada fase del proceso. La empatía hacia las personas es fundamental, ya que permite trabajar para entender a las personas.

Observar lo que hacen e interactuar con ellos otorga pistas sobre cómo piensan y sienten y ayuda a que el tester aprenda sobre lo que necesitan [21].

Sin embargo, a menudo sucede que es necesario priorizar las necesidades anheladas por los usuarios ya que es imposible probarlas a todas a la vez. Por lo tanto, se deben tomar decisiones sobre qué conjunto de requerimientos es imprescindible testear primero y cuáles se pueden retrasar hasta una etapa posterior.

A causa de ello, se solicitó a cada usuario que especifique los requerimientos que son prioritarios para el uso del software a operar. Por tal motivo, se elaboró el siguiente documento que utiliza la técnica de los 5 porqués [28]. Con esta herramienta el tester pregunta repetidamente al usuario (cinco veces o menos) por qué el requerimiento es necesario hasta que se establece la importancia de los mismos. Las respuestas revelan si el requisito es realmente necesario o si se puede cancelar o posponer una vez que se determina la prioridad. De esta manera se logra identificar y definir cuáles son los requerimientos que reflejan preeminentemente los deseos de los usuarios.

LOS 5 PORQUÉS

Objetivo: Definir qué requerimientos son prioritarios

Proyecto:	Versión:
Equipo:	Fecha:
Observaciones:	

Los 5 porqués

Planteamiento del Requisito	Por qué 1	Por qué 2	Por qué 3	Por qué 4	Por qué 5	Prioridad
Representar un vuelo en 3D	Para obtener diferentes vistas en 360º	**¿Para qué obtener vistas en 360°?** Para adquirir alejamiento/ acercamiento del objeto 3 D en todos los ángulos	**¿Para qué adquirir alejamiento/ acercamiento en 3D?** Para visualizar con exactitud el cálculo de altura	**¿Para qué visualizar con exactitud el cálculo de la altura?** Para obtener información precisa		Dar prioridad a este requerimiento
Solicitar Ayuda en Línea	Para acceder mediante el sistema al manual de usuario en pdf	**¿Para qué acceder al manual de usuario en pdf?** Para visualizar en pantalla detalles del uso del software	**¿Para qué visualizar en pantalla detalles del uso del software?** Para obtener información a cerca de las funcionalidades del del mismo	**¿Para qué obtener información de las funcionalidades del software?** Para disponer de ayuda en línea	**¿Para qué disponer de ayuda en línea?** Para consultar inquietudes	Postergar el requerimiento para la segunda etapa

Figura 7 Los 5 Porqués

Posteriormente y continuando con nuestro marco de trabajo, nos enfocamos en resolver la problemática definida para la fase de Planificación de Pruebas. Partiendo de los requerimientos priorizados, se lleva a cabo un análisis del software a probar con la participación de los usuarios. Para ello se hizo uso de la herramienta conocida como Observación Encubierta [28], la cual permite obtener información objetiva acerca de lo que se probará y cómo se probará sin interferir con los usuarios, provocando algún tipo de influencia. Se trata de observar al usuario interactuando con el software, sin entorpecer en sus acciones. A continuación, se muestra la documentación utilizada para tal fin:

OBSERVACIÓN ENCUBIERTA

Objetivo: obtener información objetiva sin interferir al usuario.

Proyecto:	Versión:
Equipo:	Fecha:
Observaciones:	

Observación Encubierta

Tipo de Usuario	Fecha	Lugar	¿Qué queremos investigar? ¿Por qué?	Conclusiones
Piloto de Brigada	25/8/2023	Puesto de Testing	Analizar el requeriminto Generar Vuelo y cada uno de los pasos que involucra para determinar qué tipo de pruebas, técnicas y estrategias utilizar.	Será necesario probar tal requerimiento aplicando pruebas de integración, de sistemas y de aceptación quedando excluidas las pruebas unitarias.
Instructor de Vuelo	5/9/2023	Puesto de Testing	Analizar el requeriminto Representar Vuelo y cada uno de los pasos que involucra para determinar qué tipo de pruebas, técnicas y estrategias utilizar.	Será necesario probar tal requerimiento aplicando pruebas de integración, de sistemas y de aceptación quedando excluidas las pruebas unitarias.

Figura 8 Observación Encubierta

Una vez concluída la actividad llevada a cabo en el paso anterior es necesario concretar el Plan de Pruebas, según lo establece el Estandar IEEE 829 [29].

Esta normativa define una estructura clara a cumplir en un plan de pruebas para que sea lo más correcta posible:

1. **Identificador único:** todo plan debe tener un identificador único. Los planes de prueba son de naturaleza dinámica, y deben mantenerse actualizados. Se debe incluir información del autor, historial de revisiones y versionado.

2. **Introducción:** indicar el propósito del plan e incluir el alcance de las pruebas, según sus niveles, tipos y técnicas.

3. **Riesgos asociados**: describir aquellas funciones muy complejas, módulos mal documentados, requisitos poco claros o que no pueden ser probados.

4. **Características a probar:** incluir una lista de lo que se probará.

5. **Características que no se probarán**: incluir una lista de lo que no se probará.

6. **Estrategia:** identificar las estrategias que se utilizarán y evaluar si requieren de capacitación. Del mismo modo indicar las métricas que se utilizarán para evaluar la calidad del software.

7. **Criterios de Aceptación y Rechazo:** establecer cuáles son los criterios de Aceptación/Rechazo del Software de acuerdo al número y la gravedad de defectos encontrados.

8. **Criterios de Suspensión y Reanudación**: especificar los criterios para la interrupción de una prueba o serie de pruebas que permitirán que estas continúen o no más allá de los mismos.

9. **Entrega de plan de pruebas**: entregar el plan de pruebas, con la identificación tentativa de los casos de prueba.

10. **Necesidades ambientales:** identificar requisitos especiales para el plan de prueba, como por ejemplo hardware específico o versiones específicas de otro software de soporte.

11. **Responsabilidades:** incluir las responsabilidades de las personas involucradas en las áreas del plan.

12. **Calendario:** fijar la estimación de las pruebas por medio de un cronograma tentativo.

13. **Planificación de riesgos y contingencias:** estimar cuales son los riesgos asociados en la planificación de las pruebas como, por ejemplo, falta de personal cuando se inician las pruebas; falta de disponibilidad de hardware o software requeridos, retrasos en la capacitación del software a probar y cambios en los requisitos a probar. Por estas razones, el cronograma muchas veces debe modificarse, para lo cual es importante evitar reducir el número de pruebas, de tal manera de prevenir la implementación de un producto de baja calidad.

Luego de definir el Plan de Pruebas continuamos con la implementación del marco de trabajo en la fase de Diseño y Desarrollo de Casos de Prueba. En esta instancia, el primer punto a abordar consiste en comprobar que los requerimientos a probar cumplan con los criterios establecidos en el estándar IEEE 830 [30]. Para desarrollar requisitos completos y

de calidad se utiliza la técnica de casos de uso [28], la cual permite transformar requerimientos en escenarios reales, contemplando todos los caminos que pueden presentarse al momento de operar el software. De igual manera, describen las interacciones entre el usuario y el software, centrándose en qué hace el software para el usuario. Se materializan en un documento que plasma las secuencias de pasos que pueden idearse a lo largo del uso de una solución, según los distintos tipos de usuario.

Esta técnica permite asegurar que los requerimientos a testear cubran los criterios citados por la normativa mencionada.

CASO DE USO

Objetivo: idear escenarios reales y asegurar que cumplan con el estándar de calidad.

Proyecto:	**Versión:**
Equipo:	**Fecha:**
Observaciones:	

CASO DE USO	Generar Vuelos				
ID CASO DE USO	CU02				
VERSION	1.0				
TRAZABILIDAD	CU01-CU03-CU04				
DESCRIPCION	Genera el o los vuelos de diferentes entradas de datos.				
PRECONDICION	Archivos generados por sistemas de grabación de datos de distintos sistemas de armas.				
CURSO NORMAL	PASOS		CURSO ALTERNO	PASOS	
	USUARIO	SISTEMA		USUARIO	SISTEMA
	1) El usuario ingresa al sistema y ejecuta el Generador de vuelos.	1) El sistema visualiza la interfaz Generador de Vuelos para el Sistema Debriefing.			
	2) El usuario activa el calendario emergente.	2) El sistema muestra el calendario emergente.			
	3) El usuario selecciona una fecha actual o pasada, elije entre los vuelos disponibles y los carga.	3) El sistema visualiza una grilla con información acerca de los vuelos seleccionados por el usuario.		3.1 El usuario elije vuelos cuyas entradas no son válidas. 3.2 El usuario seleccina mas de 20 vuelos. 3.3 El usuario no refresca vuelos disponibles durante el uso del software.	3.1 El sistema visualiza un error. 3.2 El sistema visualiza un error. 3.3 El sistema no carga la lista con los vuelos disponibles.
	4)El usuario selecciona cerrar la aplicación.	4)Fin de Caso de Uso			

Figura 9 Caso de Uso

Como se muestra en la figura 9, un caso de uso se compone de uno o más escenarios. Cada camino que se puede seguir en el caso de uso, ya sea curso normal o curso alterno, es un escenario de caso de uso.

Para ejecutar pruebas es necesario diseñarlas, tomando como referencia los casos de uso ideados anteriormente. El diseño y desarrollo de pruebas se lleva a cabo utilizando casos de prueba, los cuales se concretan en un documento donde se establecen condiciones para probar que cada caso de uso se aplique correctamente.

El propósito de las pruebas de software es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada. Son parte del proceso de control de calidad, contribuyendo a reducir el riesgo de encontrar errores en el momento de operar el software.

Existen diversos tipos de pruebas y cada tipo de prueba proporciona diferentes correcciones que permiten en su conjunto obtener un software con la menor probabilidad de fallos [31].

Figura 10 Niveles de Pruebas

En la figura anterior se describen los niveles de prueba existentes, correspondiendo el primer nivel, **a las pruebas unitarias**.

Un componente es la unidad más pequeña especificada de un software; las **pruebas unitarias** se llevan a cabo tras la construcción o realización de cada componente con el objeto de verificar que la implementación se esté llevando conforme a los estándares acordados.

Es importante mencionar, que estas pruebas las ejecuta el desarrollador verificando que su código cumple con lo solicitado y no ha violado ningún estándar que ponga en riesgo la estabilidad del software.

El segundo nivel de la pirámide hace referencia a las **pruebas de integración**, también conocidas como pruebas de interfaz, ya que comprueban la interacción entre componentes.

Las **pruebas de integración** asumen que los módulos ya han sido probados de manera individual, a través de las pruebas unitarias. Las mismas son llevadas a cabo por el equipo de testing.

Las **pruebas de sistema** se realizan cuando todo el desarrollo ha sido culminado y se cuenta con una versión preliminar del software que saldrá a producción. Esta etapa consiste en probar un sistema integrado con el objeto de comprobar el cumplimiento de requisitos especificados. Se realizan con un enfoque basado en el punto de vista del usuario y se desarrollan por el equipo de testing utilizando casos de prueba funcionales y no funcionales.

Las pruebas de sistema funcionales confirman que los requisitos para un uso específico previsto han sido cumplidos.

Las pruebas de sistema no funcionales verifican los atributos de calidad no funcionales, establecidos en el estándar 9126 [32]. Dicha normativa establece que la calidad del software se divide en atributos funcionales y atributos no funcionales, según se describe a continuación:

Atributos funcionales:

- ✓ Adecuación
- ✓ Exactitud
- ✓ Interoperabilidad
- ✓ Seguridad
- ✓ Cumplimiento de la funcionalidad

Atributos no funcionales:

- ✓ Usabilidad
- ✓ Eficiencia
- ✓ Mantenibilidad
- ✓ Portabilidad
- ✓ Fiabilidad

Cuando el software evaluado cumple con todas las características mencionadas anteriormente, podemos decir que es un software de calidad.

El último nivel detallado en la pirámide lo conforman las **pruebas de aceptación**, cuyo objetivo es obtener la aceptación o rechazo del usuario final. En esta etapa no se deberían encontrar defectos funcionales graves en el sistema. Se puede decir que las **pruebas de aceptación** son las pruebas de sistema ejecutadas por parte del usuario. Son consideradas como la fase final del proceso para crear una confianza en que el producto es apropiado para su uso, de esta manera se validará que el software satisface los requisitos del usuario. [33].

Anteriormente se mencionaron los niveles de pruebas que intervienen en el proceso de V&V. Para cada nivel de prueba existen distintos tipos de prueba y para cada tipo de prueba se aplican diferentes técnicas. Tal y cual se mencionó en párrafos anteriores, en la ejecución de las mismas, se considera importante el diseño y desarrollo de casos de prueba como herramienta necesaria para verificar los resultados esperados. Un caso de

prueba se define como la documentación que especifica las acciones a seguir para llegar a un objetivo específico (resultado esperado) que contiene datos de entrada, resultados esperados y un conjunto de condiciones de ejecución de un elemento de prueba. La herramienta de Design Thinking apropiada para simular casos de prueba, se denomina Prototipo de Software en Papel [28]. Se lleva a cabo a través de la representación en papel de un conjunto de entradas de datos que ejecutan un escenario de caso de uso para obtener datos de salida esperados. Ofrece una manera ágil y flexible de hacer tangible la navegación del software y se realiza en sesiones con usuarios finales. Se materializa a través de planillas que permiten representar pasos de ejecución, datos de entrada, resultados esperados, trazabilidad, niveles y tipos de prueba implementados:

PROTOTIPO DE SOFTWARE EN PAPEL

Objetivo: representar la ejecución en papel de un escenario de caso de uso

Proyecto:	Versión:
Equipo:	Fecha:
Observaciones:	

ID	Nombre	Descripción	Nivel de Prueba	Tipo de Prueba	Trazabilidad	Precondición	Pasos para la Ejec.	Datos de Entrada	Resultado Esperado	Resultado Actual	Estado	Autor					
												Caso de Prueba	Ejecucion C. de Prueba	Version	Resultado de la Ejecución	Fecha de Ejecución	
CP-GV-01	Generar Vuelos	Genera el o los vuelos de diferentes entradas de datos para ser reproducido en el Debriefing.	Prueba de Integración Prueba de Sistema	Prueba Funcional, GUI, Usabilidad, Documentación	CU01, CU03, CU04	Archivos generados por sistemas de grabación de datos de distintos sistemas de armas.	1	Hacer click en la opción file del menú desplegable. Seleccionar DS Flight Generator	Se debe abrir una ventana emergente "Generador de vuelo para el sistema de Debriefing-CITea"	Se abrió la ventana emergente.	Novedad	Ing.Filoniuk Ing. Martínez	Ing. Chiapponi An. Sist Dir	1.0.9	OK	13/12/2023	
							2	Seleccionar la fecha del vuelo o vuelos que se desea generar a través del selector de fechas, y luego seleccionar vuelos marcando una o mas opciones desde el casillero de verificación.	Se debe abrir el selector de fechas y visualizarse los vuelos seleccionados en la fecha elegida.	Se visualiza un árbol con vuelos disponibles donde se muestran todas las fechas de los vuelos. Se evidencia ausencia de filtro en cuanto a la fecha seleccionada y los vuelos correspondientes a dicha fecha. Asimismo se observa que el formato de fecha correspondiente al IA63 Pampa, no se condice con un formato de fecha convencional (lenguaje familiar al usuario)						
							3	Presionar el botón cargar.	Se debe visualizar una grilla con información de los vuelos correspondientes a la fecha seleccionada.	Se visualiza una grilla con información de los vuelos correspondientes a la fecha seleccionada.							
							4	Seleccionar de la grilla un vuelo previamente cargado.	Se debe visualizar el ckecklist tildado y se debe habilitar el botón generar.	Se visualizó el ckecklist tildado y se habilitó el botón generar.							

Figura 11 Prototipo de Software en Papel

Siguiendo con la implementación de nuestro marco de trabajo la siguiente problemática a resolver se enfoca en la Ejecución de Pruebas. Es en esta fase, donde se ejecutan los casos de prueba que surgen de la planilla anterior en la versión preliminar de software y se documentan las novedades y desviaciones de su comportamiento, con el objetivo de idear soluciones que contribuyan a satisfacer las necesidades de los usuarios.

Los resultados obtenidos en la ejecución de las pruebas se evidencian en una planilla de reporte de defectos, la cual es utilizada con el fin de validar conjuntamente con los usuarios la criticidad de las novedades y proponer soluciones que contribuyan a corregir los defectos y errores encontrados. Las soluciones esperadas deben ser el reflejo de las necesidades de los usuarios finales.

La documentación que se muestra a continuación revela la técnica de Lluvia de Ideas [28] implementada y sugerida por la metodología Design Thinking, donde se generaron ideas para la resolución de novedades encontradas en cada caso de prueba, ajustándose a propuestas anheladas por los usuarios.

LLUVIA DE IDEAS

Objetivo: documentar los defectos obtenidos en la ejecución de los casos de prueba y sugerir ideas para dar solución a tales desviaciones.

Proyecto:	Versión:
Equipo:	Fecha:
Observaciones:	

PLANILLA REPORTE DE DEFECTOS

ID Caso de Prueba	Nombre Caso Prueba	Nombre Defecto	Descripción	Evidencia	Estado	Severidad	Idea Sugerida en Función del Tipo de Novedad	Ambiente	Fecha Detectada	Responsable	Asignado
CP GV 01	Generar Vuelos	Ausencia Filtro para fecha seleccionada	Paso 2: Se visualiza un árbol con vuelos disponibles donde se muestran todas las fechas de los vuelos. Se evidencia ausencia de filtro en cuanto a la fecha seleccionada y los vuelos correspondientes a dicha fecha. Asimismo se observa que el formato de fecha correspondiente al IAGB Pampa, no se condice con un formato de fecha convencional (lenguaje familiar al usuario)		ABIERTO	MEDIO	**Novedades**: Funcional, GUI, Usabilidad (Heurística 2): **Ideas Sugeridas**: Garantizar el correcto funcionamiento del software tomando como referencia los atributos de calidad definidos en la norma ISO 9126, para evitar incomodidad a los usuarios. Asegurar que haya coincidencia del software con el mundo real. El mismo debe hablar el mismo lenguaje que los usuarios, en lugar de usar términos propios, que pudieran llegar a confundirlo.	TESTING	13/12/2023	Equipo Testing	Desarrollador
CP GV 05	Guardar vuelo generado	Imposibilidad para cambiar el nombre del archivo que figura por defecto	Paso 3: El sistema permite guardar el archivo con el nombre que figura por defecto, caso contrario, se visualiza un mensaje de error indicando que el archivo a guardar tiene un nombre incorrecto.		ABIERTO	MEDIO	**Novedades**: Funcional, GUI, Usabilidad (Heurística 4): **Ideas Sugeridas**: Se sugiere corregir la capacidad del software para dar respuesta al usuario con precisión y de la manera esperada. Garantizar las convenciones del mercado, apelando a ciertos estándares que hagan sentir cómodo al usuario; por tal motivo será conveniente permitir al usuario guardar el archivo con el nombre que desee.	TESTING	13/12/2023	Equipo Testing	Desarrollador
CP GV 06	Activar y desactivar un vuelo	Visualización de información de transacción anterior al desactivar un vuelo	Paso 3: Se desactivaron los vuelos seleccionados de la grilla. Se deshabilitó la sección derecha y se habilitó la sección izquierda, posibilitando comenzar nuevamente con el proceso. Se evidencia en la sección de vuelos disponibles información de la transacción anterior.		ABIERTO	MEJORA	**Novedades**: Funcional, Usabilidad (Heurística 1): **Ideas Sugeridas**: Garantizar la funcionalidad del software respetando los estandares de calidad, de tal manera que el usuario pueda operar correctamente. Asegurar que el software mantenga siempre informado al usuario sobre lo que está pasando mediante el feedback apropiado. Por cada acción que el usuario realice en el software, este debe tener una reacción en la pantalla de lo que está pasando en ese momento. Por tal motivo, sería conveniente que no visualice información de la transacción anterior, para no confundir al usuario.	TESTING	13/12/2023	Equipo Testing	Desarrollador
CP GV 07	Generar Vuelos con archivos no válidos	Generar un vuelo cuya entrada no es válida	Paso 3: No se visualizó mensaje indicando que no hay ningún vuelo seleccionado		ABIERTO	MEJORA	**Novedades**: GUI, Usabilidad (Heurística 1) **Ideas Sugeridas**: Asegurar que el software cumpla con los estandares de mercado. Permitir que el software mantenga siempre informado al usuario sobre lo que está pasando mediante el feedback apropiado, con mensajes comprensibles para el mismo.	TESTING	13/12/2023	Equipo Testing	Desarrollador
CP GG 01	Generar Gráficos	Ausencia de mensaje indicando que el vuelo fue cargado exitosamente.	Paso 2: Se visualizó la ruta donde se guardan los vuelos generados y se cargó el vuelo. Se evidencia ausencia de mensaje que describa que el vuelo fue cargado exitosamente.		ABIERTO	MEJORA	**Nodedades**: GUI, Usabilidad (Heurística 1) **Ideas Sugeridas**: Avalar que el software cumpla con los estandares de mercado. Asegurar qu el software mantenga siempre informado al usuario sobre lo que está pasando mediante el feedback apropiado, con mensajes comprensibles para el mismo.	TESTING	15/12/2023	Equipo Testing	Desarrollador
			Paso 4: Se seleccionó la variable y se generó exitosamente el gráfico correspondiente a dicha variable. La opción Generate Graphic no cumplió ninguna función, ya que al seleccionar la variable a graficar el gráfico se genera exitosamente.		ABIERTO	MEJORA	**Novedades**: Documentación **Idea Sugerida**: Garantizar que la información documentada en el Manual de Usuario coincida con la Funcionalidad del Software.				
CP GG 03	Insertar título y guardar gráfico	Ausencia de mensaje indicando que el archivo fue cargado exitosamente.	Paso 4: El archivo se guardó correctamente pero no se visualizó un mensaje indicando que fue guardado.		ABIERTO	MEJORA	**Novedades**: GUI, Usabilidad (Heurística 1) **Idea Sugerida**: Avalar que el software cumpla con los estandares de mercado. Asegurar qu el software mantenga siempre informado al usuario sobre lo que está pasando mediante el feedback apropiado, con mensajes comprensibles para el mismo	TESTING	15/12/2023	Equipo Testing	Desarrollador

Figura 12 Lluvia de Ideas

Para que la novedades e ideas sugeridas tomen curso resulta necesario transmitirlas a los desarrolladores para que hagan las correcciones pertinentes.

Luego se llevan a cabo pruebas de regresión, con el fin de garantizar que el software funcione después de implementar los cambios realizados por los desarrolladores, ya que la corrección de las novedades encontradas, podría generar un error en otro sector o modulo del producto.

Seguidamente y continuando con la ejecución del marco de trabajo en la fase Evaluación de Resultados, debemos comprobar la facilidad de uso y la experiencia del usuario al interactuar con el prototipo de software, por medio de un ejercicio de simulación en el entorno real del usuario. La técnica de Pruebas de Usabilidad [28] permite testear la funcionalidad y experiencia que viven los usuarios con las soluciones ideadas en la etapa anterior. Es una herramienta muy útil para probar y validar los requerimientos implementados.

Para lograr poner en práctica esta técnica, se les pidió a algunos usuarios que ejecuten transacciones normales con el prototipo de software, y luego se les hicieron preguntas concretas sobre la usabilidad de dicho software. Esta interacción permitió detectar la aceptación o rechazo del software por parte del usuario final.

La Figura 13 describe los problemas de usabilidad que experimentaron los usuarios. Antes de comenzar, se les explicó que este material se utilizaría para analizarlo y tomar notas, con el fin de mejorar el software. Se mencionó también que no existía inconveniente en caso de no poder realizar o completar una tarea.

PRUEBA DE USABILIDAD

Objetivo: Testear la Experiencia del Usuario sobre una posible solución

Proyecto:	Versión:
Equipo:	Fecha:
Observaciones:	

Prueba de Usabilidad			
Caso de Prueba a Evaluar	**Apunte del Caso de Prueba Generar Vuelo**	**Apunte del Caso de Prueba Reproducir Vuelo**	**Apunte del Caso de Prueba Generar Gráficos de Vuelo**
USUARIO 1	El usuario ingresó al sistema Debriefing y seleccionó la opción Generar Vuelo. Activó el calendario emergente, seleccionó una fecha, elegió entre los vuelos disponibles y lo cargó. 	El usuario ingresó al sistema y abrió el formulario de carga de vuelos para iniciar sesión de debreafing, seleccionó el archivo de vuelo generado, configuró los parametros deseados, seleccionó el vuelo a reproducir en 2D y ejecutó la reproducción. 	El usuario ingresó al sistema y ejecutó el generador de gráficos, seleccionó el archivo generado y la variable de interés a graficar. 
USUARIO 2	El usuario ingresó al sistema Debriefing y seleccionó la opción Generar Vuelo. Activó el calendario emergente, seleccionó una fecha, elegió entre los vuelos disponibles pero no pudo cargar el vuelo porque no encontró el botón. 	El usuario ingresó al sistema y abrió el formulario de carga de vuelos para iniciar sesión de debreafing, seleccionó el archivo de vuelo generado, configuró los parametros deseados, pero no pudo representar el vuelo ya que la imagen del icono a seleccionar es confusa. 	El usuario ingresó al sistema y ejecutó el generador de gráficos, seleccionó el archivo generado y la variable de interés a graficar. 

Figura 13 Prueba de Usabilidad

Luego de terminar con la prueba de usabilidad se procedió a interrogar al usuario para conocer qué tan fácil o difícil le resultó completar las tareas asignadas, además de agradecerle por participar. En tal sentido, se utilizó la técnica Evaluación de la Experiencia [28] con el objetivo de obtener información real de la experiencia de nuestros usuarios con respecto al producto software. Resultó útil poder recabar información sobre el nivel de satisfacción que ellos tienen con respecto al software testeado, clarificando si se cubrieron sus necesidades y si se ajusta a sus realidades. En la siguiente figura se muestra la técnica descripta:

EVALUACION DE L A EXPERIENCIA

Objetivo: Validar la experiencia del usuario con respecto al uso del software

USUARIO	*ASPECTOS A EVALUAR*	*VALORACIÓN*		
		SI	*ACEPTABLE*	*NO*
	1) ¿Considera que la información visualizada en pantalla de la trayectoria de vuelo en 3D y la cartografía es correcta y representativa?			
	2) ¿Los Datos de Altitud, Velocidad y Distancia entregados por el software son correctos y de utilidad?			
	3) Con respecto a la iconografía y cartografía, ¿se relacionan con la función que representa y se puede visualizar de manera efectiva?			
	4) ¿La búsqueda o selección de elementos se realiza de manera precisa?			
	5) ¿Considera de utilidad la Ayuda en pantalla?			
	6) ¿Los datos ingresados para configurar parámetros de vuelo son suficientes?			
	7) ¿Los Manuales de Usuario son de utilidad para operar el software?			
	8) La sincronización entre las diferentes interfaces (Generador de Vuelo, Debriefing, Generador de Gráficos), ¿funciona correctamente?			
	9) ¿La Visualización de la Representación en 3D es representativa?			
	10) ¿El tiempo de adaptación para operar el software es prudencial?			
	11) A la hora de evaluar un vuelo, ¿El software cumple con los Manuales de Procedimientos Aeronáuticos preestablecidos para tal fin?			
	12) ¿El Software fue estable durante su utilización?			
	13) El proceso de reinicio del software, ¿presenta dificultades al realizarlo?			
	14) El software, ¿cumple con sus expectativas?			
	15) ¿Considera Usted que el software satisface los objetivos requeridos para la simulación de vuelos?			
	16) ¿Recomendaría el software a sus colegas?			
	17) ¿Tiene alguna sugerencia de mejora? ¿Cuál?			
	Sugerencias			

Figura 14 Evaluación de la Experiencia

Rol del Usuario:

Fecha: **Firma y Aclaración**

Finalizadas las Pruebas de Usabilidad y Evaluación de la Experiencia, el último paso es considerar si el producto es apropiado o no para su uso, teniendo en cuenta las respuestas proporcionadas por ambas técnicas. Es necesario revisar los apuntes de cada tarea evaluada para luego preparar un informe que contenga los resultados de las pruebas, los problemas detectados, las recomendaciones y los resultados de la encuesta realizada a cada usuario. Toda esta información es crucial para determinar si el software cumple con las expectativas y si se ajusta a los requisitos definidos, determinando finalmente si el usuario final acepta el software o si requiere ajustes adicionales. Adicionalmente, las Pruebas de Aceptación ayudan a minimizar riesgos y a garantizar que el software se encuentre listo para su implementación en producción.
En el capítulo siguiente se detallarán con precisión las métricas utilizadas para analizar los resultados obtenidos.

La última fase en V&V es el Cierre del Proceso, donde se documenta y se prepara el informe final considerando los niveles de aceptación del software testeado. Las novedades expresadas en dicho informe son dadas a conocer al usuario final y se determina conjuntamente a los mismos, qué acciones se llevarán a cabo para llegar a la solución que se ajuste a sus necesidades y deseos. Es decir, en este punto se tomarán decisiones estratégicas de acuerdo al feedback recogido del usuario.
La técnica denominada Matriz de Feedback [28], representa en forma sistemática, ordenada y visual las impresiones que tienen los usuarios respecto a las novedades plasmadas en el informe final. En base a esta podemos visualizar información crucial que dará lugar a tres escenarios posibles:

- **Pasar el software a producción:** El feedback ha sido satisfactorio y el producto software pasa a producción para implementarlo.
- **Iterar:** el feedback muestra qué es aquello que valora el usuario del software prototipado y qué no. Por consiguiente, debemos elegir sobre qué parte volvemos a trabajar hasta una nueva implementación del proceso de V&V.

- **Abandonar el proceso:** por la cuestión que sea, no se continua con el proceso de V&V. Una razón puede ser que las validaciones con el usuario muestran un feedback muy negativo. Sea cual sea el motivo, el proceso se interrumpe.

A continuación, se muestra la Matriz de Feedback implementada, que permitió tomar decisiones respecto al rumbo a seguir en el proceso de V&V:

MATRIZ DE FEEDBACK

Objetivo: validar aspectos positivos y negativos del feedback con el usuario

Proyecto:	Versión:
Equipo:	Fecha:
Observaciones:	

Matriz de Feedback	
Novedades que agradan al usuario	**Novedades que no agradan al usuario**
• El software es amigable y fácil de usar. • La representación del vuelo en 2D y 3D coincide con el mundo real. • El software sigue un estándar consistente en todos sus módulos. • El software funciona correctamente en el entorno sugerido.	• La iconografía no visualiza una leyenda con la función que representa. • El acceso al software se realiza sin verificar que los usuarios estén autorizados a hacerlo.
Críticas constructivas que suman valor	**Nuevas ideas**
• El software podría alertar sobre posibles errores incluyendo mensajes de prevención.	• Desarrollar en el software un módulo de reportes personalizados, que permita visualizar información de los vuelos por cada usuario.

Figura 15 Matríz de Feedback

El feedback suministrado por la matriz revela que el software funciona correctamente y que las novedades encontradas no afectan a la seguridad en vuelo ni infiere a un aprendizaje negativo. Asimismo, las novedades detectadas en la ejecución de los casos de pruebas no impiden el correcto funcionamiento del software. Es una herramienta de gran utilidad para el adiestramiento a futuros Tripulantes de Aeronaves y los usuarios se encuentran conformes con el mismo.

Del mismo modo, los atributos de calidad correspondientes a Funcionalidad, Adecuación, Exactitud, Interoperabilidad, Usabilidad, Eficiencia y Portabilidad según norma ISO 9126, fueron evaluados satisfactoriamente, debiendo hacer ajustes en el atributo de Seguridad. A pesar de esto el software es aceptado por el usuario final y aprobado con observaciones, determinando su pase a producción.

V EVALUAR LOS RESULTADOS

Una vez realizado el pase a producción del software requerido por el usuario, el siguiente y último aspecto consiste en demostrar que las técnicas de Design Thinking se adecúan perfectamente al Ciclo de Vida de V&V dando lugar a la optimización del proceso. Otro factor radica en valorar el nivel de satisfacción de los usuarios, con la finalidad de conocer la apreciación final de los mismos respecto a la solución propuesta.

En consecuencia, y concluyendo con la última fase de la metodología de trabajo propuesta, este capítulo se enfoca en demostrar cómo la aplicación de Design Thinking permite obtener mejores resultados en el Ciclo de Vida de V&V. Además de medir el impacto que tiene el uso de la misma en tal proceso, para lo cual se tomarán las métricas establecidas en la norma ISO 9241-11.

5. EVALUACION DE RESULTADOS

5.1 RESULTADOS Y MEDIDAS

Utilizar la técnica de Design Thinking en el Ciclo de Vida de V&V en un proyecto de software operativo, favorece una mejora considerable en la identificación de problemas y necesidades de usuarios, recopilación de nuevas perspectivas de solución, generación de opciones de solución y prototipos que permiten representar ideas. [7]

La Figura 16 resume gráficamente como se aplican las técnicas de Design Thinking en las fases del Ciclo de Vida de V&V:

Fases del Ciclo de Vida de V&V	Etapas de Design Thinking	Técnicas de Design Thinking
Análisis de Requerimientos	Empatizar	• Mapa de Actores • Mapa de Empatía
	Definir	Los 5 Porqués
Planificación de Pruebas	Empatizar	Observación Encubierta
	Definir	Documento de Plan de Pruebas
Diseño y Desarrollo de Pruebas	Idear	Escenarios de Casos de Uso
	Prototipar	Casos de Pruebas
Ejecución de Pruebas	Idear	Lluvia de Ideas
Evaluación de resultados	Probar	• Pruebas de Usabilidad • Evaluación de la Experiencia • Pruebas de Aceptación
Cierre del Proceso	Probar	Matríz de Feedback

Figura 16 Integración del Design Thinking en el Ciclo de Vida de Verificación y Validación de Software

Asimismo, los resultados obtenidos en la aplicación de Design Thinking durante el Ciclo de Vida de V&V se detallan en la Figura 17, donde podemos observar la importancia de tener una comunicación constante con los usuarios para que el software de entrenamiento que ellos operan sea un producto usable y de calidad. [34]

Fases del Ciclo de Vida del V&V	Resultados obtenidos en su ejecución habitual	Resultados obtenidos con la aplicación de Design Thinking
Análisis de Requerimientos	Los requerimientos documentados son analizados, comprendidos y clasificados una sola vez, al comienzo del ciclo, y son utilizados a lo largo de todo el proceso sin involucrar a los usuarios. No se empatiza con los usuarios.	Los requerimientos son analizados y priorizados a través del trato directo con los usuarios. Se identifica a los usuarios intervinientes y se comprende lo que ellos esperan que haga el software.
Planificación de Pruebas	Los tipos, estrategias y técnicas de prueba son definidas en base a los requerimientos documentados.	Los tipos, estrategias y técnicas de prueba son identificadas y definidas con la participación de los usuarios.
Diseño y Desarrollo de Pruebas	Los Casos de Prueba son diseñados y desarrollados en base a requerimientos documentados.	Los Casos de Prueba son diseñados y desarrollados en base a escenarios.
Ejecución de Pruebas	Las desviaciones del comportamiento del software son documentadas y transmitidas sin tener en cuenta la opinión de los usuarios.	Las novedades que surgen al ejecutar casos de pruebas, son documentadas y validadas con el usuario en base a su criticidad, detallando ideas sugeridas para corregir tales desviaciones.
Evaluación de Resultados	Los requerimientos implementados son validados con el usuario final.	Los requerimientos implementados son validados con el usuario final.
Cierre del Proceso	Se reporta el fin del proceso sin intervención del usuario final.	Se reporta el fin del proceso con intervención del usuario final, determinando qué acciones se llevarán a cabo para llegar a la solución deseada.

Figura17 Resultados obtenidos con la aplicación de la técnica de Design Thinking

Con el propósito de medir el impacto que tiene el uso de *Design Thinking* en dicho proceso, se evalúa el atributo de usabilidad según la Norma ISO 9241-11, por ser este un

atributo primordial para la evaluación de la calidad. La misma define Usabilidad como el grado en que un producto puede ser utilizado por usuarios para lograr los objetivos específicos con efectividad, eficiencia y satisfacción en un contexto de uso. Tales métricas se detallan a continuación:

***Medida de Efectividad*:** evalúa que el usuario que realiza el ejercicio cumpla de forma correcta sus objetivos. Para cuantificar esta métrica se elaboró el siguiente indicador:

PTC = = ***Cantidad de tareas que realizó el usuario para cumplir el objetivo de una prueba *100***

Cantidad de tareas que debería realizar el usuario para cumplir el objetivo de una prueba

Si PTC>= 70%: Efectividad satisfactoria

Si PTC< 70%: Efectividad no satisfactoria

Medida de Eficiencia: evalúa que el usuario cumpla con el tiempo promedio estimado para realizar el ejercicio planteado. Para ello se implementó el siguiente indicador:

TR= Tiempo requerido por el usuario para completar las tareas en la ejecución de la prueba *100

Tiempo estándar de ejecución de la prueba

Si TR>= 70%: Eficiencia satisfactoria

Si TR< 70%: Eficiencia no satisfactoria

***Medida de la Satisfacción del Usuario*:** evalúa la satisfacción del usuario mediante observación directa y teniendo en cuenta la encuesta obtenida en la Evaluación de la Experiencia.

En nuestro caso real el Indicador de Efectividad PTR para un vuelo con dos aeronaves en forma simultánea arrojó un porcentaje del 90,9 %. Cabe aclarar que una de las tareas necesarias para cumplir con el objetivo de la prueba, no fue realizada por el usuario con

el fin de acortar tiempo. Pese a esto el índice de efectividad se cumple de manera satisfactoria. El Indicador de Eficiencia TR para el mismo vuelo determinó un valor del 88,13 %. Se considera un porcentaje dentro de los rangos previstos.

Las respuestas obtenidas en las encuestas, como así también gestos, expresiones faciales y actitudes frente al uso del software, difirió, entre un usuario y otro, en función de sus percepciones, experiencias y expectativas. Pese a ello, el nivel de satisfacción fue aceptable.

Cabe mentar, que el software utilizado como caso real, intenta replicar o simular la experiencia de pilotear una aeronave en particular, de la forma más realista posible. Para esto, se utilizan los parámetros típicos del avión en vuelo, como la velocidad y la altitud. Esto le permite al piloto formarse y al instructor, evaluarlo. En este contexto la aceptación del usuario no se refiere a la apreciación estética, sino más bien a la capacidad que tiene el software de ser lo más representativo de la realidad. El análisis de las encuestas realizadas reveló las siguientes novedades:

- La iconografía no visualiza una leyenda con la función que representa.
- Los procedimientos de resolución de emergencias no están incluidos.
- Se sugiere agregar información en tiempo real de altura y velocidad.
- Se recomienda que la aeronave acuse recibo cuando se superen las limitaciones de altura y velocidad.

La implementación de las métricas, permite cuantificar los resultados y obtener porcentajes correspondientes a la evaluación de la eficacia y la eficiencia. Estas métricas, junto a la satisfacción del usuario, resultan ser el complemento ideal para la evaluación de la usabilidad del software, logrando resultados que proporcionan un mayor nivel de confiabilidad, completitud y objetividad en las pruebas. [35]

Finalmente, se deduce que la aplicación del enfoque *Design Thinking* en el proceso de V&V mejora los resultados de las pruebas, debido a que los usuarios finales son nuestros actores más importantes y con los que se interactúa directamente.

Por consiguiente, la posibilidad de realizar un nuevo ciclo de V&V debido a que el primero no cubra las expectativas queda desestimada, debido a que en cada momento se cuenta con la participación de los usuarios.

De igual manera, se infiere que la importancia de combinar técnicas cuantitativas, como lo son la eficacia y la eficiencia, con otra cualitativa, como la satisfacción del usuario, en el proceso de pruebas de usabilidad, permiten medir cómo interactúan los usuarios con el software y qué tan fácil les resulta usarlo, lo cual nos posibilita obtener un producto de calidad. [35]

Así pues, se recomienda capacitar a los integrantes de cada proyecto sobre las herramientas que propone Design Thinking antes de empezar con el proceso de V&V. De tal forma que puedan tener un mejor entendimiento de la metodología, como así también posean un conocimiento adecuado de cuál herramienta utilizar para cada interacción con los usuarios finales.

VI CONCLUSIONES

Los resultados obtenidos en este Trabajo Final demuestran que las fases que intervienen en el proceso de Ciclo de Vida de V&V, pueden ser optimizadas mediante la aplicación de las Técnicas de Design Thinking. El marco de trabajo propuesto pretende ser la base para conseguir una mejora sustancial en el desarrollo de software correctamente verificado y validado. El mismo debe ser conocido y asimilado por todos los especialistas involucrados en un proyecto software, para impulsar el cambio hacia formas de trabajo más eficientes incluyendo al usuario como fuente estratégica.

El permitir que el equipo de testers conozca a los usuarios genera empatía y un mejor entendimiento de las necesidades reales y requerimientos que normalmente no se obtienen en la etapa de Análisis de Requerimientos.

Design Thinking constituye una propuesta que propicia la participación del usuario en todo el ciclo de vida de V&V. Su carácter iterativo permite adaptarse a los cambios, estableciendo un mecanismo que permite asegurar la calidad del software y mantener al usuario satisfecho.

VII REFERENCIAS

[1] Villón Moreno, M. S., & Zapata Balarezo, K. A. (2019). Plataforma tecnológica para contribuir a la planeación urbana en la ciudad de Guayaquil dirigido a la transportación enfocado al diseño de pruebas que permitan la verificación y validación del software usando herramientas de pruebas ágiles (agile testing Frameworks) (Bachelor's thesis, Universidad de Guayaquil Facultad de Ciencias Matemáticas y Físicas Carrera de Ingeniería en Sistemas Computacionales)

[2] https://www.atsistemas.com/es/blog/la-importancia-del-testing-de-software-y-de-la-automatizacin-de-pruebas

[3] Valdivia, J. J. G., Quentasi, S. M. Z., Yana, D. M. C., Apaza, R. E. C., & Vera, Y. P. (2020). Design Thinking en la Planificación de Pruebas de Software. *Innovación y Software*, *1*(2), 40-51.

[4] Osis, V. F. C., Soto, D. Q., Huarca, A. C., & Suyo, J. C. (2022). Casos de Estudio de Design Thinking en las etapas de Análisis y Diseño del Desarrollo de Software. *Innovación y Software*, *3*(1), 17-29.

[5] Espinoza Vásquez, J. C., & Espinoza Zapata, E. E. (2017). Marco de trabajo en base a Design Thinking y metodologías ágiles de desarrollo de software.

[6] Valdivia, J. J. G., Quentasi, S. M. Z., Yana, D. M. C., Apaza, R. E. C., & Vera, Y. P. (2020). Design Thinking en la Planificación de Pruebas de Software. *Innovación y Software*, *1*(2), 40-51.

[7] Zapana, G. I. H., Castro, N. H., Tico, M. A. S., Choquehuanca, E. D. C., & Bejarano, A. D. T. (2021). Aplicación del método Design Thinking en el área de requerimientos de software. *Innovación y Software*, *2*(1), 43-52.

[8] Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), Std 610-1990

[9] Carrizo, D., & Alfaro, A. (2018). Método de aseguramiento de la calidad en una metodología de desarrollo de software: un enfoque práctico. *Ingeniare. Revista chilena de ingeniería*, *26*(1), 114-129.

[10] Cristiá, M. (2009). Introducción al testing de software. *Recuperado el*, *14*.

[11] Software Testing: Conoce El Ciclo De Vida De Las Pruebas De Software (11 de enero de 2021) https://trans-ti.com/2021/01/11/software-testing-conoce-el-ciclo-de-vida-de-las-pruebas-de-software/

[12] Manual de Procedimientos de Análisis Operativo (Experimental) (2017), MPL2017

[13] Acosta, A. O., & TAPASCO, L. M. C. (2011). Impacto de las pruebas no funcionales en la medición de la calidad del producto software desarrollado (Doctoral dissertation, Universidad Tecnológica de Pereira. Facultad de Ingenierías Eléctrica, Electrónica, Física y Ciencias de la Computación. Ingeniería de Sistemas y Computación).

[14] GOMEZ FUENTES, M. D. C., CERVANTES OJEDA, JORGE, & GONZALEZ PEREZ, P. P. (2019). Fundamentos de ingeniería de software.

[15] Sastoque, S., Narváez, C., & Garnica, G. (2016). Metodología para la construcción de Interfaces Gráficas Centradas en el Usuario. Nuevas Ideas en Informática Educativa, 12, 314-324.

[16] Osis, V. F. C., Soto, D. Q., Huarca, A. C., & Suyo, J. C. (2022). Casos de Estudio de Design Thinking en las etapas de Análisis y Diseño del Desarrollo de Software. Innovación y Software, 3(1), 17-29.

[17] Resano, R. (2004). Design thinking. Universitat Oberta de Catalunya.

[18] International Organization for Standardization (ISO), Std 9241-11, 2018

[19] Mascheroni, M. A., Greiner, C. L., Petris, R. H., Dapozo, G. N., & Estayno, M. G. (2012). Calidad de software e ingeniería de usabilidad. In XIV Workshop de Investigadores en Ciencias de la Computación.

[20] Acosta, A. E. (2011). AgilUs: Construcción ágil de la Usabilidad. Citado na, 54.

[21] Espinoza Vásquez, J. C., & Espinoza Zapata, E. E. (2017). Marco de trabajo en base a Design Thinking y metodologías ágiles de desarrollo de software.

[22] Mejía, N. A., & Moreno, M. E. T. (2009). Técnicas de levantamiento de requerimientos con innovación. In Presentado en el Cuarto Congreso Colombiano de Computación 4CCC. Sociedad Colombiana de Computación S (CO) (Vol. 2).

[7] Zapana, G. I. H., Castro, N. H., Tico, M. A. S., Choquehuanca, E. D. C., & Bejarano, A. D. T. (2021). Aplicación del método Design Thinking en el área de requerimientos de software. Innovación y Software, 2(1), 43-52.

[23] ¿Qué son los casos de prueba? ¿Cómo escribir casos de prueba relacionados con el software? https://visuresolutions.com/es/what-are-test-cases-how-to-write-software-related-test-cases/

[24] González, J. F. P., Mayo, F. J. D., Rodríguez, J. J. G., & Cuaresma, M. J. E. (2014). Pruebas de aceptación orientadas al usuario: contexto ágil para un proyecto de gestión documental. Ibersid: revista de sistemas de información y documentación, 8, 73-80.

[25] OVIEDO, Juan Octavio, PRINCIPI, Ariel Cristian, RUMIE VITTAR, Juan Pablo, PÉREZ, Cristian Ricardo, SANTONATO, Luciano. (2021). Definición y Clasificación del Sistema de Debriefing para FAS D-AV 180

[26] ¿Cuáles son los pasos del análisis de actores? https://moqups.com/es/templates/business-strategy/stakeholder-mapping/

[27] Mapa de actores: La herramienta de diseño ideal para visualizar las relaciones entre stakeholders. https://designthinkingespaña.com/mapa-de-actores-diseno-de-servicios

[28] Técnicas de Innovación con Design Thinking. https://designthinking.es/tecnicas-de-innovacion/

[29] Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), Std 829 - 2008

[30] Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), Std 830 – 1998

[31] Merino García, L. (2021). Desarrollo de un sistema automático de validación y verificación de software.

[32] International Organization for Standardization (ISO), Std 9126 – 1992

[33] Campos Chiu, C. (2015). Las pruebas en el desarrollo de software.

[34] Filoniuk, V. R., Martínez, M. S., Diz, A. C., & Arias, S. E. (2023). Cómo integrar Design Thinking en el Ciclo de Vida de Verificación y Validación de Software para Simuladores de Vuelo. Innovación y Software, 4(2), 25-35.

[35] Martinez, M. S., Martínez, D. I., Filoniuk, V. R., Chiappori, G. G., Diz, A. C., & Arias, S. E. (2022). Aplicación de Norma ISO 9241-11 para la Evaluación de la Usabilidad en Simuladores de Vuelo. Innovación y Software, 3(2), 70-80.