Universidad Nacional Autónoma de México
Programa de Posgrado en Ciencias de la Administración

Propuesta de un ciclo de pruebas de software para empresas u organizaciones dedicadas al desarrollo de sistemas

T e s i s

Que para optar por el grado de:

Maestro en Informática Administrativa

Gestión de los servicios de tecnología de la información

Presenta:

Raúl Ramírez Urbano

Tutor:
Dra. Lucia Patricia Carrillo Velázquez
Facultad de Contaduría y Administración

Ciudad de México, mayo 2023

AGRADECIMIENTOS

A Dios por nunca dejarme solo y estar presente en cada etapa de mi vida. Gracias por todo lo que me has dado.

A mi madre por ser la mujer más increíble que he conocido. Sin tus exigencias nunca habría podido conocer de todo lo que era capaz. Gracias por siempre alentarme a no darme por vencido. El título de maestro es para ti.

A mi padre por siempre mostrarme el valor del trabajo duro, y la visión a futuro que se debe de tener en ciertos aspectos de la vida. También, por los buenos chistes. Gracias por siempre dar lo mejor de ti.

A mi hermano por los buenos, malos, y chistosos momentos que hemos vivido. Sin duda, siempre nos vamos a reír de todo.

A mis verdaderos amigos por los buenos momentos, la compañía, las anécdotas, y sobre todo las risas. Su amistad significa mucho.

A mi amada universidad por darme la oportunidad de estudiar una licenciatura y un posgrado.

A mi tutora por sus enseñanzas y acompañamiento en este proceso. Su presencia ayudó mucho a darle vida a esta investigación.

A mis profesores de licenciatura y de posgrado por su entrega y responsabilidad en mis años de estudio.

A mi canario. Por ser la mejor compañía para escuchar música.

CONTENIDO

INTRODUCCIÓN

La industria del software se ha transformado en un negocio dominante en las economías del mundo, ya que permite ser integrado a casi todos los aspectos de la vida; sin embargo, hay muchas limitantes para tener un sistema de calidad, por ejemplo: tiempos cortos de desarrollo, sobrecarga de trabajo, omisión de una etapa de pruebas antes de liberar el sistema, etc. En este sentido, el software que no funciona correctamente, que no cumple con la calidad esperada por el cliente, puede ocasionar muchos problemas, incluida la pérdida de dinero, mala reputación, e incluso lesiones o la muerte a los usuarios finales (En el caso de sistemas de transporte o de salud). Como respuesta a esta necesidad dentro del ciclo de vida de desarrollo, se encuentran metodologías, Normas ISO, estándares, y certificaciones. Las cuales, tienen como propósito brindar un panorama técnico acerca de las pruebas de software. También, es importante mencionar que existen investigaciones que de acuerdo con un contexto específico buscan la formalización de las pruebas como un medio de calidad.

Es importante mencionar que a pesar de las ventajas que ofrecen la ejecución de pruebas muchas empresas no las realizan debido a los cortos periodos de tiempo de desarrollo, y en su defecto también al poco conocimiento teórico y práctico que tienen de las mismas. Con relación a lo anterior, nace la generación de la siguiente investigación: “Propuesta de un ciclo de pruebas de software para empresas u organizaciones dedicadas al desarrollo de sistemas”. La cual, tiene como finalidad, formalizar un proceso de pruebas basado en un enfoque teórico y práctico que pueda ser estudiado por las empresas que necesitan aumentar la calidad de sus sistemas, mediante las pruebas de software.

RESUMEN CAPITULAR

Capítulo 1: Problema real

Para comenzar a conocer el problema real de esta investigación este capítulo comienza describiendo la integración del software en las actividades diarias del ser humano, y como la falta de calidad en los sistemas puede originar diversos problemas, entre ellos: pérdidas económicas, mala reputación para los desarrolladores y a la organización que representan. Así mismo, para ejemplificar lo anterior, se describen tres ejemplos de empresas u organizaciones que tuvieron repercusiones económicas, y sociales, debido a la falta de calidad en sus proyectos.

Capítulo 2: Antecedentes

Con la finalidad de conocer los antecedentes de la calidad en sistemas de información se hace mención del origen de la ingeniería de software, y de algunos modelos tradiciones y ágiles, que en sus metodologías mencionan el proceso de pruebas de software como una medida para generar calidad en los proyectos de desarrollo. De la misma manera, referente a temas especializados, se describen los estándares y organizaciones que hacen referencias a las pruebas de software.

Capítulo 3: Estado del arte

En este capítulo, se hará referencia a algunos proyectos de investigación que han sido propuestos y se relacionan con la temática: “Pruebas de software”. Tema de esta tesis. Con la finalidad de realizar un análisis que permita encontrar dentro de sus aportaciones puntos que puedan ser mejorados a través de esta investigación.

Capítulo 4: Marco de investigación

Con relación al marco de investigación. Este capítulo, inicia con el problema de investigación. El cual, se basa en la carencia de un proceso de pruebas de software basado en un enfoque teórico y práctico. Así mismo, dentro de la justificación se explica que la razón de este estudio tiene como finalidad aportar una propuesta técnica que permita atender las debilidades encontradas en la literatura relacionada a las pruebas de software.

Como objetivo se planea la generación de una propuesta técnica acerca de un ciclo de pruebas de software. Que permita establecer un enfoque teórico, etapas, y actividades para su gestión.

De acuerdo con lo anterior, en esta sección también se formula la pregunta de investigación, la cual en esencia busca responder si la falta de atención hacia las pruebas de software ¿Está relacionada al desconocimiento del enfoque teórico y práctico de las mismas?

Por último, se concibe la hipótesis de la investigación, la cual busca responder si la fundamentación, enseñanza, y el establecimiento de una metodología relacionada a las pruebas de software, permitirá su adopción dentro de las áreas de desarrollo, transformándose en una etapa primordial para el desarrollo de sistemas.

Capítulo 5: Metodología de la investigación

En esta fase se plantea la metodología de la investigación. En donde se describe que la presente investigación es documental, la cual (Rocha, 2016): Consiste en analizar diferentes recursos de información. Con el propósito de proponer o mejorar una determinada investigación. En este caso, el proceso de pruebas de software.

Capítulo 6: Marco de referencia

En este capítulo se muestra el enfoque teórico, práctico y metodológico que conforma la presente investigación. De la misma manera, se presenta el marco de referencia para cada enfoque. Esto, con la finalidad de conocer la procedencia de la información.

Capítulo 7: Propuesta de un ciclo de pruebas de software para empresas u organizaciones dedicadas al desarrollo de sistemas

Con la finalidad de dar a conocer los apartados que componen esta propuesta. En este capítulo se explican los apartados del del enfoque teórico, práctico y metodológico. Así mismo, se exponen las cinco fases de la metodología que se diseñó. De tal manera que su comprensión, puede ser entendida de manera sencilla y práctica.

Capítulo 8: Caso práctico

En este capítulo, se expone la aplicación de la presente propuesta. La cual, se llevó a cabo dentro de una pequeña oficina de servicios de cómputo ubicada en la Ciudad de México (CDMX). Esta, pertenece al sector público y por motivos de privacidad y de confidencialidad será omitida.

Actualmente, este departamento cuenta con múltiples servicios. Entre ellos destacan los siguientes: programa de formación de becarios, programas de servicio social, cursos en línea, salas de cómputo, y desarrollo de sistemas internos. Al tener diversas actividades que involucran la utilización de equipo de cómputo. Se han visto en la necesidad de desarrollar una aplicación web que permita gestionar los bienes de la organización. De esta manera, la metodología desarrollada, será puesta en marcha en este capítulo, permitiendo conocer el estado de calidad del sistema desarrollado.

CAPÍTULO 1: PROBLEMA REAL

El desarrollo de software actualmente es una de las actividades más importantes en el mundo ya que permite ser integrada a casi todos los aspectos de la vida. Ha invadido: la economía, la comunicación, el transporte, la salud, etc. (Mera Paz, 2016).

Actualmente, existen muchas maneras de probar software. Todo mundo necesita software autosuficiente; sin embargo, no todos realizan las pruebas pertinentes y se preocupan por la calidad de los sistemas de información (Pressman, 2010).

Pressman, menciona que la calidad de un sistema de información se mide y se observa, a través de las siguientes propiedades: Usabilidad, funcionalidad, confiabilidad, eficiencia y susceptibilidad (Pressman, 2010).

A continuación, se describe cada una de ellas:

- Usabilidad: Relacionada a la comprensión general de un sitio (Pressman, 2010).
- Funcionalidad: Comprende elementos de operatividad (Pressman, 2010).
- Confiabilidad: Ligada con el adecuado funcionamiento de los elementos pensados para el usuario (Pressman, 2010).
- Eficiencia: Corresponde al desempeño y al tiempo de respuesta por parte de un sitio (Pressman, 2010).
- Susceptibilidad (Facilidad de mantenimiento): Conlleva la habilidad de corrección y adaptabilidad (Pressman, 2010).

Las metodologías del ciclo de vida de desarrollo de software ágil y tradicional en sus respectivos modelos integran una etapa de pruebas que tiene por objetivo verificar que el sistema cumpla con diversos criterios de calidad antes de su liberación y entrega final al

cliente (Pressman, 2010); sin embargo, al omitir esta fase o al realizar de manera deficiente, se pueden originar diversos problemas, entre ellos:

- Pérdidas económicas: Un defecto encontrado en etapas iniciales cuesta menos que un defecto encontrado en la etapa de mantenimiento (Pressman, 2010).
- Inversión de tiempo e incremento de recursos humanos para solucionar los desperfectos encontrados por el usuario: Al tener defectos de este tipo se debe de contar con un equipo de trabajo ágil y eficiente que prevenga los efectos colaterales del defecto encontrado y generé las pruebas necesarias para que evaluar que este ya no se presenta (Pressman, 2010).
- Mala reputación para los desarrolladores y a la organización que representan: Las opiniones, y comentarios negativos. Además, la falta de contratos es un elemento difícil de cuantificar cuando se produce software de mala calidad (Pressman, 2010).

EJEMPLOS DE SISTEMAS INFORMÁTICOS QUE HAN GENERADO UN IMPACTO SOCIAL O ECONÓMICO DEBIDO A SU MAL FUNCIONAMIENTO

Con la finalidad de ejemplificar las consecuencias de la omisión o poca importancia de la etapa de pruebas, tema central de esta investigación. A continuación, se presentan tres casos que ayudarán a comprender la relevancia de generar una etapa de pruebas de software en los sistemas de información y los efectos que está puede generar al no implementarse correctamente.

CASO 1: FALLA EN LOS MISILES PATRIOT

La guerra del golfo fue una disputa que comenzó el 2 de agosto de 1990 y culminó el 28 de febrero de 1991. En ella se encontraban 34 países de las Naciones Unidas liderados por Estados Unidos contra la República de Irak (García, 2020).

Como armamento encontrado en este acontecimiento, se identificaron los misiles Scud. Los cuales, eran misiles balísticos utilizados por la República de Irak (Artehistoria, s. f.). Por otra parte, Estados Unidos contaba con misiles Patriot, que son de carácter defensivo, y se encargaban de destruir los misiles que llegaban por parte de Irak. Estos misiles, operaban a través de un sistema móvil de defensa que elimina simultáneamente múltiples objetivos bajo un ambiente de medidas establecidas por el usuario (Ministerio De Defensa Del Gobierno de España, s. f.).

La misión de los misiles Patriot, era predecir la trayectoria del misil enemigo (SCUD) y destruirlo, esto a través de dos variables: la velocidad y el tiempo. El sistema de defensa primero expresaba el tiempo de lanzamiento en décimas de segundos. Posteriormente, transformaba el tiempo en números binarios. Una vez hecha la primera conversión, el sistema hacía otra transformación en décimas de segundos, que se acercará al primer número binario establecido, y con ello, procedía a lanzar el misil que se aproximaba al campo enemigo, pero no tomaron en cuenta que el sistema acumulaba un tiempo equivalente a 0,000000095 segundos por misil lanzado (García, 2020).

El problema real se presentó cuando el sistema llevaba 100 horas funcionando. Con esta sobrecarga de trabajo, se acumularon los segundos por misil lanzado. El sistema entendió que era una falsa alarma tener tanto tiempo almacenado para el lanzamiento de los misiles.

Además, esto se refutó al hacer la primera conversión a binarios. El número binario sobrepasaba el tamaño permitido al igual que el tiempo, lo que originó que los misiles Patriot no despegaran y los misiles Scud se estrellaran contra el campamento militar, dejando a 28 soldados muertos y lastimando a 100 personas en el campo de batalla (García, 2020).

CASO 2: ERROR MATEMÁTICO EN EL PROCESADOR PENTIUM DE INTEL

En octubre de 1994 se presentaba al mundo el procesador Pentium de Intel, en esos momentos era el chip más avanzado de su época; a excepción de un ligero error encontrado por los usuarios: presentaba una falla en las operaciones matemáticas, en específico, en las divisiones. Este error le costó a la compañía 475 millones de dólares y generó que en diciembre de ese mismo año Intel se viera afectada en la Bolsa de Valores de Nueva York, debido a que IBM detuvo la venta de sus computadoras que incluían ese procesador. (Jiménez, 1995).

Como medida de apoyo al usuario. Intel abrió 40 líneas de atención al cliente, siguiendo la política de reemplazar los Pentium defectuosos de manera gratuita. Esto, con la finalidad de conservar la vida del chip; sin embargo, su reputación ya se había visto dañada por la mayoría de los consumidores (Jiménez, 1995).

CASO 3: EXPLOSIÓN DEL COHETE ARIANE 5

En 1996, “El Ariane 5” era un proyectil novedoso, diseñado para instalar satélites en la órbita espacial; dicho proyecto se elaboró a lo largo de 10 años y costó 6 mil millones de euros. Era el primer cohete fruto del más ambicioso proyecto de la industria espacial europea, pero, debido a un error de software en el programa de control guiado, este, estalló en vuelo sobre La Guyana Francesa el 4 junio de 1996 (González, 1996).

Cabe mencionar que antes del Ariane 5 existía el Ariane 4, este último fue un éxito en todos los aspectos, por lo que en ese momento se decidió utilizar su sistema de control guiado en el Ariane 5; excepto que los desarrolladores omitieron que, al ser un cohete más potente, el software para el sistema de control tenía que ser igual. En consecuencia, el sistema falló por exceso de información. Con ello, generó datos erróneos a la computadora central, que a su vez estableció ubicaciones imprecisas y colocó al cohete en una trayectoria imposible. El resultado fue que el cohete se autodestruyó a 3.400 metros de altura sobre la base espacial de Kourou, dejando a un lado la inversión hecha por la industria espacial europea (González, 1996).

ANÁLISIS

La falta de calidad en los sistemas de información es una problemática que se debe de tomar con mayor relevancia dentro de las organizaciones dedicadas al desarrollo.

Aplicar un proceso de pruebas de software correctamente que permita conocer el estado de calidad en este tipo de productos, es algo seriamente a considerar. Es fundamental mencionar que dependiendo del tipo de producto o servicio que se brinde este proceso deberá de ser tan riguroso o breve como sea su desempeño en la vida diaria.

CAPÍTULO 2: ANTECEDENTES

Para entender cómo se ha ido dando forma al concepto de calidad en programas de software. Dentro de este capítulo, se mencionan los orígenes de la Ingeniería de Software, y cómo a raíz de ella se han generado diversas metodologías tradicionales, y ágiles que señalan dentro de sus procesos la etapa de pruebas como componente de calidad. Además, se mencionan los organismos que han promovido el conocimiento y uso de la fase de pruebas de software,

entre ellos se encuentran: El Organismo Internacional de Normalización (ISO), La Comisión Electrotécnica Internacional (IEC), El Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), y La Junta Internacional de Calificaciones de Pruebas de Software (ISTQB). Estos, se enfocan a enseñar en qué consisten las pruebas de software y dar una visión específica de este tema. Por lo mismo, también se describen dentro de este apartado debido a su importancia en el tema de esta investigación.

INGENIERÍA DE SOFTWARE

El origen de la Ingeniería de Software comienza después de la Segunda Guerra Mundial, los importantes avances tecnológicos tuvieron lugar en Estados Unidos de América (EE. UU.) y se generaron en el ámbito de la industria militar. Con los avances de esa época y la aparición de lenguajes de alto nivel (Lenguajes de computadora cercanos al lenguaje humano), se empezaron a crear sistemas más avanzados (Pantaleo & Rinaudo, 2015).

En la década de 1960, ocurrió un suceso que estableció un antes y un después dentro del contexto histórico del desarrollo de software. Se creó el sistema operativo IBM OS/360. Este fue uno de los mayores proyectos de la época, y generó un llamado a crear una disciplina que se encargara de gestionar el desarrollo de software (Pantaleo & Rinaudo, 2015).

En 1968 se empieza a dar forma al nombre de Ingeniería de Software, a través del Comité de Tecnología de la Organización del Tratado del Atlántico Norte (OTAN). En dicha conferencia se comienza a utilizar la expresión “Ingeniería de Software”, para referirse al desarrollo de sistemas y, por ende, hacia el año 1980, se concluye que es necesaria la creación de una disciplina, que se ocupe sistemáticamente del diseño y la construcción de productos de software llamada formalmente: “Ingeniería de Software”, generando una representación

de los pasos que se deben de seguir para generar sistemas de información (Pantaleo & Rinaudo, 2015).

En 1984 se crea el Instituto de Ingeniería del Software (SEI) como centro de investigación y desarrollo del gobierno federal de los Estados Unidos de América con el objetivo de promover el crecimiento de la Ingeniería de Software en sus diferentes aspectos relacionados a los sistemas de información (Pantaleo & Rinaudo, 2015).

En 1990 El Instituto de Ingenieros Eléctricos y Electrónicos (IEE) hace público el estándar 610.12 que establece los conceptos relacionados a la Ingeniería del Software. Desde entonces, esta disciplina se estudia en diferentes universidades del mundo presentando sus conceptos básicos y los temas relacionados a ella (Pantaleo & Rinaudo, 2015).

MODELOS TRADICIONES

Los modelos de proceso prescriptivo o modelos tradicionales han generado una estructura de trabajo sobre el desarrollo de software haciendo atención en la planificación y control del proyecto, así como en acciones y tareas de ingeniería de software (Pressman, 2010).

Con la finalidad de profundizar en este tipo de modelos, se mencionan tres principales metodologías que en su estructura mencionan la etapa de pruebas de software como elemento de calidad en los sistemas de información y buscan atender el problema real de esta investigación.

MODELO EN CASCADA

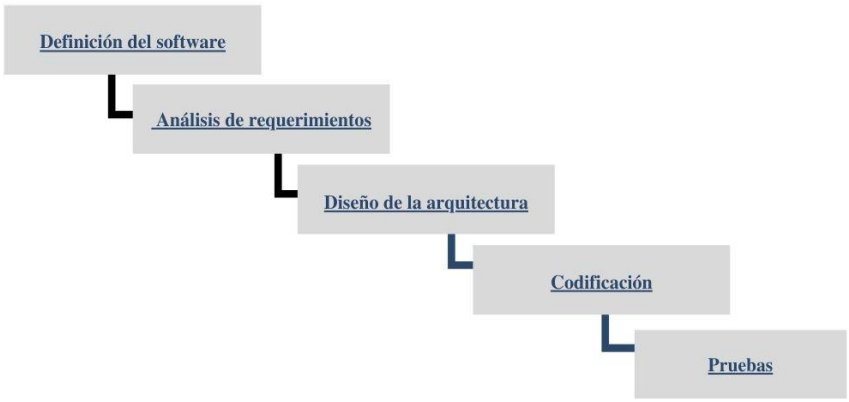
Es un modelo de desarrollo secuencial. En este, el proyecto está dividido en fases que deben de realizarse de manera secuencial. Este proceso se compone de seis etapas. Las cuales, se mencionan a continuación (Pantaleo & Rinaudo, 2015):

1. Definición del software: Comprende la visión del producto desde un enfoque comercial (Pantaleo & Rinaudo, 2015).
2. Análisis de requerimientos: Implica la comprensión del software a desarrollar con la finalidad de comprender la viabilidad del proyecto (Pantaleo & Rinaudo, 2015).
3. Diseño: Atiende una idea o esbozo del sistema de información. (Pantaleo & Rinaudo, 2015).
4. Codificación: Corresponde a la actividad de codificar el producto con base en el diseño de la arquitectura (Pantaleo & Rinaudo, 2015).
5. Pruebas: En esta etapa se verifica que el producto realizado satisfaga los requerimientos establecidos en la etapa de análisis. Además de comprobar que el sistema cumpla con el comportamiento esperado (Pantaleo & Rinaudo, 2015).

Como podemos ver en la Figura 1. Las etapas del modelo en cascada se encuentran escalonadas. De tal manera que el inicio de una depende de la finalización de la anterior. Dejando hasta el final la etapa de pruebas de software.

Figura 1

Modelo en cascada



MODELO DEL PROCESO INCREMENTAL

El modelo incremental se compone de avances paralelos, generando adelantos completos en cada una de sus fases. Cada etapa de esta metodología, se mencionan a continuación (Pressman, 2010):

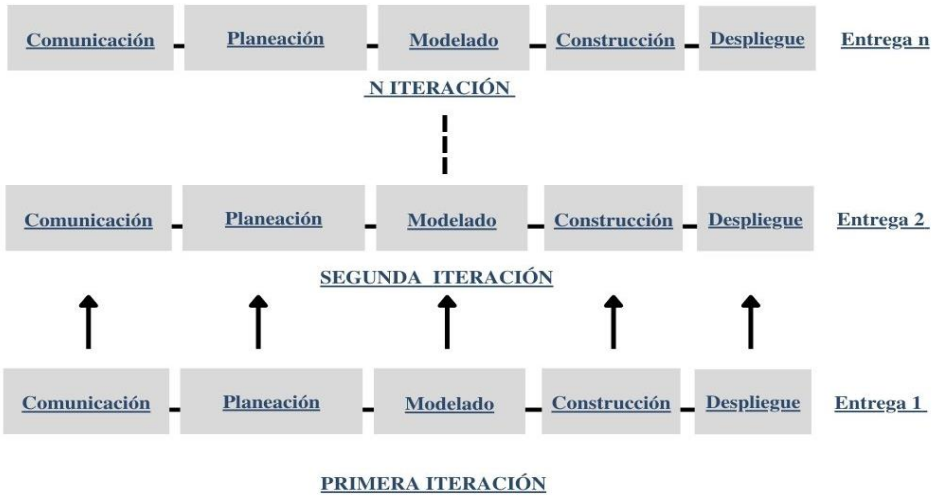
1. Comunicación: Establece la relación contractual entre la organización desarrolladora y los posibles clientes (Pressman, 2010).
2. Planeación: Conlleva la organización del proyecto (Pressman, 2010).
3. Modelado: Enfoca las actividades de análisis y diseño (Pressman, 2010).
4. Construcción: Encaminada a la generación de código y pruebas (Pressman, 2010).
5. Despliegue: Realiza la entrega y retroalimentación del producto (Pressman, 2010).

Como podemos distinguir en la Figura 2. Las etapas del modelo incremental se realizan de manera paralela. Haciendo una entrega una vez que finalizan todas las etapas, consiguiendo

un avance simultáneo en la realización del proyecto. Involucrando la etapa de pruebas, en la fase de construcción.

Figura 2

Modelo del proceso incremental



MODELO EN ESPIRAL

Es una metodología que se adapta con la idea de generar prototipos de manera secuencial. Las fases de este modelo se trabajan en una espiral, forma que representa un avance y el cumplimiento de las fases del proyecto. Estas, se mencionan a continuación (Pressman, 2010):

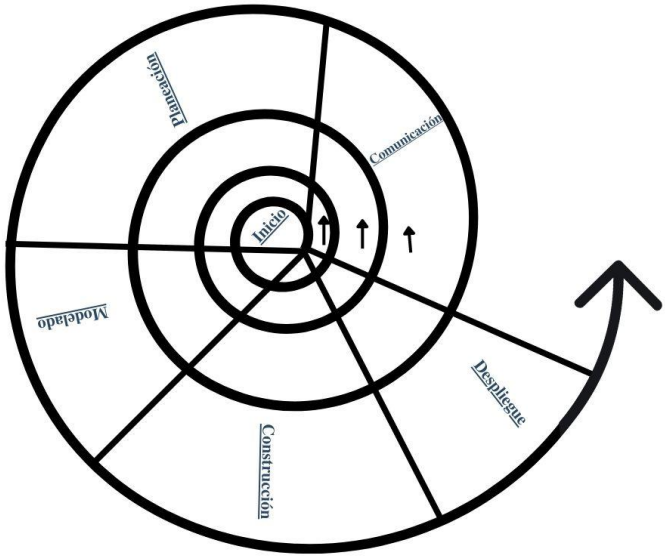
1. Comunicación: Establece la relación contractual entre la organización desarrolladora y los posibles clientes (Pressman, 2010).
2. Planeación: Contempla la estimación, programación y análisis de riesgo del proyecto (Pressman, 2010).

3. Modelado: Encamina las actividades de análisis y diseño (Pressman, 2010).
4. Construcción: Focalizada en la generación de código y pruebas (Pressman, 2010).
5. Despliegue: Realiza la entrega y retroalimentación del producto (Pressman, 2010).

Como podemos percibir en la Figura 3. Las etapas del modelo en espiral se realizan de manera individual. Finalizada la primera fase, se comienza con la segunda. Una vez completado el espiral se cubre un adelanto del proyecto. Conforme este siga su progreso, el espiral irá creciendo, agregando más tareas a las etapas del proyecto. Implicando la etapa de pruebas, en la fase de construcción.

Figura 3

Modelo del proceso en espiral



MODELOS ÁGILES

A principios de 1990, a medida que la informática comenzó a crecer, el desarrollo de software encontró una inestabilidad. Conocida como: "La crisis del desarrollo de aplicaciones". Esta,

hacía referencia a los problemas que existían al validar un requerimiento y su implementación. El tiempo entre las actividades anteriormente mencionadas eran de 3 años, lo cual significaba mucho tiempo (Roche, 2018).

El 11 de febrero de 2001 en Utah un grupo de 17 personas crearon el manifiesto para el desarrollo ágil. El cual, es un documento que se compone de 4 valores enfocados al cambio de mentalidad y a una nueva cultura organizativa (Ramírez G., 2020). Los valores se describen a continuación:

1. Individuos e interacciones sobre procesos y herramientas (Ramírez G., 2020).
2. Software funcionando sobre documentación exhaustiva (Ramírez G., 2020).
3. Colaboración con el cliente sobre negociación contractual (Ramírez G., 2020).
4. Respuesta ante el cambio seguir un plan (Ramírez G., 2020).

Dichos valores se concentran en los siguientes 12 principios:

1. Satisfacción del cliente a través de la entrega constante y anticipada de ‘software’ que permita generar valor al proyecto (Ramírez G., 2020).
2. Requisitos cambiantes, incluso en etapas tardías del desarrollo. Esto con la finalidad de proporcionar ventaja competitiva al cliente (Ramírez G., 2020).
3. Entrega de un software operativo preferentemente en un periodo corto de tiempo (Ramírez G., 2020).
4. Los responsables de negocio y los desarrolladores interactúan de forma constante durante todo el proyecto (Ramírez G., 2020).
5. Los proyectos se generan a través de personas motivadas. (Ramírez G., 2020).

6. Comunicación cara a cara con el propósito de tener una comunicación más efectiva entre el equipo de desarrollo, y los miembros del equipo (Ramírez G., 2020).
7. Software funcional como una medida principal de progreso (Ramírez G., 2020).
8. Promover el desarrollo sostenible. Todos los involucrados en el proyecto deben de colaborar en un ritmo constante de forma indefinida (Ramírez G., 2020).
9. La atención continua mejora la agilidad (Ramírez G., 2020).
10. Notar la cantidad de tareas no realizadas, es fundamental para el desarrollo ágil (Ramírez G., 2020).
11. Los equipos auto-organizados promueven mejores arquitecturas, requisitos y diseños (Ramírez G., 2020).
12. En periodos cortos de tiempo el equipo analiza cómo ser más ágil (Ramírez G., 2020).

En conclusión. Las metodologías ágiles buscan brindar flexibilidad y libertad durante el proceso de desarrollo de software (Pantaleo & Rinaudo, 2015).

Con la finalidad de profundizar en este tipo de modelos, se mencionan tres principales metodologías que en su estructura mencionan la etapa de pruebas como elemento de calidad en los sistemas de información y buscan atender el problema real de esta investigación.

PROGRAMACIÓN EXTREMA (XP)

La programación extrema se desempeña a través de un conjunto de normas que ocurren a través de cuatro actividades: planeación, diseño, codificación y pruebas (Pressman, 2010).

1. Planeación: Engloba la captación de requerimientos con la finalidad de entender el contexto del producto y la funcionalidad esperada. Dentro de esta etapa se generan

las historias de usuario. Las cuales, son redactadas por los clientes y evaluadas por el equipo de desarrollo (Pressman, 2010).

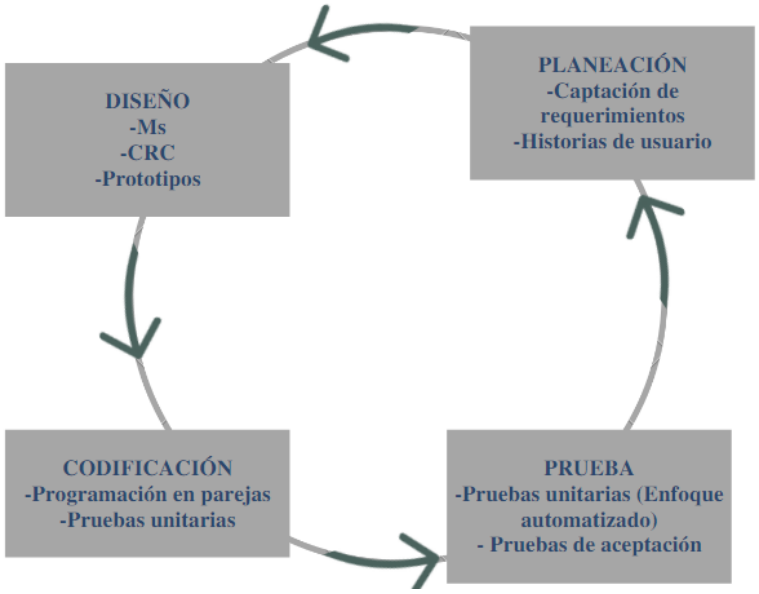
2. Diseño: Sigue el principio MS (Mantenlo sencillo). Durante esta fase se toman como base las historias de usuario y las tarjetas CRC (Clase-Responsabilidad-Colaborador). Estas, permiten hacer una proyección sobre lo que se va a diseñar. Además, para diseños complejos se desarrollan prototipos. Su objetivo es evaluar el diseño con el cliente de manera oportuna y ágil (Pressman, 2010).
3. Codificación: Después de haber trabajado en el diseño preliminar, se genera el código del producto final. El cual, es desarrollado bajo el esquema: "Programación en parejas". Este, promueve que dos programadores colaboren en un mismo espacio de trabajo con el propósito de generar código para un mismo requerimiento. Asimismo, dentro de esta etapa se generan pruebas unitarias para validar la funcionalidad de cada historia de usuario desarrollada (Pressman, 2010). Una prueba unitaria consiste en aislar una parte del código y comprobar que funciona de acuerdo con las especificaciones del cliente (Pressman, 2010).
4. Pruebas: Durante esta fase se realizan pruebas unitarias iguales a las de la etapa anterior; sin embargo, están enfocadas a implementarse bajo una estructura automatizada. Es decir, deben de ser ejecutadas con facilidad y en repetidas veces. También, se generan pruebas de aceptación con el usuario. Estas, se enfocan en la interpretación que tiene el cliente con el sistema, las cuales parten de la etapa de planeación (Pressman, 2010).

Como podemos percibir en la Figura 4. Las etapas del modelo Programación Extrema (XP) tienen como propósito involucrar y hacer entregables de manera oportuna al cliente.

Por lo mismo, las fases de este modelo se colocan dentro de un círculo. El cual, representa la agilidad de este modelo, dejando la etapa de pruebas como una etapa última fase.

Figura 4

Modelo de Programación Extrema



SCRUM

Scrum se basa en el manifiesto ágil y es utilizado para administrar labores dentro de los equipos de desarrollo. Entre sus elementos principales tenemos los siguientes (Pressman, 2010):

Retraso: Comprende la lista de prioridades que tiene el equipo de desarrollo con el cliente. Estas, están enfocadas a las características del producto que se requieren con urgencia por parte del consumidor final (Pressman, 2010).

Sprint: Consisten en lapsos de tiempo, comúnmente con duración de 30 días, y abarcan unidades de trabajo de la lista de prioridades y sus respectivas pruebas, tanto manuales como automatizadas. Una vez formalizado el sprint este no puede modificar debido a que ya se han establecido las actividades por realizar (Pressman, 2010).

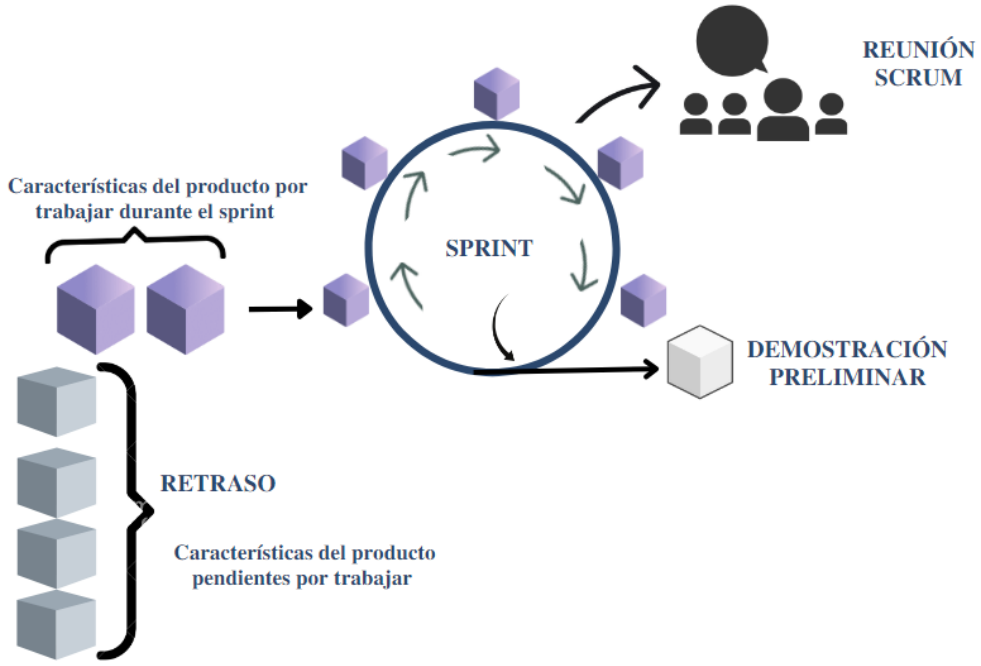
Reuniones scrum: Son juntas de corto tiempo, generalmente de 15 minutos. En donde, cada integrante del equipo comenta su última actividad realizada, los obstáculos que ha tenido para poder realizarla y sus planes en lo que llega la siguiente reunión (Pressman, 2010).

Demostraciones preliminares: Focalizada a realizar entregas que presente un incremento en el software desarrollado. De tal manera, que el avance de la funcionalidad que se haya realizado durante el sprint pueda ser demostrada y evaluada por el cliente (Pressman, 2010).

Como podemos percibir en la Figura 5. Se observan los procesos principales que comprenden el modelo Scrum y su interacción entre cada uno de ellos. Incorporando la etapa de pruebas dentro del sprint como un proceso previo a la demostración preliminar. (Pressman, 2010).

Figura 5

Modelo Scrum



CRYSTAL

Dentro de este modelo hay una metodología para cada proyecto. Estás, se identifican por colores, de la misma manera que sucede con los cristales. El nivel de opacidad del color indica un mayor número de personas implicadas en el desarrollo, lo que refleja el tamaño del proyecto y, por lo tanto, la necesidad de mayor control en el proceso (Cadavid, 2013).

Las metodologías de este modelo se mencionan a continuación:

1. Metodología limpia: Utilizada para la generación de un solo equipo de hasta 6 personas o menos. Utilizada normalmente cuando se tiene una documentación sólida

del proyecto a desarrollar y un solo equipo puede generar y probar el producto esperado (Virender, 2021).

2. Metodología amarilla: Contempla entre 7 y 20 personas en diferentes equipos. Seleccionada cuando se tienen identificados a los integrantes del equipo de desarrollo. La elaboración de código se da a través de los usuarios finales. Reduciendo la necesidad de demasiada documentación. Por lo tanto, se vuelve fácil para el desarrollador comprender su trabajo. La generación de pruebas automatizadas es de suma importancia dentro de esta metodología ya que se utilizan para resolver los errores más rápido (Virender, 2021).
3. Metodología naranja: Estipulada para equipos de entre 21 y 40 personas. En esta metodología se divide a los equipos según sus habilidades (Análisis, diseño, desarrollo, y pruebas), generando un desarrollo incremental y una liberación cada 3 o 4 meses (Virender, 2021).
4. Metodología roja: Concretamente para equipos de entre 41 y 80 personas. Los equipos se forman y dividen según el trabajo requerido, realizando diferentes funciones (Análisis, diseño, desarrollo, y pruebas) para la liberación del proyecto (Virender, 2021).

Como podemos percibir en la Figura 6. Se observan las cuatro metodologías que se aplican para este modelo. Cada una corresponde al número de individuos que se necesitan para llevar a cabo el desarrollo de software, contemplando de diferente manera la etapa de pruebas de acuerdo con el recurso humano disponible.

Figura 6

Modelo Crystal

NORMAS ISO

El Organismo Internacional de Normalización (ISO), es una asociación que trabaja en formar un modelo de calidad, con la finalidad de garantizar la satisfacción y expectativas de los clientes. Nació en 1947 y tiene presencia en 91 estados asociados (Isotools, 2022). A inicios del año 1980, la ISO nombró juntas técnicas para que colaboraran en la generación de reglas habituales que fuesen aprobadas universalmente (Isotools , 2022).

El desarrollo de las reglas ISO ha sido relevante porque ha permitido diversificarse en diferentes ramas, las cuales tratan diferentes aspectos, como las pruebas de software. Actualmente estas normas están presentes en muchas organizaciones y generan valor agregado a las instituciones que cumplen con ellas (Isotools , 2022).

IEC

La Comisión Electrotécnica Internacional (IEC), es una organización mundial de miembros sin fines de lucro, cuyo trabajo sustenta la infraestructura de calidad y el comercio internacional de productos eléctricos y electrónicos. Su labor facilita la innovación técnica, aumenta la seguridad de las personas, el desarrollo de infraestructura, el acceso a la energía eficiente, la urbanización inteligente, y la mitigación del cambio climático. Además, reúne a más de 170 países y proporciona una plataforma de normalización global, neutral e independiente de 20,000 expertos en todo el mundo, al mismo tiempo publica alrededor de 10,000 normas internacionales que, junto con la evaluación de la conformidad, proporcionan el marco técnico para construir elementos de calidad, como los sistemas de información (IEC, s. f.).

IEEE

El Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), surgió en 1884. Es un grupo selecto de ingenieros dedicados a la regulación de las tecnologías en general. Entre ellas destacan; la electrónica, las ciencias, y las tecnologías de la información. Cuenta con alrededor de 425 000 miembros en 160 territorios. Es la más grande sociedad mundial sin ánimo de lucro formada por expertos en distintas ramas de la ingeniería. Entre ellas, la ingeniería de software que comprende la calidad de software dentro de sus ramas (IEEE, 2021).

ISTQB

La Junta Internacional de Calificaciones de Pruebas de Software (ISTQB), fue fundada en noviembre de 2002 y es una asociación que busca promover el valor de las pruebas de

software como profesión para individuos y organizaciones. Actualmente la ISTQB certifica diferentes roles de las pruebas de software mediante las mejores prácticas y técnicas que han resistido la prueba del tiempo. Además, utilizan como base las ISO referente a pruebas de software otorgadas por la IEC y EEE. Estar certificado bajo la ISTQB significa que la persona sigue las mejores prácticas, técnicas y estándares para las pruebas de software (ISTQB, s. f.).

CAPÍTULO 3: ESTADO DEL ARTE

A continuación, se hará referencia a algunos proyectos de investigación que han sido propuestos y se relacionan con la temática: "Pruebas de software". Tema de esta tesis.

En Argentina, se realizó una propuesta tecnológica para valorar la usabilidad de prototipos de software. Esto, se ejecutó a través de preguntas que permitían evaluar la opinión de los clientes con respecto a lo presentado. Para ello, se analizaron diferentes nociones de Interfaz de Usuario (IU). La intención era establecer el valor que otorgan a la usabilidad las organizaciones que generan software en la región del Nordeste Argentino (NEA). Los resultados recabados en este análisis permitieron encontrar que las compañías de software dan poca intervención a los clientes en las etapas de diseño del sistema desarrollado (Mascheroni et al., 2014).

En el caso de Cuba, las empresas dedicadas al desarrollo han generado modelos de referencia, los cuales, definen un conjunto de buenas prácticas para las pruebas de software, aunque, no toman en cuenta lo definido en la Organización Internacional de Normalización (ISO). Con respecto a las buenas prácticas se generó una propuesta que desarrolló procesos internos en organizaciones de este tipo, con la finalidad de: aumentar la eficiencia de las pruebas de software, acreditarse y certificarse en las normas NC-ISO/IEC 9001: 2008, NC-ISO/IEC 17025:2006 y NC- ISO/IEC (Capote García et al., 2014).

Con relación al país de Colombia, se generó un artículo que analizó el proceso de pruebas de software de la empresa SYSNET. El cual, está fundamentado en la norma ISO 9001 y el modelo CMMI. Para su ejecución, se utilizó como base de prueba las especificaciones funcionales de la plataforma ASISTO, presentando de esta manera, como la aplicación de pruebas de software permite generar calidad en los sistemas de información (Blanquicett et al., 2018).

Respecto al caso de Brasil, se llevó a cabo una investigación, la cual fue un estudio experimental dentro de una escuela para conocer el impacto de la reutilización de casos de prueba durante el aprendizaje de la programación. El objetivo era evaluar si el uso de casos de prueba puede mejorar la calidad de los programas implementados por los estudiantes. En este sentido, el ejercicio involucró dos temas de programación: vectores y manipulación de matrices. Los estudiantes se dividieron al azar en dos grupos, en los que un grupo de estudiantes recibió solo la especificación del programa y el otro grupo recibió la especificación del programa y las pruebas a aplicar. Al finalizar, la calidad de los programas aumentó de 5.3 a 7.4 al usar un enfoque de pruebas. Con ello, se pudo aumentar la calidad de los programas generados por los estudiantes, motivándolos a aplicar pruebas de software durante el desarrollo de los programas informáticos (S. Brito et al., 2012).

La tesis “Visión preventiva de las pruebas en el software” presenta una guía de fácil entendimiento basado en la experiencia y observaciones de expertos en pruebas de software. Esta, fue implementada en el Sistema de Inscripciones de Aspirantes (SINSAS) del Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS), en donde las pruebas fueron vistas como una ayuda para la identificación y corrección de todos los conflictos que se presentaron durante la etapa de análisis y diseño del SINSAS (Herrera, 2001).

La “Guía para la elaboración de un plan de pruebas para el desarrollo de sistemas de software: el caso del Sistema Integral de Información Financiera (SIIF) en la Comisión Nacional de Seguros y Fianzas (CNSF)”, se menciona que dentro de esta comisión no existe un área de pruebas de software, las pruebas son realizadas por el personal de la subdirección de mesa de ayuda y son ejecutadas cuando los sistemas son liberados. La guía propone identificar a los participantes del SIIF y asignar roles, los cuales tengan diferentes actividades, entre ellas: generar planes, diseños y casos de prueba (De las Nieves, 2007).

La “Propuesta para la implementación del proceso de pruebas de software en una institución educativa” propone la implementación de un proceso de pruebas de software en el Centro de Informática de la Facultad de Contaduría y Administración de la UNAM (CIFCA). La finalidad de este proyecto fue el de incrementar la calidad en los sistemas desarrollados, reducir el costo de mantenimiento de los sistemas y minimizar el impacto tanto tecnológico como económico. Para ello, se utilizó el Modelo Especializado en Pruebas (TMMI) y sus diferentes niveles para generar dicha propuesta (González, 2015).

En la investigación “Las pruebas en el desarrollo de software”, se explica de manera concisa el papel de las pruebas en las metodologías de desarrollo. Asimismo, se mencionan temas como: técnicas, tipos y etapas de pruebas. Además, presenta el proceso de pruebas al módulo “Agenda”. En donde se intenta determinar si el módulo cumple con los elementos mínimos necesarios para ser liberado basándose en la teoría encontrada (Campos, 2015).

En el trabajo “Desarrollo de un modelo de pruebas y calidad de software para la empresa seguros ATLAS S.A”. Se manejaron temas como: Metodologías de desarrollo, pruebas de software, y la situación actual de la empresa ATLAS. En esta investigación, se propone un

modelo de pruebas que tiene como finalidad asegurar que los requerimientos del cliente se cumplan y que la corrección de errores sea en etapas tempranas (Espinosa, 2016).

La Junta Internacional de Cualificaciones de Pruebas de Software (ISTQB). Es una organización de certificación de la calidad del software. Esta, se encarga de soportar y definir un esquema de certificación internacional para el tema de las pruebas de software. Aporta el plan de estudios y el glosario para el entendimiento de las pruebas de software. Dentro de sus planes de estudios se manejan tres niveles: básico, avanzado, y experto. Cada uno de ellos enfocado a comprender diferentes actividades de las pruebas de software (ISTQB, s. f.).

La ISO/IEC/IEEE 29119 define un conjunto de normas para las pruebas de software, se especifican conceptos clave y generales; los temas que maneja son: introducción a las pruebas de software, planes de prueba, estrategias de prueba, marcos de trabajo, diseño de pruebas, ejecución de pruebas, administración de proyectos enfocado a pruebas, defectos, y administración de incidencias. (ISO, 2021).

La ISO/IEC 25010 plantea un modelo de calidad formado por las siguientes ocho propiedades: adecuación funcional, eficiencia de desempeño, compatibilidad, usabilidad, fiabilidad, seguridad, mantenibilidad y portabilidad. Cada una de ellas interactúan con las diferentes características de los sistemas informáticos (ISO, 2011).

La ISO/IEC 20246 propone un marco para revisiones de productos de trabajo (casos de uso, historia de usuarios y código fuente), las cuales pueden consultar y utilizar todas las organizaciones involucradas en la gestión, desarrollo, prueba y mantenimiento de sistemas y software (ISO, 2017).

ANÁLISIS DEL ESTADO DEL ARTE

A continuación, se presenta el análisis del estado del arte. En este apartado se revisarán las debilidades de cada autor con respecto a su aportación a las pruebas de software.

Tabla 1

Debilidades identificadas dentro de la literatura analizada

Autor	Debilidades
Autor 1	D1.1.
Propuesta Tecnológica para Contribuir a la Evaluación de la Usabilidad del Software Referencias (Mascheroni et al., 2014).	Al aplicar pruebas de usabilidad en un prototipo se trabaja sobre un sistema sin funcionalidad, de tal manera que la interacción del usuario solo es superficial. Por lo mismo, no se conoce como reaccionará el usuario al trabajar con las funcionalidades una vez implementadas. Esto puede ocasionar que lo previamente validado genere confusión y actualizaciones en las interfaces previamente creadas.

Autor	Debilidades
	D1.2. Enfocarse solamente a la experiencia de usuario puede ocasionar que se genere una expectativa alta del sistema aun cuando no se ha completado el desarrollo.
	D1.3. La aplicación de pruebas de usabilidad siempre requerirá la disponibilidad de los clientes finales, la cual puede variar dependiendo de sus actividades y horarios.
	D1.4. Las pruebas de usabilidad al implementarse en etapas tempranas limitan la posibilidad de cambios futuros.
Autor 2 Estrategia para desarrollar la perspectiva Procesos internos en un laboratorio de pruebas de software (Capote García et al., 2014).	D2.1. Dentro de los procesos internos de la estrategia se menciona capacitar al personal según su rol, los cuales no se definen. Asimismo, falta señalar en que temas se van a capacitar.
Autor	Debilidades

	D2.2. La estrategia plantea aumentar la eficiencia de las pruebas de software desde un ámbito gerencial y no interno de las áreas de desarrollo. De esta manera las áreas de desarrollo conocen que deben de hacerlas; sin embargo, no conocen como llevarlas a cabo.
	D2.3. La poca información brindada acerca de las pruebas de software no alcanza para comprender cómo llevar a cabo tal proceso.
Autor 3 Prácticas de Pruebas desde la Industria de Software. La Plataforma ASISTO como Caso de Estudio (Blanquicett et al., 2018).	D3.1. Describe el proceso de pruebas. El cual, se rige por un formato llamado “Plan de pruebas”. En donde se observan los módulos y las pruebas por aplicar, pero no especifican cuáles serán los entregables que surgirán a raíz del plan de pruebas. Estos pueden ser: casos de pruebas, reportes de defectos, etc.
	D3.2. Dentro de las prácticas se mencionan los diferentes tipos de pruebas que se aplican dentro de la

	plataforma, pero no mencionan su significado y de qué manera las llevaron a cabo.
Autor	Debilidades
	D3.3. Menciona que realizan un reporte de defectos encontrados; sin embargo, no especifican el tratamiento de la incidencia reportada. Es decir. Cuál fue su resolución final y quién fue el encargado de atenderla.
	D3.4. Las prácticas de pruebas se hacen de acuerdo con un checklist. Este es un tipo de técnica; y esta información podría incorporarse al plan de pruebas.
	D3.5. El aplicar pruebas basadas en una lista de verificación tiene como limitante poner a prueba la imaginación del probador para encontrar defectos. Haciendo de esto que las pruebas aplicadas solo cumplan con elementos ya definidos, limitando la posibilidad de encontrar nuevos defectos.

Autor	Debilidades
Autor 4 Una experiencia sobre la aplicación de pruebas de software para la enseñanza de cursos de introducción a la programación (S. Brito et al., 2012).	D4.1. Describe la importancia de realizar pruebas de software, y cómo estás ayudan a la calidad de los sistemas de información; sin embargo, no profundiza en la teoría y en lo existente a este tema.
	D4.2. Para la realización de este estudio a los alumnos se les otorgó los elementos por probar. Omitiendo la autocrítica dentro de sus propios programas. De esta manera se limita a los futuros desarrolladores pensar de qué manera sus programas podrían tener una mejor calidad.
	D4.3. La aplicación de pruebas utilizada dentro de este estudio estuvo enfocada al código; sin embargo, al atender un solo enfoque, se omite pensar en la visión general del sistema, ya que los defectos pueden encontrarse en requerimientos mal redactados, que al

	llevarse a código presenten inconsistencias con las solicitudes del cliente.
Autor	Debilidades
Autor 5 Visión preventiva de las pruebas en el software (Herrera, 2001).	D5.1. Define conceptos básicos relacionados a las pruebas de software, pero hay carencia de un enfoque teórico que permita conocer más del tema.
	La metodología propuesta consta de tres fases, pero omite el tratamiento de las incidencias reportadas. Se comprende que la propuesta finaliza hasta que se identifica una incidencia. Dejando a un lado la pregunta ¿Cómo resolver lo encontrado?
	D5.2. La propuesta aborda la técnica de prueba de requisitos y con ella toma la base de una visión preventiva. La cual atiende a la importancia de contar con requerimientos establecidos para llevar a cabo el desarrollo; sin embargo, en entornos ágiles o de requerimientos cambiantes esta técnica puede presentar complejidades.

Autor	Debilidades
Autor 6 Guía para la elaboración de un plan de pruebas para el desarrollo de sistemas de software: El caso del SIIF en la CNSF (De las Nieves, 2007).	D6.1. La propuesta presenta artefactos de entrada y salidas, dentro de sus fases; sin embargo, algunos artefactos de entrada se repiten en cada fase. Dejando de manera confusa en que momento los artefactos de entrada dejan de actualizarse
	D6.2. La guía está desarrollada al uso de la herramienta Rational. La cual, permite gestionar el proceso de pruebas de software, sin embargo, no hay documentación de esta.
	D6.3. Las fases descritas en la guía omiten presentarse en el caso prácrito. En este sentido, solo se aprecia cómo se atendió el proceso de pruebas dentro de la herramienta rational omitiendo lo descrito anteriormente.
	D6.4. El material desarrollado dentro de la guía presenta las secciones precondiciones y postcondiciones. Los cuales carecen de información. En caso de no tener

	estos datos, sería importante justificar por qué no se presentan
Autor	Debilidades
Autor 7 Propuesta para la implementación del proceso de pruebas de software en una institución educativa (González, 2015).	D7.1. En la propuesta se presenta el desarrollo del plan de pruebas en donde se toman como base diferentes elementos que permiten conocer el alcance de las pruebas; sin embargo, no se ven desarrollados en la propuesta.
	D7.2. En la etapa de ejecución, existe el levantamiento de defectos, y a su vez también se encuentra el reporte de incidencias. En este sentido no se entiende cuál es la diferencia entre una y otra.
	D7.3. Para la fase de diseño se propone identificar documentación referente a los sistemas de información, pero no siempre se cuenta con los mismos. Partir de elementos sin fomentar la imaginación del probador a partir de un sistema limita la versatilidad de las pruebas.

Autor	Debilidades
Autor 8	D8.1.
Las pruebas en el desarrollo de software (Campos, 2015).	En el marco teórico se mencionan las técnicas de pruebas, pero de manera muy breve. Siendo este un punto más extenso, debería de dedicársele un apartado más extenso.
	D8.2.
	Plantea un enfoque teórico de las pruebas de software. Sin ejemplos, y material de ayuda que apoye a comprender cuál es el sentido de esta labor.
	D8.3.
	Dentro del caso práctico se encontraron dos defectos, los cuales se indicaron; sin embargo, no se reportaron. Tener la habilidad de comunicar defectos es una actividad principal de las pruebas de software.
Autor	Debilidades
Autor 9	D9.1.
Desarrollo de un modelo de pruebas y calidad de software para la empresa seguros ATLAS (Espinosa, 2016).	Define que el probador es el encargado de la valoración de la calidad; sin contemplar las actividades que realizará para llevar a cabo dicha actividad.

Autor	Debilidades
	D9.2. El desarrollador es el encargado de generar tres actividades de pruebas, las cuales, pueden estar viciadas al ser la persona que desarrolló el sistema.
	D9.3. Menciona la elaboración de un plan de pruebas, pero omite presentar material de apoyo que muestre cómo realizarlo.
Autor 10 ISTQB (ISTQB, 2018)	D10.1. Excluye presentar material de apoyo como plantillas o ejercicios prácticos que permitan reforzar la teoría.
	D10.2. Menciona conceptos que no vienen en la teoría, por lo mismo, la definición de estos queda a la expectativa de otros materiales referente a pruebas de software.
	D10.3. Carece de ejemplos prácticos que permitan comprender la teoría.
Autor 11 ISO/IEC/IEEE 29119 (ISO, 2021)	D11.1. Presenta plantillas que ayudan al entendimiento, pero comprenderlas y aplicarlas requieren una curva de aprendizaje.

	D11.2. Comprende descripciones de procesos de prueba que definen los procesos de prueba de software a nivel organizacional, nivel de gestión de pruebas y niveles de prueba dinámicos, pero su integración involucra la comprensión de más áreas de una organización. Con ello su implementación se vuelve compleja debido a la comunicación entre áreas.
Autor	Debilidades
Autor 12 ISO/IEC 25010 (ISO, 2011)	D12.1. Presenta un modelo para llevar a cabo las pruebas de software; sin embargo, es un proceso avanzado para las personas que van empezando en esta área.
	D12.2. Algunos conceptos presentados requieren conocer de otros temas. En este sentido la documentación otorgada no es suficiente para implementar el modelo de pruebas.
	D12.3. Estudiar cada fase del modelo involucra conocer software y técnicas que permitan resultados concretos, las cuales no se mencionan en la documentación.
Autor 13 ISO/IEC 20246 (ISO, 2017)	D13.1. Plantea la revisión de productos de trabajo, como: casos de uso, historia de usuarios y código fuente, pero si en las organizaciones no se cuenta con este tipo de documentación la aplicación de esta ISO queda fuera del entorno de pruebas.

CAPÍTULO 4: MARCO DE INVESTIGACIÓN

PROBLEMA DE INVESTIGACIÓN

El no tener un proceso de pruebas de software basado en un enfoque teórico y práctico genera procedimientos ambiguos o confusos dentro de las áreas de desarrollo. Dichos procedimientos, atienden peticiones específicas, que se limitan a atender un requerimiento o funcionalidad de un sistema en proceso de desarrollo. Además, el no tener una teoría detrás de estos, generan vacíos en el personal que está llevando a cabo la actividad, ya que no conocen el fundamento de lo que están realizando, y solo se limitan a la práctica. Minimizando nuevos aportes para el aumento de la calidad de los sistemas.

JUSTIFICACIÓN DE LA INVESTIGACIÓN

La razón de este estudio tiene como finalidad aportar una propuesta técnica que permita atender las debilidades encontradas en la literatura relacionada a las pruebas de software. De tal manera, que las áreas de desarrollo que necesiten llevar a cabo un proceso de pruebas de software puedan conocer una metodología y el enfoque teórico que hay detrás de la misma para atender la calidad de los sistemas de información.

OBJETIVO DE LA INVESTIGACIÓN

Generar una propuesta técnica acerca de un ciclo de pruebas de software. Que permita establecer un enfoque teórico, etapas, y actividades para su gestión. Podrá ser llevada a cabo, por las áreas de desarrollo que necesiten implementar un proceso de calidad de software.

PREGUNTA DE INVESTIGACIÓN

La falta de atención hacia las pruebas de software ¿Está relacionada con el desconocimiento del enfoque teórico y práctico de las mismas?

HIPÓTESIS

Fundamentar, enseñar, y establecer una metodología relacionada a las pruebas de software, permitirá su adopción dentro de las áreas de desarrollo, transformándose en una etapa primordial para el desarrollo de sistemas.

CAPÍTULO 5: METODOLOGÍA DE LA INVESTIGACIÓN

INVESTIGACIÓN DOCUMENTAL

La presente investigación es documental, la cual (Rocha, 2016): Consiste en analizar diferentes recursos de información. Con el propósito de proponer o mejorar una determinada investigación. En este caso, el proceso de pruebas de software. Para ello, en este estudio, se analizarán: propuestas, estándares, casos reales, artículos y trabajos de investigación. En donde, se buscarán inconsistencias. Con el fin de realizar una propuesta que ayude a comprender de mejor manera este tema desde un enfoque teórico hasta un entorno práctico.

CAPÍTULO 6: MARCO DE REFERENCIA

A continuación, se presentan los marcos de referencia que se utilizaran para generar la base teórica, práctica y metodológica de esta propuesta técnica.

Tabla 2

Marcos de referencia y su aporte al enfoque teórico

ENFOQUE TEÓRICO	
Marco de referencia	Proceso por cubrir de la propuesta
ISTQB (ISTQB, 2018)	Definición de pruebas de software
	La psicología y las pruebas de software
	Diferencia entre aseguramiento de la calidad y las pruebas de software
	Objetivos principales de las pruebas de software
	Conceptos clave
	• Error
	• Falla
	• Incidencia
	Principios de las pruebas de software
	• Las pruebas muestran la presencia de defectos, no su ausencia
	• Las pruebas exhaustivas son imposibles
	• Las pruebas tempranas ahorran tiempo y dinero
	• Los defectos se agrupan
	• Paradoja del pesticida
	• Las pruebas dependen del contexto
	• La ausencia de errores es una falacia
	Niveles de pruebas
	• Nivel de componente
	▪ Objetivos
	▪ Elementos de revisión
	• Nivel de integración
	▪ Objetivos
	▪ Elementos de revisión
	• Nivel de sistema
	▪ Objetivos
	▪ Elementos de revisión
	• Nivel de aceptación
ISTQB (ISTQB, 2018)	▪ Objetivos
	▪ Elementos de revisión
	Tipos de prueba
	• Pruebas funcionales
	• Pruebas no funcionales

ENFOQUE TEÓRICO	
Marco de referencia	Proceso por cubrir de la propuesta
ISTQB (ISTQB, 2018)	• Pruebas de caja blanca
	• Pruebas de caja negra
	• Pruebas relacionadas con el cambio
	▪ Pruebas de confirmación
	▪ Pruebas de regresión
	• Pruebas estáticas
	• Pruebas dinámicas
ENFOQUE PRÁCTICO	
Marco de referencia	Proceso por cubrir de la propuesta
ISO/IEC 20246 (ISO, 2017)	Técnicas de pruebas
	• Técnicas de pruebas de caja negra
	▪ Segmentación de equivalencia
	▪ Análisis del valor límite
	▪ Pruebas de la tabla de decisiones
	▪ Pruebas de transición de estado
	▪ Pruebas de caso de uso
	• Técnicas de pruebas de caja blanca
	▪ Pruebas de sentencia
	▪ Pruebas de decisión
ISO/IEC/IEEE 29119 (ISO, 2021)	• Técnicas de pruebas basadas en la experiencia
	▪ Predicción de errores
	▪ Pruebas exploratorias

Tabla 3

Marcos de referencia y su aporte al enfoque metodológico

METODOLOGÍA	
Marco de referencia	Proceso por cubrir de la propuesta
ISO/IEC 25010 (ISO, 2011)	Etapa 1. Planificación
ISO/IEC/IEEE 29119 (ISO, 2021)	Etapa 2. Diseño y ejecución
ISO/IEC 25010 (ISO, 2011)	Etapa 3. Gestión
	Etapa 4. Reporte de incidencias
	Etapa 5. Finalización

CAPÍTULO 7: PROPUESTA DE UN CICLO DE PRUEBAS DE SOFTWARE PARA EMPRESAS U ORGANIZACIONES DEDICADAS AL DESARROLLO DE SISTEMAS

El marco teórico previamente mencionado será desglosado a través de dos enfoques. Teórico y práctico. Cada uno de ellos, servirá para darle forma a esta propuesta, en donde se tomarán los conceptos, procesos, y recomendaciones que ayuden a entender las pruebas de software de una manera clara y entendible. En este sentido, el enfoque teórico servirá de apoyo al momento de llevar a la práctica todas las fases de esta forma de la metodología propuesta.

ENFOQUE TEÓRICO

El enfoque teórico descrito a continuación tiene como finalidad ser la base de conocimiento para el entendimiento de la metodología propuesta.

DEFINICIÓN DE PRUEBAS DE SOFTWARE

Es el proceso en el cual se somete a un sistema de información a un conjunto de escenarios y casos previamente definidos con el fin de identificar la mayor cantidad de defectos posibles antes de su liberación al usuario o cliente final (ISTQB, 2018).

LA PSICOLOGÍA Y LAS PRUEBAS DE SOFTWARE

En la psicología humana existe un fenómeno llamado sesgo de confirmación. El cual, puede dificultar la aceptación de información que no esté de acuerdo con las creencias actuales (BBC News Mundo, 2017). En la situación de los desarrolladores de software, esto se manifiesta al esperar que su código sea conveniente, poseen un sesgo de confirmación que obstaculiza la aprobación de que el código sea erróneo. Además, esto puede dificultar que los individuos comprendan o acepten la información derivada de las pruebas. Por otro lado,

es un rasgo humano común culpar al portador de malas noticias, y la información producida por las pruebas constantemente tienen puntos de mejora. Para intentar minimizar estas percepciones, la información acerca de deficiencias y fallos debe comunicarse de forma constructiva (ISTQB, 2018).

DIFERENCIA ENTRE ASEGURAMIENTO DE LA CALIDAD Y LAS PRUEBAS DE SOFTWARE

Mientras que las personas a menudo usan la frase aseguramiento de la calidad para referirse a las pruebas de software. El aseguramiento de la calidad y las pruebas no son lo mismo, pero están relacionadas a través del concepto: "Gestión de la calidad". La gestión de la calidad incluye tanto el aseguramiento de la calidad como el control de la calidad. El aseguramiento de la calidad generalmente se enfoca en la observancia de los procesos adecuados, mientras tanto, el control de calidad involucra varias actividades, entre ellas, las de prueba, que apoyan el logro de niveles adecuados de calidad (ISTQB, 2018).

OBJETIVOS PRINCIPALES DE LAS PRUEBAS DE SOFTWARE

Los objetivos de las pruebas de software se pueden comprender en los siguientes puntos:

- Prevenir defectos (ISTQB, 2018).
- Encontrar fallos y defectos (ISTQB, 2018).
- Evaluar requerimientos, historias de usuario, epopeyas, y casos de uso, también conocidos como productos de trabajo (ISTQB, 2018).
- Validar que los sistemas de información funcionen de acuerdo con lo esperado por los clientes y usuarios finales (ISTQB, 2018).
- Aumentar la calidad esperada de los sistemas de información (ISTQB, 2018).

CONCEPTOS CLAVE

Muchas veces: error, falla e incidencia son usados como sinónimos, pero en realidad son cosas distintas. De una forma sencilla se pueden definir como:

ERROR

Acción que se realiza sin intención de cometerla. En el caso del desarrollo de sistemas, se pueden cometer muchos errores, entre ellos: sintaxis, lógica, interfaz de usuario, implementación, etc. (ISTQB, 2018).

FALLA

Es la manifestación del error. En ocasiones, se identifican fallas en los sistemas de información, sin embargo, no se conocen los errores que las detonan (ISTQB, 2018).

INCIDENCIA

Es la evidencia de una falla. En este punto se conocen los pasos para poder reproducirla o los factores externos o internos que la generan (ISTQB, 2018).

PRINCIPIOS DE LAS PRUEBAS DE SOFTWARE

LAS PRUEBAS MUESTRAN LA PRESENCIA DE DEFECTOS, NO SU AUSENCIA

Las pruebas ayudan a encontrar fallas; sin embargo, no demuestran la ausencia de defectos (ISTQB, 2018).

LAS PRUEBAS EXHAUSTIVAS SON IMPOSIBLES

Intentar probar todas las funcionalidades de un sistema no es factible, solo en casos triviales. Para ello, se pueden utilizar diferentes técnicas de pruebas que permitan abarcar la mayor cantidad de funcionalidades con el fin de encontrar la mayor cantidad de defectos (ISTQB, 2018).

LAS PRUEBAS TEMPRANAS AHORRAN TIEMPO Y DINERO

Realizar pruebas tempranas en el ciclo de vida de desarrollo del software ayuda a reducir o eliminar cambios costosos (ISTQB, 2018).

LOS DEFECTOS SE AGRUPAN

Agrupar los defectos permitirá conocer en dónde enfocar esfuerzos durante el desarrollo (ISTQB, 2018).

PARADOJA DEL PESTICIDA

Si las mismas pruebas se repiten una y otra vez, con el tiempo los mismos casos o escenarios de prueba ya no encontrarán nuevos defectos. Para detectar nuevas incidencias, será necesario generar pruebas diferentes a las originadas en un principio (ISTQB, 2018).

LAS PRUEBAS DEPENDEN DEL CONTEXTO

En algunos casos las pruebas podrán ser similares a las de otro sistema de información; sin embargo, su contexto será diferente. Las restricciones o permisiones estarán ligadas al usuario o cliente final (ISTQB, 2018).

LA AUSENCIA DE ERRORES ES UNA FALACIA

Es una falacia (es decir, una creencia errónea) esperar que simplemente encontrar y corregir una gran cantidad de defectos garantice el éxito de un sistema. Ya que la calidad final es otorgada por el usuario (ISTQB, 2018).

NIVELES DE PRUEBAS

Los niveles de prueba son actividades que se organizan y administran juntas. Actualmente, existen cuatro niveles, siendo el primero el más bajo: Prueba de componentes, prueba de integración, prueba de sistema, y prueba de aceptación. Estas se aplican al software en diferentes etapas del proceso de desarrollo (ISTQB, 2018)

NIVEL DE COMPONENTE

Las pruebas de componente o también conocidas como pruebas de unidad. Se enfocan en componentes que se pueden probar por separado a nivel de código (ISTQB, 2018).

OBJETIVOS:

- Reducir el riesgo (ISTQB, 2018).
- Encontrar defectos en el componente (ISTQB, 2018).
- Crear confianza en la calidad del componente (ISTQB, 2018).
- Prevenir que los defectos escapen a niveles de prueba más elevados (ISTQB, 2018).

ELEMENTOS DE REVISIÓN:

- Tablas de bases de datos (ISTQB, 2018).
- Código y estructura de datos (ISTQB, 2018).
- Componentes, unidades o módulos (ISTQB, 2018).

NIVEL DE INTEGRACIÓN

Las pruebas de integración. Se centran en las interacciones entre componentes o sistemas (ISTQB, 2018).

OBJETIVOS:

- Reducir el riesgo (ISTQB, 2018).
- Crear confianza en la calidad de las interfaces (ISTQB, 2018).
- Encontrar defectos (que pueden estar en las interfaces o en los diferentes componentes relacionados) (ISTQB, 2018).
- Prevenir que los defectos escapen a niveles de prueba más elevados (ISTQB, 2018).

ELEMENTOS DE REVISIÓN:

- Sistemas ligados al software (ISTQB, 2018).
- Bases de datos ligadas al software (ISTQB, 2018).
- Comunicación entre sistemas (ISTQB, 2018).
- Interfaces (ISTQB, 2018).

NIVEL DE SISTEMA

Las pruebas de sistema. Se centran en el comportamiento y las capacidades de todo un sistema o producto, a menudo considerando las tareas de extremo a extremo que el sistema puede realizar (ISTQB, 2018).

OBJETIVOS:

- Reducir el riesgo (ISTQB, 2018).
- Validar que el sistema está completo y funciona como se espera (ISTQB, 2018).
- Crear confianza en la calidad del sistema como un todo (ISTQB, 2018).
- Encontrar defectos (ISTQB, 2018).
- Prevenir que los defectos escapen a niveles de prueba más o producción más elevados (ISTQB, 2018).

ELEMENTOS DE REVISIÓN:

- Aplicaciones móviles (ISTQB, 2018).
- Sistemas de hardware/Software (ISTQB, 2018).
- Configuración del sistema y datos de configuración (ISTQB, 2018).

NIVEL DE ACEPTACIÓN

Las pruebas de aceptación, generalmente se enfocan al comportamiento y las capacidades de todo un sistema o producto (ISTQB, 2018).

OBJETIVOS:

- Crear confianza en la calidad del sistema como un todo (ISTQB, 2018).

ELEMENTOS DE REVISIÓN:

- Operatividad y configuración del sistema (ISTQB, 2018).

TIPOS DE PRUEBA

PRUEBAS FUNCIONALES

Las pruebas funcionales de un sistema involucran pruebas que evalúan las funciones que debe realizar el sistema. Los requisitos funcionales pueden describirse en productos de trabajo, como son: historias de usuario, casos de uso o especificaciones funcionales, o en su defecto pueden no estar documentados. Para ello es importante recordar que las funciones son “lo que” el sistema debe hacer (ISTQB, 2018).

PRUEBAS NO FUNCIONALES

Las pruebas no funcionales de un sistema evalúan las características de los sistemas y el software, como son la usabilidad, la eficiencia del rendimiento o la seguridad. La prueba no funcional es la prueba de “qué tan bien” se comporta el sistema (ISTQB, 2018).

PRUEBAS DE CAJA BLANCA

Las pruebas de caja blanca derivan en pruebas basadas en la estructura interna del sistema. La estructura interna puede incluir código, arquitectura, flujos de trabajo y/o flujos de datos dentro del sistema. El diseño y la ejecución de pruebas de caja blanca pueden involucrar habilidades o conocimientos especiales, como la forma en que se construye el código o cómo se almacenan los datos (ISTQB, 2018).

PRUEBAS DE CAJA NEGRA

Las pruebas de caja negra verifican la funcionalidad del software ignorando su estructura interna. Esto quiere decir que se omite la revisión de código, y de los componentes internos que hagan funcionar el sistema. Este tipo de prueba toma como base las especificaciones

técnicas como requerimientos, casos de uso, e historias de usuario. Es decir, solo se centran en las entradas y salidas de la aplicación, sin preocuparse por la estructura interna, solo importa que, al realizar cierta acción, la salida sea la indicada según los requerimientos. (ISTQB, 2018).

PRUEBAS RELACIONADAS CON EL CAMBIO

Cuando se realizan cambios en un sistema, ya sea para corregir un defecto o debido a una actualización. Se deben realizar pruebas para confirmar que los cambios no han causado consecuencias adversas o imprevistas dentro del software. Para ello, existen dos tipos de pruebas para verificar cambios. Estas son las pruebas de confirmación y de regresión (ISTQB, 2018).

PRUEBAS DE CONFIRMACIÓN

Las pruebas de confirmación se enfocan en verificar que los defectos reportados se hayan atendido correctamente. Para ello, se pueden reproducir los pasos que generaron las incidencias y verificar si estos se encuentran solucionados en su totalidad (ISTQB, 2018).

PRUEBAS DE REGRESIÓN

Es posible que un cambio realizado debido a una actualización pueda afectar accidentalmente el comportamiento del software, ya sea dentro de un mismo componente, o en otros. Tales efectos secundarios no deseados pueden repercutir en la calidad del sistema. Es por ello, que las pruebas de regresión involucran la ejecución de pruebas sobre las actualizaciones hechas para detectar efectos secundarios no deseados (ISTQB, 2018).

PRUEBAS ESTÁTICAS

Las pruebas estáticas son aquellas que se hacen sin necesidad de ejecutar el código. Un ejemplo de este tipo de pruebas puede ser la revisión estática de código. El análisis de código estático escanea el código en busca de errores y vulnerabilidades comúnmente conocidas, como fugas de memoria o desbordamientos. Cabe mencionar, que el análisis también puede incluir reglas de codificación. Por ejemplo, errores de sintaxis (ISTQB, 2018).

PRUEBAS DINÁMICAS

Las pruebas dinámicas son aquellas que se hacen ejecutando el código fuente. En esta prueba se comprueba el comportamiento del software, el uso de memoria / CPU y el rendimiento general del sistema. Generalmente se enfocan en comportamientos visibles externamente. De ahí el nombre "Dinámico" (ISTQB, 2018).

ENFOQUE PRÁCTICO

TÉCNICAS DE PRUEBAS

Las técnicas de prueba se dividen principalmente en dos rubros: Técnicas de caja negra y Técnicas de caja blanca. Cada una hace referencia al tipo de prueba anteriormente mencionado (Prueba de caja negra y caja blanca) La primera enfocada a las entradas y salidas de un sistema y la segunda a sus componentes internos. Adicionalmente, se agregan las técnicas de pruebas basadas en la experiencia (Predicción de errores y pruebas exploratorias). Las cuales, derivan de la habilidad y la intuición del responsable de pruebas y de su experiencia con sistemas similares (ISO, 2017).

TÉCNICAS DE PRUEBAS DE CAJA NEGRA

Las técnicas de prueba de caja negra se concentran en las entradas y salidas de un software. sin hacer referencia a su estructura interna. Para ello, existen las siguientes técnicas: segmentación de equivalencia, análisis del valor límite, pruebas de la tabla de decisiones, pruebas de transición de estado y pruebas de caso de uso. Las cuales, se describen a continuación (ISO, 2017).

SEGMENTACIÓN DE EQUIVALENCIA

La segmentación de equivalencia divide los datos en segmentos de tal manera que se espera que todos los miembros de un segmento dado se prueben de la misma manera (Kaner et al., 2013). Existen segmentaciones de equivalencia para valores válidos y no válidos, y estos pueden dividirse en tipo de dato, caracteres, etc. (ISO, 2017).

Consideraciones

- Los valores válidos son valores que deben ser aceptados por el componente (ISO, 2017).
- Los valores no válidos son valores que deben ser rechazados por el componente (ISO, 2017).
- Cada valor debe pertenecer a una y solo una segmentación de equivalencia (ISO, 2017).
- Cuando se usan segmentaciones de equivalencia no válidas estas deben probarse individualmente, ya que al probarse en conjunto los fallos se pueden enmascarar en un solo conjunto (ISO, 2017).

Ejemplo 1

El siguiente ejemplo tiene como propósito hacer una representación de la técnica segmentación de equivalencia. Para ello, se utiliza el campo “Contraseña” como base de la prueba.

Campo por probar: Contraseña | Ingrese su contraseña |

Consideraciones: El campo contraseña permite un máximo de 6 y 10 caracteres.

¿Cómo probarlo bajo la técnica segmentación de equivalencia?

Segmentación 1	Caso de prueba	Descripción de la prueba: Valor válido	Resultado esperado	Resultado real
Campo contraseña	Id 1	Ingresar 6 caracteres	El sistema lo permite	---
	Id 2	Ingresar 7 caracteres		---
	Id 3	Ingresar 8 caracteres		---
	Id 4	Ingresar 9 caracteres		---
	Id 5	Ingresar 10 caracteres		---
Segmentación 2	Caso de prueba	Descripción de la prueba: valor inválido	Resultado esperado	Resultado real
Campo contraseña	Id 1	Ingresar 1 carácter	El sistema no lo permite	---
	Id 2	Ingresar 2 caracteres		---
	Id 3	Ingresar 3 caracteres		---

	Id 4	Ingresar 4 caracteres		---
	Id 5	Ingresar 5 caracteres		---
	Id 6	Ingresar un número mayor a 10 caracteres		---

ANÁLISIS DEL VALOR LÍMITE

El análisis de valor de límite es una herramienta de apoyo para la segmentación de equivalencia; sin embargo, solo se puede utilizar cuando la segmentación está ordenada, y se conocen los valores mínimo y máximo de la segmentación (ISO, 2017).

Ejemplo 2

Retomando el ejemplo anterior se utilizará el campo “Contraseña” para realizar la técnica de valores límite.

Segmentación	Caso de prueba	Descripción de la prueba	Resultado esperado	Resultado real
Campo contraseña	Id 1	Ingresar de 0 a 5 caracteres en el campo de contraseña	El sistema no lo permite	---
	Id 2	Ingresar de 6 a 10 caracteres en el campo de contraseña	El sistema lo permite	---

	Id 3	Ingresar un número mayor a 10 caracteres	El sistema no lo permite	---

PRUEBAS DE LA TABLA DE DECISIONES

Las tablas de decisiones son una buena manera de seleccionar casos de prueba, ayudan a encontrar problemas o ambigüedades en las especificaciones o requerimientos. Al crear tablas de decisiones, el probador debe de identificar las condiciones de entrada, las posibles combinaciones entre cada condición, y las condiciones de salida. Estas forman las filas de la tabla, generalmente con las condiciones en la parte superior y las acciones en la parte inferior (ISO, 2017).

Las combinaciones se dan a través de la siguiente regla. Se toma como base el número dos y se eleva de acuerdo con el número de condiciones. Obteniendo la siguiente fórmula (ISO, 2017):

$$2^{(n\ condiciones)} = número\ de\ combinaciones.$$

Una vez obtenido el número de combinaciones, este se divide entre dos para conocer la cantidad de escenarios verdaderos y falsos (ISO, 2017).

Número de combinaciones / 2 = cantidad escenarios verdaderos y falsos

Para conocer la cantidad de escenarios verdaderos y falsos de las siguientes condiciones se toma la cantidad de escenarios verdaderos y falsos de la condición anterior y se divide entre dos. Consiguiendo la siguiente fórmula (ISO, 2017):

Cantidad escenarios verdaderos y falsos de la condición anterior / 2 = Cantidad escenarios verdaderos y falsos de la segunda condición

Cabe mencionar que estas tres fórmulas aplican para “n” cantidad de condiciones. Siendo así que se obtienen todas las cantidades de combinaciones posibles (ISO, 2017).

Ejemplo 3

Para poner en práctica la técnica prueba de la tabla de decisiones. Existe un sistema que permite realizar la compra de un vehículo. Para ello, primero se debe ingresar el modelo del vehículo y el color de este. Al ingresar ambos datos el sistema permite continuar con la compra, al omitir alguno el sistema presenta un mensaje de error.

Formulación de la tabla

Paso 1. Identificar condiciones

Condiciones
¿Se ingresó el modelo del vehículo?
¿Se ingresó el color del vehículo?

Paso 2. Identificar todas las posibles combinaciones

Para conocer las combinaciones se utiliza la siguiente fórmula: $2^{(2 \text{ condiciones})} = 4$ *combinaciones.*

Condiciones	Combinación 1	Combinación 2	Combinación 3	Combinación 4
¿Se ingresó el modelo del vehículo?				
¿Se ingresó el color del vehículo?				

Para conocer la cantidad de escenarios verdaderos o falsos se realiza la siguiente fórmula: *4 combinaciones / 2 = 2.* Esto significa que la tabla para la primera condición tendrá 2 condiciones verdaderas y 2 condiciones falsas.

De esta manera se obtiene la siguiente tabla:

Condiciones	Combinación 1	Combinación 2	Combinación 3	Combinación 4
¿Se ingresó el modelo del vehículo?	Verdadero	Verdadero	Falso	Falso
¿Se ingresó el color del vehículo?				

Para conocer la cantidad de escenarios verdaderos o falsos de la siguiente condición se realiza la siguiente fórmula: *2 / 2 = 1*. Esto significa que la tabla para la segunda condición tendrá una condición verdadera y una falsa. Obteniendo la siguiente tabla:

Condiciones	Combinación 1	Combinación 2	Combinación 3	Combinación 4
¿Se ingresó el modelo del vehículo?	Verdadero	Verdadero	Falso	Falso
¿Se ingresó el color del vehículo?	Verdadero	Falso	Verdadero	Falso

Paso 3. Identificar las salidas correspondientes

Como anteriormente se mencionó al ingresar ambas condiciones el sistema permite continuar con la compra, al omitir alguna de ellas, el sistema presenta un mensaje de error. Por lo mismo, la tabla se actualiza de la siguiente manera:

Condiciones	Combinación 1	Combinación 2	Combinación 3	Combinación 4
¿Se ingresó el modelo del vehículo?	Verdadero	Verdadero	Falso	Falso
¿Se ingresó el color del vehículo?	Verdadero	Falso	Verdadero	Falso
Acciones/Salidas				

¿Se ingresaron ambos datos?	El sistema procesa la solicitud	El sistema presenta el mensaje de error	El sistema presenta el mensaje de error	El sistema presenta el mensaje de error

Paso 4. Redactar los casos de prueba

Una vez identificadas las condiciones y sus respectivas salidas. Se procede a redactar el caso de prueba. El cual, será el elemento formal que contendrá cada combinación y sus respectivos pasos a ejecutar.

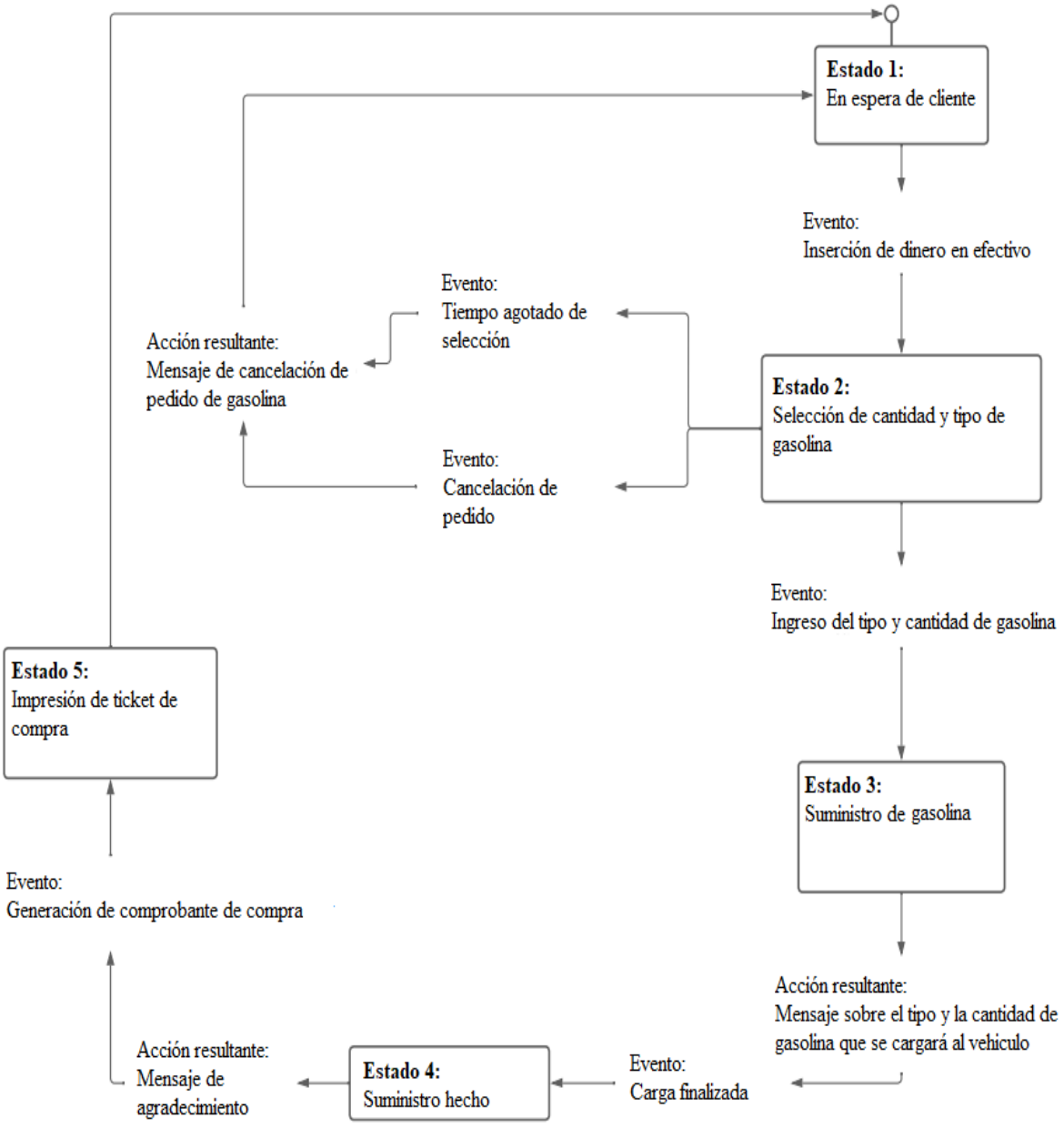
Para su redacción se recomienda consultar la segunda fase de la metodología propuesta, llamada" Diseño".

PRUEBAS DE TRANSICIÓN DE ESTADO

Un diagrama de transición de estado muestra los posibles estados del software, de esta manera se pueden observar los diferentes caminos que se pueden originar de un estado "A" a un estado "B". Para ello, primero se deberán de identificar los estados a probar. Después, la transición que genera el cambio, y el evento que lo desencadena. Por último, la acción resultante al pasar de un estado a otro. Cabe mencionar, que este tipo de prueba es adecuada para modelar escenarios de negocio que tengan estados específicos. Además, permite corroborar la usabilidad del sistema en cuanto a la navegación (ISO, 2017).

Ejemplo 4

El siguiente diagrama de estados está enfocado a un sistema de venta de gasolina. En él, se pueden observar los estados, eventos y acciones del software que el usuario deberá recorrer para poder adquirir el producto. Cabe mencionar que este diagrama se puede generar con solo interpretar el software que se desea probar.



PRUEBAS DE CASO DE USO

Un caso de uso describe cómo interactúa un usuario final con un sistema de información en circunstancias específicas. Esta interacción se basa en: una historia, un lineamiento de tareas, una descripción, o una representación gráfica. La cual, después será transformada en un requerimiento. En resumen, un caso de uso ilustra el software o sistema desde el punto de vista del usuario final (ISO, 2017).

Las pruebas de este tipo están enfocadas al cumplimiento de los requerimientos del software. Para ello, se toma en cuenta lo descrito en los casos de uso y se valida que lo mencionado se cumpla. Con esto, se verifica que lo acordado en reuniones previas esté presente en el sistema. Los casos de uso más comunes son los de tipo UML (Lenguaje Unificado de Modelado); sin embargo, los podemos encontrar en diversas presentaciones. Dependerá de la institución de qué manera los plasme (ISO, 2017).

TÉCNICAS DE PRUEBAS DE CAJA BLANCA

Las pruebas de caja blanca se basan en los componentes internos (Código) de los sistemas de información, con el objetivo de mejorar el uso eficiente del software. Para ello, existen las siguientes técnicas: Pruebas de sentencia y de decisión (ISO, 2017).

PRUEBAS DE SENTENCIA

Las sentencias de programación generalmente acostumbran a tener un carácter que establece su final. Por ejemplo, un punto y coma (;). Esto depende del lenguaje de programación que se esté utilizando. En ciertos lenguajes de programación las sentencias permanecen numeradas, de tal manera, que si existe cualquier error de sintaxis (o alguna advertencia), el

compilador entrega un mensaje con el número de sentencia donde ha sido encontrado (Alegsa, 2002).

Las pruebas de sentencia se encargan de ejecutar las sentencias de código de un software. Esto se puede realizar en un módulo general o en una funcionalidad específica (ISO, 2017).

Para obtener la cobertura que se cubrió durante las pruebas se toma el número de sentencias ejecutadas, y se divide entre el número total de sentencias que componen el módulo o la funcionalidad requerida. Cabe mencionar que normalmente la cobertura se representa en porcentaje. Dejando de esta manera la siguiente fórmula (ISO, 2017):

 # de sentencias ejecutadas / # total de sentencias del módulo o funcionalidad = % de cobertura cubierto

PRUEBAS DE DECISIÓN

Un elemento de decisión dentro del código de un sistema se puede comprender desde las sentencias de control. Por ejemplo, if, else, switch, y case. Actualmente, los sistemas de información se componen de entradas y salidas, las cuales derivan de las sentencias de control. En este sentido, las pruebas de decisión son llevadas a cabo a través de la ejecución de sentencias de control que se encuentran en un módulo o en una funcionalidad específica. Para ello, se deben de generar pruebas con relación a cada sentencia y sus resultados posibles de acuerdo con las entradas y salidas ya definidas en el código (ISO, 2017).

Para obtener la cobertura que se cubrió durante las pruebas se debe de tomar el número de resultados ejecutados para cada sentencia de control y este se debe de dividir entre el número total de resultados de la sentencia de control. Cabe mencionar que normalmente la cobertura se representa en porcentaje. Dejando de esta manera la siguiente fórmula (ISO, 2017):

 # de resultados ejecutados de la sentencia de control / # total de resultados posibles de la sentencia de control = % de cobertura cubierto

TÉCNICAS DE PRUEBAS BASADAS EN LA EXPERIENCIA

Al ejercer técnicas de prueba fundamentadas en la vivencia, los casos de prueba se derivan de la capacidad de intuición del probador, y de su vivencia con tecnologías semejantes. Estas técnicas tienen la posibilidad de ser útiles para detectar pruebas que no se identificaron de forma sencilla con otras técnicas más metódicas (ISO, 2021).

PREDICCIÓN DE ERRORES

La predicción de errores es una técnica utilizada para anticipar la ocurrencia de defectos. Toma como base el conocimiento del probador y se complementa con las siguientes preguntas (ISO, 2021).

- ¿Cómo ha funcionado el software en el pasado? (ISO, 2021).
- ¿Qué tipo de errores son los más comunes durante la fase de diseño y desarrollo que tienden a cometer los desarrolladores? (ISO, 2021).
- ¿Qué fallos han ocurrido en otros sistemas de información realizados por el mismo equipo de trabajo? (ISO, 2021).

La aplicación de este tipo de prueba se apoya en la creación de listas con los posibles errores, fallas, e incidencias más comunes. Estas se construyen a partir de la experiencia o de alguna base de conocimiento que muestre los problemas que con más frecuencia se repiten en los sistemas desarrollados (ISO, 2021).

PRUEBAS EXPLORATORIAS

Las pruebas exploratorias consisten en examinar el software desarrollado de forma inmediata a medida que se diseñan e implementan las pruebas. Esto se lleva a cabo, a través de sesiones. Las cuales, son un enfoque práctico en el que se planifica poco y el esfuerzo se centra en probar. Como medida de planificación existe una especificación llamada “Contrato de prueba”. En donde, se menciona el alcance de la prueba, y los objetivos. En este tipo de prueba las actividades de ejecución se realizan en paralelo sin documentar formalmente las condiciones o los casos de prueba utilizados. Además, se toman algunas notas sobre los defectos encontrados y cualquier pensamiento sobre posibles pruebas adicionales para luego generar un informe que detalle los hallazgos (ISO, 2021).

Ejemplo 5

El siguiente formato ejemplifica el contrato de prueba. El cual, es utilizado como especificación para llevar a cabo las pruebas exploratorias. Este formato puede ser utilizado siempre que se requieran iniciar pruebas de este tipo. Solo bastará con adaptar el objetivo, requerimientos, notas e incidencias al sistema en cuestión.

CONTRATO DE PRUEBA	
Objetivo	
Dentro de este apartado se coloca el objetivo de las pruebas. Es decir, se coloca la funcionalidad o componentes por revisar. Por ejemplo, analizar la navegación del menú principal	
Requerimientos	
Sistema operativo	Dentro de esta sección se especifican los sistemas operativos que se estarán utilizando para llevar a cabo las pruebas. Por ejemplo, Windows 10 con arquitectura de 64 bits
Ambientes de trabajo:	Dentro de este apartado se definen los equipos que se estarán utilizando, así como el sistema operativo con el que trabajan y los navegadores que se estarán manejando y sus respectivas versiones para las pruebas. Por ejemplo: Google Chrome de 64 bits Mozilla Firefox de 64 bits Microsoft Edge de 64 bits
Fecha de inicio	Dentro de este dato se coloca la fecha y hora de inicio de las pruebas exploratorias. Por ejemplo: 18/07/2022 a las 03:00 p.m.
Fecha de fin:	Dentro de este apartado se coloca la fecha y hora de fin de las pruebas exploratorias. Por ejemplo:

	18/072022 a las 05:00 p.m.
Probador:	Dentro de esta sección se nombra al o los responsables encargados de llevar a cabo las pruebas exploratorias. Por ejemplo: Raúl Ramírez Urbano
Archivos o datos de prueba:	Dentro de este campo se especifican en caso de ser necesarios los archivos o datos de pruebas que se utilizarán para llevar a cabo las pruebas exploratorias. Por ejemplo: PDF de 50 mb para medir la respuesta del servidor responsable de guardar archivos Datos de pruebas de la base de datos Prueba.sql que cuenta con la información de todos los usuarios.
Notas de prueba	
Notas de prueba	Dentro de este dato se colocan aquellas situaciones que no se hayan considerado dentro de las pruebas y que sean de suma importancia comunicar. Por ejemplo. Actualmente el menú presenta 5 opciones; sin embargo, faltó plasmar en la documentación técnica y en el sistema la sexta opción relacionada con los datos de contacto del cliente.
Incidencias	

Dentro de este apartado se describen las incidencias encontradas durante el periodo de las pruebas exploratorias. Por ejemplo:	
Incidencia 1	En el menú principal al dar clic a la primera opción, el sistema tiene un retraso de 2 segundos en mostrar la sección seleccionada.
Incidencia 2	Al revisar el menú principal en el navegador Google Chrome las opciones se presentan con un formato diferente al mostrado en Mozilla Firefox y Microsoft Edge
Incidencia 3	En el menú principal al dar clic a la cuarta opción, el sistema omite presentar la opción seleccionada.

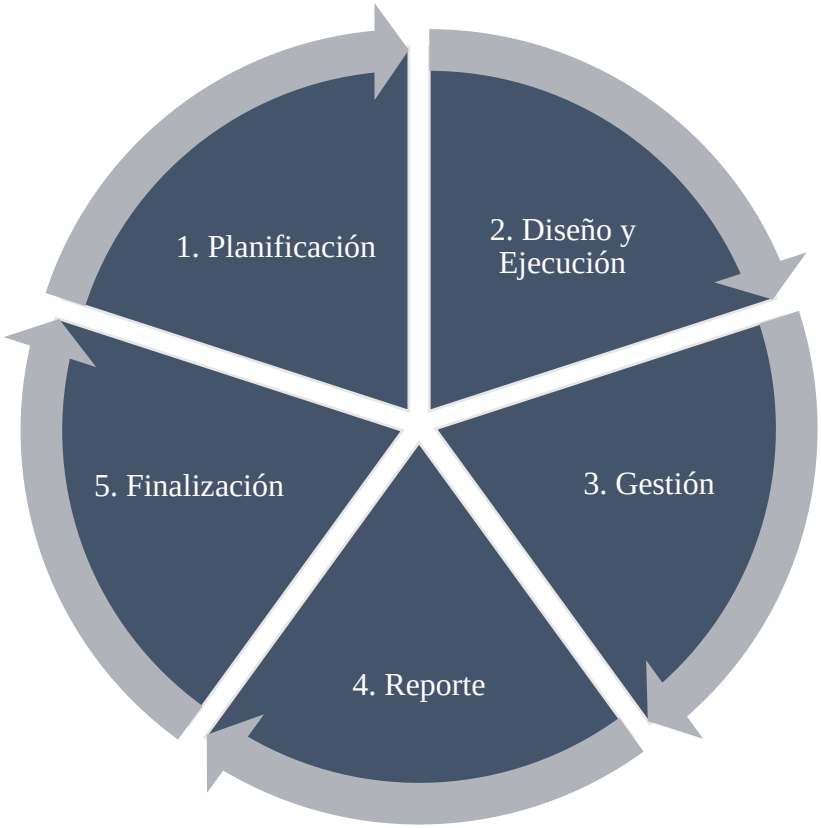
METODOLOGÍA

La siguiente metodología se compone de cinco etapas. Las cuales son: Planificación, diseño/ejecución, reporte, gestión y finalización. Cada una presenta de manera detallada su descripción y los entregables que se deben de generar. En su conjunto forman un ciclo. El cual llamaremos ciclo de pruebas de software. En este sentido, se considera como ciclo de pruebas el conglomerado de actividades que son necesarias para probar el funcionamiento completo de uno o más módulos en un periodo de tiempo específico.

Como podemos percibir en la Figura 7. Se observan las cinco fases del ciclo de pruebas. En donde se aprecia que el inicio de cada etapa es la finalización de la fase anterior.

Figura 7

Ciclo de pruebas de software



ETAPA 1. PLANIFICACIÓN

Dentro de esta etapa se analizan los objetivos, y el propósito de las pruebas. Así como los elementos de planificación para iniciar las pruebas. Como entregable de esta fase se entrega el plan de pruebas. El cual, es un documento que presenta los puntos más importantes para iniciar las pruebas (ISO, 2011).

Ejemplo 6

El siguiente formato ejemplifica el plan de pruebas con sus respectivas secciones por completar.

Plan de pruebas	
El presente documento es la base para llevar a cabo las pruebas de software. Tiene como finalidad describir los elementos que se tomarán en cuenta para la ejecución del ciclo de pruebas.	
Número de ciclo (ISO, 2011).	Dentro de esta sección se especifica el número de ciclo que permitirá llevar un histórico acerca de las veces que se ha probado una funcionalidad o módulo.
Objetivo de las pruebas (ISO, 2011).	Dentro de esta sección se especifican los elementos que se evaluarán durante las pruebas. Estos pueden ser: revisar la funcionalidad de un módulo en específico, validar que el sistema de manera general satisfaga las expectativas del cliente y de los usuarios, o la identificación de defectos en módulos que se encuentran en ambientes de producción
Alcance (ISO, 2011).	Dentro de este apartado se definen todos los elementos que se vayan a probar. Haciendo énfasis en lo que se pretende cubrir. Es importante, que dentro del alcance se comprendan los elementos que no forman parte de las pruebas. Esto para evitar confusiones

Equipo de trabajo (ISO, 2011).	En esta sección se especifican los roles que estarán participando en las pruebas. También, se puede agregar de qué manera lo harán. Si será en estado remoto o presencial. Los roles pueden ser: Líder técnico, líder de análisis, líder de desarrollo, líder de pruebas, becario de análisis, becario de desarrollo, becario de pruebas, etc.
Calendario de trabajo (ISO, 2011).	Dentro de este apartado se definen los días y los horarios en los que se estarán realizando las pruebas o puede agregarse solo el rango de fechas en las que se estará trabajando esta actividad. Por ejemplo, la fecha de inicio y fecha de fin
Ambientes de trabajo (ISO, 2011).	Dentro de este apartado se definen los equipos que se estarán utilizando, así como el sistema operativo con el que trabajan y los navegadores que se estarán manejando y sus respectivas versiones para las pruebas.
URL del sistema o aplicación (ISO, 2011).	Dentro de esta sección se ingresará la URL del sistema con la finalidad de tener presente su ubicación.
Entregables (ISO, 2011).	Para esta sección se definen los documentos técnicos que se entregarán una vez formalizadas y realizadas las pruebas. Estos pueden ser: El plan de pruebas, lista de casos de pruebas, reporte de defectos, e informe sobre las pruebas.

<table>
<tr><td colspan="3">Riesgos (ISO, 2011).</td></tr>
<tr><td>Riesgos</td><td>Impacto</td><td>Probabilidad</td></tr>
<tr><td colspan="3">Dentro de este segmento se describen los riesgos que podrían presentarse durante la actividad de pruebas. Así como su impacto y probabilidad. De tal manera que este análisis ayude a prever escenarios imprevistos.</td></tr>
<tr><td colspan="3">Tipos de pruebas por aplicar (ISO, 2011).</td></tr>
<tr><td colspan="3">Dentro de esta sección se mencionan los tipos de pruebas que se aplicarán para llevar a cabo esta actividad. Cabe mencionar que estos se describen con más detalle en el enfoque teórico. Por lo mismo es de suma importancia revisarlo. Es preciso comentar que dentro del plan de pruebas puede existir más de un tipo de prueba por aplicar.</td></tr>
<tr><td colspan="3">Tareas (ISO, 2011).</td></tr>
<tr><td colspan="3">Dentro de este apartado se describen las tareas que cada integrante de trabajo se compromete a realizar durante el periodo de pruebas.</td></tr>
</table>

ETAPA 2. DISEÑO Y EJECUCIÓN

La norma ISO 29119. Es la encargada de definir: conceptos, procesos, técnicas y documentación de las pruebas de software. En su contenido, especifica que un caso de prueba es un elemento importante para la correcta identificación de errores (ISO, 2021)

De acuerdo con la ISO 29119 un caso de prueba se compone por los siguientes elementos (ISO, 2021):

- Identificador: Establece el orden con el cual se ejecutarán las pruebas (ISO, 2021).
- Descripción/ Resumen: Presenta una breve descripción o resumen de la prueba por realizar (ISO, 2021).
- Prioridad: Toma en cuenta los siguientes valores: Alta, media, y baja. Son utilizadas para conocer su importancia de ejecución (ISO, 2021).
- Precondiciones: Establece las condiciones previas que deben de existir para poder ejecutar la prueba (ISO, 2021).
- Datos de entrada: Define los datos necesarios para llevar a cabo la prueba. En ocasiones, pueden no ser necesarios (ISO, 2021).
- Pasos: Define los pasos que serán necesarios para llevar a cabo la prueba (ISO, 2021).
- Resultado esperado: Establece el resultado que se obtendrá al finalizar la ejecución de la prueba (ISO, 2021).
- Resultado obtenido: Presenta el resultado obtenido al finalizar la ejecución de la prueba. Este puede ser exitoso o fallido (ISO, 2021).

Estos, permiten conocer a detalle la composición de las pruebas ejecutadas. Con la información anterior se generarán los casos de prueba de los objetivos propuestos en el plan de pruebas. Así mismo, una vez diseñadas serán ejecutadas con la finalidad de conocer los resultados obtenidos.

A continuación, se presentan los conceptos establecidos por la ISO 29119 para el diseño de casos de prueba (ISO, 2021).

Tabla 4

Formato para el diseño de casos de prueba

Casos de prueba							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido

ETAPA 3. GESTIÓN

Dentro de esta sección se específica cómo se gestionan las incidencias que sean reportadas por el área de pruebas. Estas, son generadas dentro de la fase 3. Diseño y ejecución, y solo harán caso a los resultados fallidos. En este punto se hace utilización de un diagrama de flujo que permite visualizar cómo será la gestión de las incidencias reportadas (ISO, 2011).

ETAPA 4. REPORTE

El reporte de defectos es una cualidad que todo probador en sistemas debe de tener. Esta permitirá comunicar las incidencias encontradas durante las revisiones de pruebas. Con la finalidad de cumplir con la necesidad de redactar defectos específicos, claros y completos, será necesario tomar en cuenta las siguientes consideraciones para la redacción de defectos.

- Tener un esquema.

Es de gran ayuda tener un esquema que, con menor o mayor detalle, trace una guía que debe seguir el texto. Esto evita perderse y, al asignar espacios a los puntos que se abordarán, le otorga estructura y equilibrio al escrito, y elude la desproporción.

[Ubicación] [Acción] [lo que realiza el sistema]

Ejemplo:

En él [Inicio de sesión]", al [ingresar menos de tres caracteres en el campo contraseña.] [El sistema permite iniciar sesión no importando si la contraseña es correcta]

- Utilizar la forma impersonal al redactar.

El contenido de los trabajos académicos y técnicos debe redactarse siempre en forma impersonal, por lo que el uso de pronombres y/o adjetivos personales tales como: "yo", "mío", "nosotros", "nuestro", etc., queda restringido.

Por ejemplo, evitar lo siguiente:

“El sistema no me permite el acceso a la página principal a pesar de que estoy introduciendo mi usuario y contraseña de manera correcta”

- Omitir frases negativas

Cuando se leen, escuchan o incluso se dicen expresiones negativas, el cerebro interpreta que algo malo ocurre. Bajo ese estado, es muy difícil que las personas estén lo suficientemente abiertas para que se produzca la comunicación de manera eficaz.

Por ejemplo:

“No se entiende la funcionalidad de personalización en el sistema, si no es útil debería eliminarse.”

- Evitar elegir las palabras por bonitas, sino por su significado.

Debe elegirse cada palabra por su significado; se sugiere evitar adjetivos elogiosos como espléndido, excelente, magistral, maravilloso, etc.

Por ejemplo:

“Sería maravilloso que se le agregaran más imágenes a la página principal, ya que esto le daría una excelente presentación”

- Redundancia.

Es el empleo de palabras innecesarias para expresar una idea o concepto.

Por ejemplo:

“El sistema no realiza la búsqueda, aún después de un largo **lapso de tiempo**” (lapso).

“Se sugiere hacer un **breve resumen** de la información presentada” (resumen).

- Leer lo escrito de forma autocrítica.

Una vez logrado un primer borrador, las adecuaciones dependen de la capacidad crítica de uno mismo. Esta tarea, sin embargo, se dificulta por la cercanía con el texto; es indispensable fortalecer la facultad de autocrítica creando distancia para leer lo escrito como si fuera de otra persona. Ayuda, en esta tarea, leer en voz alta, ya que se descubrirán mejoras de tipo: puntuación, rima, y monotonía. Las cuales, escapan a la lectura visual.

De acuerdo con la ISO 25010 un reporte de defectos se compone por los siguientes elementos (ISO, 2011):

- Identificador: Establece el orden con el cual se identificarán los defectos (ISO, 2011).
- Nombre del probador: Identifica a la persona que identificó y reportó el defecto (ISO, 2011).
- Nombre del responsable: Identifica a la persona que será asignada en resolver la incidencia (ISO, 2011).

- Título: Dentro de esta sección se coloca un resumen acerca del defecto. De tal manera, que con pocas palabras se entienda la falla encontrada. Para ello, se puede utilizar la utilización de un esquema. El cual, fue definido al principio de esta fase (ISO, 2011).
- Resumen: Presenta información que complementa el título del defecto. Dentro de este apartado se detalla de mejor manera la incidencia identificada. En caso contrario con el título de algo general, se trabaja en una redacción más específica (ISO, 2011).
- Pasos para reproducir el defecto: Establece los pasos por seguir para replicar la incidencia reportada (ISO, 2011).
- Prioridad: Para establecer la prioridad. Se utilizan los siguientes valores (ISO, 2011).
    - Alta: Indica que la incidencia debe de atenderse con sentido de urgencia, debido a que el defecto reportado perjudica en mayor grado la operación del sistema.
    - Media: Señala que la incidencia puede esperar por su resolución; sin embargo, no deja de ser importante. Ya que el defecto reportado se centra en una sección específica del sistema.
    - Baja: Marca que la incidencia no afecta en la operación del sistema, pero, es importante el defecto, ya que puede representar temas de redacción o de usabilidad del sistema.
- Información adicional: En esta sección se puede colocar información que complemente el defecto. Puede ser: una precondición que genere la falla, un usuario en específico, un sistema operativo, un tipo de navegador, un archivo de pruebas, etc. (ISO, 2011).

- Capturas de pantalla. Para esta sección será importante respaldar el defecto encontrado mediante una imagen la cual, nos permita visualizar los defectos reportados (ISO, 2011).
- Estado de la incidencia. En este punto se coloca el estado actual de la incidencia reportada. Cabe mencionar que el tratamiento de las incidencias se define en la Etapa 3 Gestión de incidencias (ISO, 2011).
- Severidad.

  Contemplar una severidad para cada incidencia permitirá enfocar esfuerzos de resolución de defectos (ISO, 2011). Cómo severidad se pueden contemplar los siguientes criterios:

  - Bloqueante: Se aplica cuando el sistema no permite continuar con el flujo que se está probando, debido a una mala implementación.
  - Crítico: Aplica cuando la funcionalidad por probar se encuentra incompleta y los requerimientos implementados son deficientes al momento de la operación.
  - Alta: Aplica cuando el sistema permite finalizar el flujo que se está probando; sin embargo, durante su ejecución se identificaron elementos con carencia de respuesta. Por ejemplo, en: Ocultamiento de menús, carga de imágenes, botones, cajas de texto, visualización de archivos, omisión de envío de correos, ordenamiento ascendente y descendente en listados, Consistencia de información, etc.
  - Media: Aplica cuando se identifica una redacción incorrecta y/o faltas de ortografía en la documentación técnica y en el sistema.

- Baja: Aplica cuando exista una recomendación de mejora en las funcionalidades que se están probando. Por ejemplo, cambiar la fuente del texto, el color de algunos elementos, incrementar el tamaño de la letra, sustituir imágenes complementarias al sistema, etc.

Los cuales, permiten conocer a detalle los defectos identificados

A continuación, se presentan los conceptos establecidos por la ISO 25010 para el reporte de defectos (ISO, 2011).

Tabla 5

Formato para el reporte de defectos

Reporte de defectos	
ID	
Nombre del probador	
Nombre del responsable	
Título	
Resumen	
Pasos para reproducir el defecto	
Prioridad	
Información adicional	
Capturas de pantalla	

Estado de la incidencia	
Severidad	

ETAPA 5. FINALIZACIÓN

Para la última fase de esta metodología se deberá comunicar al responsable del proyecto lo trabajado durante la etapa de pruebas. Para ello. Se generará un reporte que permitirá visualizar de manera detallada lo que especificó en el plan de pruebas, y el estado de las incidencias reportadas.

En este sentido se puede utilizar los siguientes elementos.

Tabla 6

Formato de finalización de pruebas

Finalización de pruebas	
Número de ciclo de pruebas	Este dato se extrae del plan de pruebas. El cual debió de haberse realizado en un principio
Objetivo de las pruebas	Este dato se extrae del plan de pruebas. El cual debió de haberse realizado en un principio
Calendario de trabajo	Este dato se extrae del plan de pruebas. El cual debió de haberse realizado en un principio
URL del sistema	Este dato se extrae del plan de pruebas. El cual debió de haberse realizado en un principio
Equipo de trabajo y tareas	Este dato se extrae del plan de pruebas. El cual debió de haberse realizado en un principio

Incidencias reportadas				
Id	Nombre del responsable	Estado	Prioridad	Severidad
Este dato se extrae de la cuarta fase, y debió de haber sido generado para poder llevar un seguimiento de las incidencias reportadas	Este dato se extrae de la cuarta fase, y debió de haber sido ingresado para poder llevar un seguimiento de las incidencias reportadas	Este dato se extrae de la cuarta fase, y debió de haber sido generado para poder llevar un seguimiento de las incidencias reportadas	Este dato se extrae de la cuarta fase, y debió de haber sido generado para poder llevar un seguimiento de las incidencias reportadas	Este dato se extrae de la cuarta fase, y debió de haber sido generado para poder llevar un seguimiento de las incidencias reportadas

CAPÍTULO 8: CASO PRÁCTICO:

La presente propuesta será llevada a cabo dentro de una pequeña oficina de servicios de cómputo ubicada en la Ciudad de México (CDMX). La cual, pertenece al sector público y por motivos de privacidad y de confidencialidad será omitida.

Actualmente, este departamento cuenta con múltiples servicios. Entre ellos destacan los siguientes: programa de formación de becarios, programas de servicio social, cursos en línea, salas de cómputo, y desarrollo de sistemas internos.

Al tener diversas actividades que involucran la utilización de equipo de cómputo. Se han visto en la necesidad de desarrollar una aplicación web que permita gestionar los bienes de la organización. Hoy en día este sistema se encuentra en una fase de iniciación. Es decir, que no todas sus funcionalidades han sido desarrolladas o se encuentran en proceso de desarrollo; sin embargo, dentro de la planeación y alcance de dicho sistema se encuentran los siguientes módulos: Usuarios, Equipo, Orden de reparación, Inventarios, y Búsqueda.

Como podemos observar en la Figura 8. Se presentan los módulos del sistema

Figura 8

Módulos del sistema

Una vez conocidos los módulos del sistema, se generó una reunión con el equipo de desarrollo para conocer los alcances de las pruebas de software. Con esto iniciamos la primera etapa de la metodología.

ETAPA 1. PLANIFICACIÓN

<table>
<tr><td colspan="2">Plan de pruebas</td></tr>
<tr><td colspan="2">El presente documento es la base para llevar a cabo las pruebas de software. Tiene como finalidad describir los elementos que se tomarán en cuenta para la ejecución del ciclo de pruebas.</td></tr>
<tr><td>Número de ciclo</td><td>#1</td></tr>
<tr><td>Objetivo de las pruebas</td><td>• Evaluar la usabilidad del módulo registrar usuario
• Evaluar la funcionalidad de los campos:
o Nombre
o Apellido paterno
o Apellido materno
o Domicilio
o Teléfono
o Celular
o RFC
o Número de cuenta</td></tr>
<tr><td>Alcance</td><td>Como alcancé se verificarán las funcionalidades previamente definidas. También, se contempla el diseño de casos de pruebas, reporte de incidencias, el formato de finalización de pruebas. Quedará fuera del alcance de las pruebas, las comprobaciones de seguridad y la corrección de incidencias. En este sentido, dentro de</td></tr>
</table>

	este primer ciclo solo se identificarán defectos. También es importante mencionar, que quedará fuera verificar la compatibilidad de la herramienta en múltiples navegadores.
Equipo de trabajo	Rol: Becario de pruebas (Remoto)
Calendario de trabajo	Fecha de inicio: 10 de agosto de 2023 Fecha de fin: 19 de agosto de 2023
Ambientes de trabajo	Equipo de trabajo: Computadora de escritorio de 64 bits con Windows 10 Home. Navegador utilizado para las pruebas: Google Chrome versión 104.0.5112.102 de 64 bits
URL del sistema o aplicación	URL del sistema: Por motivos de confidencialidad de omite la URL
Entregables	• Plan de pruebas • Diseño de casos de pruebas • Ejecución de casos de pruebas • Reporte de defectos • Informe sobre las pruebas.
Riesgos	

Riesgos	Impacto	Probabilidad
Realizar las pruebas en un ambiente con una configuración distinta a la de producción.	Alto	Baja
El servidor o equipo para las pruebas no esté configurado correctamente.	Alto	Baja
No dar valor a las observaciones realizadas por el equipo de pruebas y por tal motivo no se apliquen los cambios sugeridos.	Alta	Media
Ejecutar las pruebas en navegadores diferentes a los utilizados por el usuario final.	Medio	Bajo
Carecer de un ambiente de pruebas, por lo que se tengan	Medio	Bajo

que ejecutar en el ambiente de desarrollo.		
Tipos de pruebas por aplicar		
• Pruebas funcionales. • Pruebas No funcionales. • Pruebas Caja negra.		
Tareas		
Tareas del becario de pruebas: • Generar el plan de pruebas • Diseñar casos de prueba. • Ejecutar casos de prueba. • Crear un esquema de gestión de incidencias. • Reportar incidencias. • Generar el reporte de finalización de las pruebas.		

ETAPA 2. DISEÑO Y EJECUCIÓN

Casos de prueba campo **Nombre**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P1	Ingresar letras en minúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	abcdefghijklmnñopqrstuvwxyz	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Nombre” ingresar los caracteres: abcdefghijklmnñopqrstuvwxyz y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P2	Ingresar las vocales en minúscula con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	áéíóú	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Nombre” ingresar los caracteres: áéíóú y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P3	Ingresar letras en mayúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	ABCDEFGHIJKLMNÑOPQRSTUVWXYZ	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Nombre” ingresar los caracteres: ABCDEFGHIJKLMNÑOPQRSTUVWXYZ y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P4	Ingresar las vocales en mayúscula con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	ÁÉÍÓÚ	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Nombre” ingresar los caracteres: ÁÉÍÓÚ y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P5	Ingresar números en	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	1234567890	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Nombre” ingresar los caracteres: 1234567890 y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres numéricos	Exitoso

Casos de prueba campo **Nombre**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P6	Ingresar una combinación de mayusculas, minusculas y acentos	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Raúl	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Nombre” ingresar los caracteres: Raúl y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P7	Ingresar caracteres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	/*-+0°!"#$%&/()=?¡*¨+´}{-.,-.,	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Nombre” ingresar los caracteres: /*-+0°!"#$%&/()=?¡*¨+´}{-.,-., y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P8	Omitir ingresar información	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Sin datos	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Nombre” omitir ingresar información y dar clic al botón "Guardar" y dar clic al botón "Guardar"	El sistema debe de presentar el mensaje de obligatoriedad	Exitoso

Casos de prueba campo **Apellido paterno**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P1	Ingresar letras en minúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	abcdefghijklmnñopqrstuvwxyz	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido paterno” ingresar los caracteres: abcdefghijklmnñopqrstuvwxyz y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido

Casos de prueba campo **Apellido paterno**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P2	Ingresar las vocales en minúscula con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	áéíóú	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido paterno” ingresar los caracteres: áéíóú y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P3	Ingresar letras en mayúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	ABCDEFGHIJKLMNÑOPQRSTUVWXYZ	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido paterno” ingresar los caracteres: ABCDEFGHIJKLMNÑOPQRSTUVWXYZ y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P4	Ingresar las vocales en mayúscula con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	ÁÉÍÓÚ	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido paterno” ingresar los caracteres: ÁÉÍÓÚ y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P5	Ingresar números en	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	1234567890	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido paterno” ingresar los caracteres: 1234567890 y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres numéricos	Exitoso
P6	Ingresar una combinación de mayusculas, minusculas y acentos	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Ramírez	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido paterno” ingresar los caracteres: Ramírez y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido

Casos de prueba campo **Apellido paterno**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P7	Ingresar caracteres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	/*-+0°!"#$%&/()=?¡*¨+´}{-.,-.,	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido paterno” ingresar los caracteres: /*-+0°!"#$%&/()=?¡*¨+´}{-.,-., y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P8	Omitir ingresar información	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Sin datos	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido paterno” omitir ingresar información y dar clic al botón "Guardar" y dar clic al botón "Guardar"	El sistema debe de presentar el mensaje de obligatoriedad	Exitoso

Casos de prueba campo **Apellido materno**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P1	Ingresar letras en minúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	abcdefghijklmnñopqrstuvwxyz	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido materno” ingresar los caracteres: abcdefghijklmnñopqrstuvwxyz y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P2	Ingresar las vocales en minúscula con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	áéíóú	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido materno” ingresar los caracteres: áéíóú y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido

Casos de prueba campo **Apellido materno**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P3	Ingresar letras en mayúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	ABCDEFGHIJKLMNÑOPQRSTUVWXYZ	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido materno” ingresar los caracteres: ABCDEFGHIJKLMNÑOPQRSTUVWXYZ y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P4	Ingresar las vocales en mayúscula con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	ÁÉÍÓÚ	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido materno” ingresar los caracteres: ÁÉÍÓÚ y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P5	Ingresar números en	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	1234567890	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido materno” ingresar los caracteres: 1234567890 y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres numéricos	Exitoso
P6	Ingresar una combinación de mayusculas, minusculas y acentos	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Ramírez	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido materno” ingresar los caracteres: Ramírez y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Fallido
P7	Ingresar caracteres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	/*-+0°!'#$%&/()=?¡*¨+´}{-.,-.,	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido materno” ingresar los caracteres: /*-+0°!'#$%&/()=?¡*¨+´}{-.,-., y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Fallido

Casos de prueba campo **Apellido materno**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P8	Omitir ingresar información	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Sin datos	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Apellido materno” omitir ingresar información y dar clic al botón "Guardar"	El sistema debe de presentar el mensaje de obligatoriedad	Exitoso

Casos de prueba campo **Domicilio**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P1	Ingresar letras en minúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	abcdefghijklmnñopqrstuvwxyz	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Domicilio” ingresar los caracteres: abcdefghijklmnñopqrstuvwxyz y	El sistema debe de permitir el ingreso de los caracteres	Exitoso
P2	Ingresar las vocales en minúscula con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	áéíóú	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Domicilio” ingresar los caracteres: áéíóú y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso
P3	Ingresar letras en mayúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	ABCDEFGHIJKLMNÑOPQRSTUVWXYZ	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Domicilio” ingresar los caracteres: ABCDEFGHIJKLMNÑOPQRSTUVWXYZ y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso

Casos de prueba campo **Domicilio**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P4	Ingresar las vocales en mayúscula con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	ÁÉÍÓÚ	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Domicilio” ingresar los caracteres: ÁÉÍÓÚ y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso
P5	Ingresar números	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	1234567890	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Domicilio” ingresar los caracteres: 1234567890 y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso
P6	Ingresar caracteres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	# . ,	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Domicilio” ingresar los caracteres: # . , y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso
P7	Ingresar una combinación de mayúsculas, minúsculas con números y caracateres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Pedregal de Sto. Domingo, Papalotl #54	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Domicilio” ingresar los caracteres: Domingo, Papalotl #54 y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso

Casos de prueba campo **Domicilio**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P8	Ingresar caracteres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	*/°"$%&()=*+´{}[]¿?	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Domicilio” ingresar los caracteres: */°"$%&()=*+´{}[]¿? y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Fallido
P9	Omitir ingresar información	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Sin datos	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Domicilio” omitir ingresar información y dar clic al botón "Guardar" y dar clic al botón "Guardar"	El sistema debe de presentar el mensaje de obligatoriedad	Exitoso

Casos de prueba campo **Teléfono**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P1	Ingresar letras en minúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	abcdefghijklmnñopqrstuvwxyz	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Teléfono” ingresar los caracteres: abcdefghijklmnñopqrstuvwxyz y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P2	Ingresar números	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	1234567890	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Teléfono” ingresar los caracteres: 1234567890 y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso

Casos de prueba campo **Teléfono**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P3	Ingresar caracteres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	*/°"$%&()=*+´{}[]¿?	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Teléfono” ingresar los caracteres: */°"$%&()=*+´{}[]¿? y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P4	Omitir ingresar información	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Sin datos	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Teléfono” omitir ingresar información y dar clic al botón "Guardar"	El sistema debe de presentar el mensaje de obligatoriedad	Exitoso

Casos de prueba campo **Célular**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P1	Ingresar letras en minúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	abcdefghijklmnñopqrstuvwxyz	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Célular” ingresar los caracteres: abcdefghijklmnñopqrstuvwxyz y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P2	Ingresar números	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	1234567890	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Célular” ingresar los caracteres: 1234567890 y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso

Casos de prueba campo **Célular**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P3	Ingresar caracteres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	*/°"$%&()=*+´{}[]¿?	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Célular” ingresar los caracteres: */°"$%&()=*+´{}[]¿? y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P4	Omitir ingresar información	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Sin datos	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Célular” omitir ingresar información y dar clic al botón "Guardar"	El sistema debe de presentar el mensaje de obligatoriedad	Exitoso

Casos de prueba campo **Número de cuenta**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P1	Ingresar letras en minúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	abcdefghijklmnñopqrstuvwxyz	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Número de cuenta” ingresar los caracteres: abcdefghijklmnñopqrstuvwxyz y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P2	Ingresar números	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	1234567890	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Número de cuenta” ingresar los caracteres: 1234567890 y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso

Casos de prueba campo **Número de cuenta**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P3	Ingresar caracteres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	*/°"$%&()=*+´{}[]¿?	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Número de cuenta” ingresar los caracteres: */°"$%&()=*+´{}[]¿? y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P4	Omitir ingresar información	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Sin datos	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Número de cuenta” omitir ingresar información y dar clic al botón "Guardar" y dar clic al botón "Guardar"	El sistema debe de presentar el mensaje de obligatoriedad	Exitoso

Casos de prueba campo **RFC**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P1	Ingresar letras en minúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	abcdefghijklmnñopqrstuvwxyz	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “RFC” ingresar los caracteres: abcdefghijklmnñopqrstuvwxyz y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Fallido
P2	Ingresar las vocales en minúscula con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	áéíóú	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “RFC" ingresar los caracteres: áéíóú y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Fallido

Casos de prueba campo **RFC**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P3	Ingresar letras en mayúscula	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	ABCDEFGHIJKLMNÑOPQRSTUVWXYZ	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “RFC” ingresar los caracteres: ABCDEFGHIJKLMNÑOPQRSTUVWXYZ y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso
P4	Ingresar las vocales en mayúscula con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	ÁÉÍÓÚ	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “RFC” ingresar los caracteres: ÁÉÍÓÚ y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Fallido
P5	Ingresar números en	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	1234567890	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “RFC” ingresar los caracteres: 1234567890 y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres numéricos	Exitoso
P6	Ingresar caracteres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	/*-+0°!"#$%&/()=?¡*¨+´}{-.,-.,	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “RFC” ingresar los caracteres: /*-+0°!"#$%&/()=?¡*¨+´}{-.,-., y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Fallido
P7	Omitir ingresar información	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	Sin datos	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “RFC” omitir ingresar información y dar clic al botón "Guardar"	El sistema debe de presentar el mensaje de obligatoriedad	Exitoso

Casos de prueba campo **Correo**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P1	Ingresar la estructura del correo completa con letras minúsculas	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	cuenta@dominio.dominio	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: cuenta@dominio.dominio y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso
P2	Ingresar la estructura del correo completa con letras minúsculas y números	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	cuenta123@dominio.dominio	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: cuenta123@dominio.dominio y dar clic al botón "Guardar"	El sistema debe de permitir el ingreso de los caracteres	Exitoso
P3	Ingresar la estructura del correo completa solamente con caracteres especiales	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	?!+/$%&@dominio.dominio	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: ?!+/$%&@dominio.dominio y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Fallido
P4	Ingresar la estructura del correo sin dominio	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	cuenta@	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: cuenta@ y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P5	Ingresar la estructura del correo sin el punto final de separación	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	cuenta@dominio	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: cuenta@dominio y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Fallido

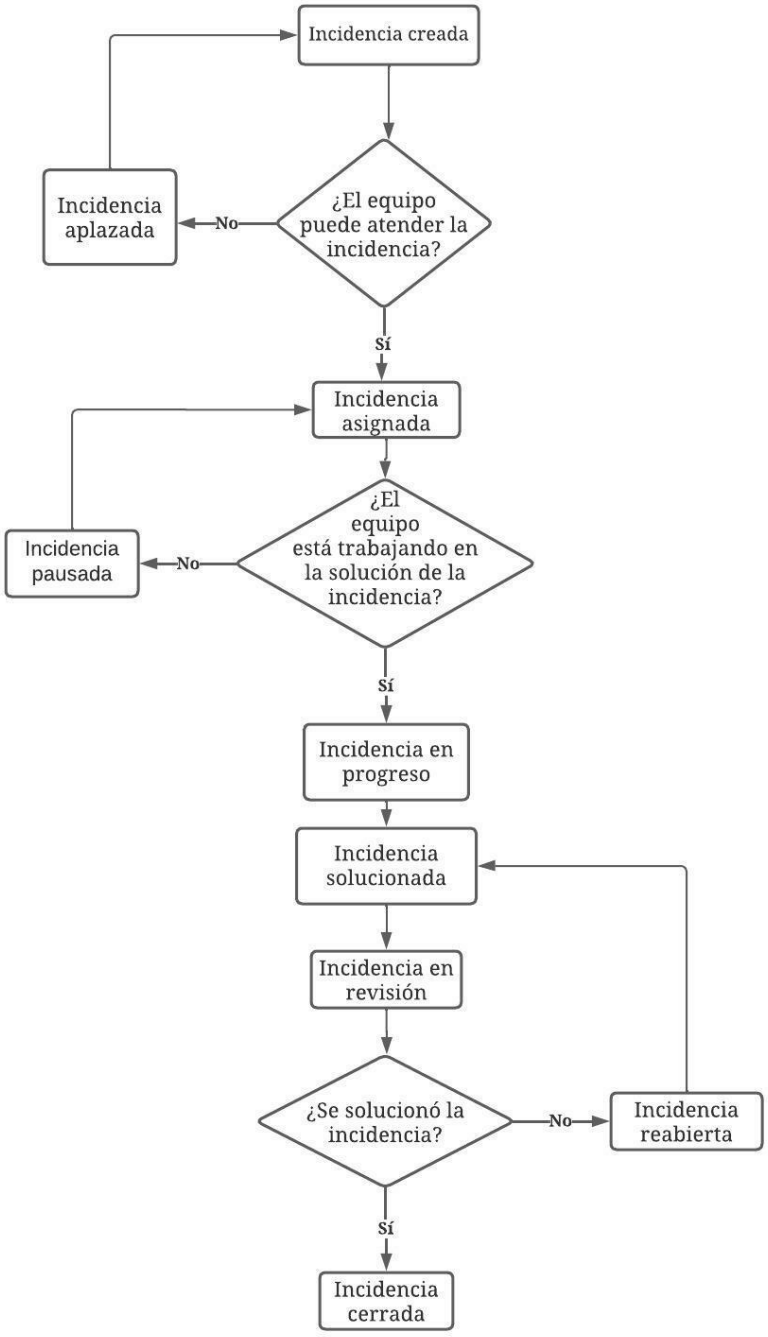
Casos de prueba campo **Correo**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P6	Ingresar la estructura del correo sin el segundo nombre del dominio	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	cuenta@dominio.	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: cuenta@dominio. y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P7	Ingresar la estructura del correo con doble punto	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	cuenta@dominio..dominio	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: cuenta@dominio..dominio y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P8	Ingresar la estructura del correo sin el nombre del inicio	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	@dominio.domino	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: @dominio.domino y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P9	Ingresar la estructura del correo con triple nombre de dominio	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	cuenta@dominio.dominio.dominio	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: cuenta@dominio.dominio.dominio y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Fallido
P10	Ingresar la estructura del correo con doble arroba	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	cuent@s123@dominio.dominio	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: cuent@s123@dominio.dominio y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso

Casos de prueba campo **Correo**							
Id	Descripción / Resumen	Prioridad	Precondiciones	Datos de entrada	Pasos	Resultado esperado	Resultado obtenido
P11	Ingresar la estructura del correo sin arroba	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	cuentadominio.dominio	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: cuentadominio.dominio y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso
P12	Ingresar la estructura del correo con acento	Media	Ingresar con un usuario administrador para poder hacer el registro de un nuevo usuario	electrónico@dominio.dominio	1. Ingresar al sistema con un usuario administrador 2. Dar clic a la opción “Usuarios” 3. Dar clic al botón “Registrar usuario” 4. En el campo “Correo” ingresar los caracteres: cuenta@dominio.dominio y dar clic al botón "Guardar"	El sistema debe de omitir el ingreso de los caracteres	Exitoso

ETAPA 3. GESTIÓN

Ejemplo 7

El siguiente diagrama es una representación de la gestión de incidencias que será llevado a cabo durante la etapa de pruebas.



A continuación, se describe cada uno de los estados:

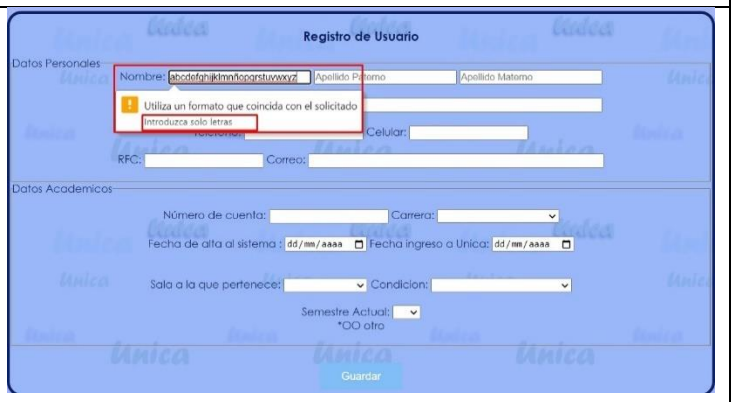
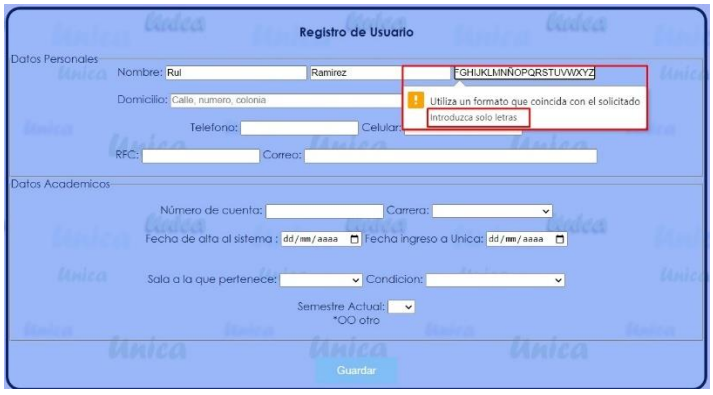
- Incidencia creada: Será utilizado una vez reportado un defecto, y es colocado por la persona que lo identificó.
- Incidencia asignada: Representa que el defecto ha sido asignado a un integrante del equipo de trabajo para su solución, y es colocado por la persona que reportó el defecto.
- Incidencia aplazada. Indica que la incidencia reportada presenta un grado mayor de complejidad al esperado, y su solución está relacionada con servicios ajenos a los del equipo de trabajo. Por ejemplo, un proveedor de servicios en nube, alojamiento, solicitudes a bases de datos externas a la compañía, etc. Es importante indicar que este estado será colocado por el responsable en resolver la incidencia.
- Incidencia pausada. Refleja que la incidencia reportada está siendo atendida, pero fue detenida debido a que se encontraron elementos técnicos que deben de ser revisados por los integrantes del equipo. Estos elementos pueden se: Rendimiento en los servidores, procesamiento de datos, requerimientos no validados por el equipo, etc. Cabe mencionar que este estado será colocado por el responsable en solucionar la incidencia.
- Incidencia en progreso: Este estado hace referencia a que la incidencia está en progreso de solucionarse y es asignado por el responsable en solventar la incidencia.
- Incidencia solucionada: Indica que la incidencia ha sido corregida en su totalidad y es asignado por el responsable en atender la incidencia.

- Incidencia en revisión: Señala que la incidencia está siendo revisada y que el defecto reportado ya no se presenta. Este estado es asignado por la persona que reportó el defecto.
- Incidencia reabierta: Especifica que la incidencia no fue corregida y que actualmente se sigue presentando. Este estado es asignado por la persona que lo reportó el defecto.
- Incidencia cerrada: Indica que la incidencia está resuelta en su totalidad y es asignado por la persona que identificó el defecto.

ETAPA 4. REPORTE

Reporte de defectos	
ID	#Incidencia1
Nombre del probador	Raúl
Nombre del responsable	Integrante de desarrollo
Título	En los campos “Nombre”, “Apellido paterno” y “Apellido materno” al ingresar letras mayúsculas, minúsculas, con acento y sin acento, de manera consecutiva, el sistema le indica al usuario “Utiliza un formato que coincida con el solicitado”.
Resumen	En los campos “Nombre”, “Apellido paterno” y “Apellido materno” al ingresar letras mayúsculas, minúsculas, con acento y sin acento, de manera consecutiva, el sistema le indica al usuario “Utiliza un formato que coincida con el solicitado”. Al revisar el

	código fuente de los campos, se identificó que el atributo “Pattern” fue definido como un elemento que acepta letras consecutivas. Debido a esto se presenta el error. Además, que falta especificar en la validación las letras con acento, tanto mayúsculas como minúsculas. Con la finalidad de contemplar dentro del registro el formato general de los nombres y apellidos. Se recomienda que el sistema acepte una combinación de letras mayúsculas, minúsculas con acento y sin acento.
Pasos para reproducir el defecto	1. Ingresar al usuario con un perfil de administrador. 2. Dar clic al botón “Registrar Usuario”. 3. En el campo “Nombre” ingresar la cadena de caracteres: abcdefghijklmnñopqrstuvwxyz 4. Dar clic al botón “Guardar”. 5. Repetir el paso 3 y 4 por separado para cada una de las siguientes cadenas: áéíóú ABCDEFGHIJKLMNÑOPQRSTUVWXYZ ÁÉÍÓÚ Raúl o Ramírez (En caso de los apellidos) 6. Replicar los pasos 3, 4, y 5 para los campos “Apellido paterno”, y “Apellido materno”

	7. Observar que el sistema para cada campo presenta el mensaje “Utiliza un formato que coincida con el solicitado”.
Prioridad	Alta
Información adicional	Ninguna
Capturas de pantalla	  

	
Estado de la incidencia	Incidencia creada
Severidad	Crítico

Reporte de defectos	
ID	#Incidencia2
Nombre del probador	Raúl
Nombre del responsable	Integrante de desarrollo
Título	En el campo “Domicilio” al ingresar los caracteres: */°"$%&()=*+´{}[]¿?. El sistema omite la validación y permite guardar el elemento ingresado.
Resumen	En el campo “Domicilio” al ingresar los caracteres: */°"$%&()=*+´{}[]¿?. El sistema omite la validación y permite guardar el elemento ingresado. Al revisar el código fuente del campo, se identificó que el atributo “Pattern” fue omitido, colocando solo el elemento de obligatoriedad. Se recomienda que el sistema tenga un elemento de validación (Pattern) que esté especificado con los caracteres que acepta.
Pasos para reproducir el defecto	1. Ingresar al usuario con un perfil de administrador.

	2. Dar clic al botón “Registrar Usuario”. 3. En el campo “Domicilio” ingresar la cadena de caracteres: */°"$%&()=*+´{}[]¿? 4. Dar clic al botón “Guardar”. 5. Observar que el sistema permite guardar los caracteres ingresados.
Prioridad	Alta
Información adicional	Ninguna
Capturas de pantalla	
Estado de la incidencia	Incidencia creada
Severidad	Crítico

Reporte de defectos	
ID	#Incidencia3
Nombre del probador	Raúl
Nombre del responsable	Integrante de desarrollo

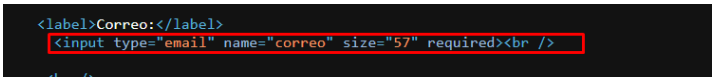
Título	En el campo “RFC” al ingresar letras minúsculas, con acento y caracteres especiales. El sistema omite la validación y permite guardar los elementos ingresados.
Resumen	En el campo “RFC” al ingresar letras minúsculas, con acento y caracteres especiales. El sistema omite la validación y permite guardar los elementos ingresados. De acuerdo con la estructura del “RFC”, esta se compone de letras mayúsculas y números. Omitiendo caracteres especiales. Al revisar el código fuente del campo, se identificó que el atributo “Pattern” fue omitido, colocando solo el elemento de obligatoriedad. Se recomienda que el sistema permita guardar el “RFC” con su nomenclatura habitual y se agregue el elemento “Pattern” con las validaciones correspondientes.
Pasos para reproducir el defecto	1. Ingresar al usuario con un perfil de administrador. 2. Dar clic al botón “Registrar Usuario”. 3. En el campo “Nombre” ingresar la cadena de caracteres: abcdefghijklmnñopqrstuvwxyz 4. Dar clic al botón “Guardar”. 5. Repetir el paso 3 y 4 por separado para cada una de las siguientes cadenas: aéíóú

	ÁÉÍÓÚ /*-+0°!"#$%&/()=?¡*¨+´}{-.,-., 6. Observar que el sistema permite el guardado de las cadenas anteriormente mencionadas.
Prioridad	Alta
Información adicional	Ninguna
Capturas de pantalla	  

	 <label>Correo:</label> <input type="email" name="correo" size="57" required>
Estado de la incidencia	Incidencia creada
Severidad	Crítico

Reporte de defectos	
ID	#Incidencia4
Nombre del probador	Raúl
Nombre del responsable	Integrante de desarrollo
Título	En el campo “Correo” al ingresar caracteres especiales, y una nomenclatura distinta. El sistema omite la validación y permite guardar los elementos ingresados.
Resumen	En el campo “Correo” al ingresar caracteres especiales, y una nomenclatura distinta. El sistema omite la validación y permite guardar los elementos ingresados. De acuerdo con la estructura de un correo electrónico.

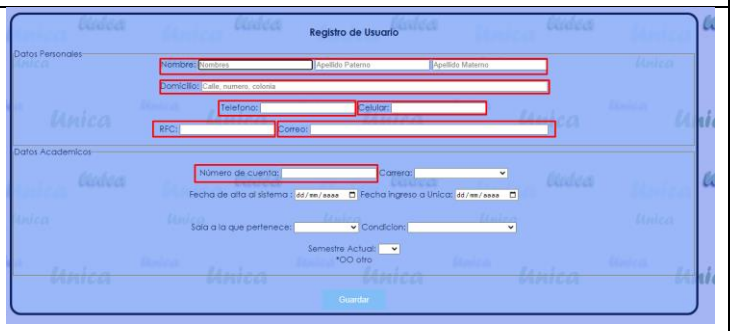
	Esta se compone de letras, números y guión bajo. Al revisar el código fuente del campo, se identificó que el atributo “Pattern” fue omitido, colocando solo el elemento de obligatoriedad. Se recomienda que el sistema permita guardar el correo electrónico con su nomenclatura habitual y se agregue el elemento “Pattern” con las validaciones correspondientes.
Pasos para reproducir el defecto	1. Ingresar al usuario con un perfil de administrador. 2. Dar clic al botón “Registrar Usuario”. 3. En el campo “Correo” ingresar la cadena de caracteres: ?!+/$%&@dominio.dominio 4. Dar clic al botón “Guardar”. 5. Repetir el paso 3 y 4 por separado para cada una de las siguientes cadenas: cuenta@dominio cuenta@dominio.dominio.dominio 6. Observar que el sistema permite el guardado de los elementos ingresados.
Prioridad	Alta
Información adicional	Ninguna

Capturas de pantalla	   
Estado de la incidencia	Incidencia creada
Severidad	Crítico

Reporte de defectos	
ID	#Incidencia5
Nombre del probador	Raúl
Nombre del responsable	Integrante de desarrollo
Título	Al navegar dentro del módulo “Registrar usuario”. El sistema presenta algunas palabras sin acento.
Resumen	Al navegar dentro del módulo “Registrar usuario”. El sistema presenta las palabras “Telefono”, “Academicos”, y “Condicion” sin acento. Se recomienda que el sistema presente estas palabras con acento, debido a que esto mejora la comprensión del sistema.
Pasos para reproducir el defecto	1. Ingresar al usuario con un perfil de administrador. 2. Dar clic al botón “Registrar Usuario”. 3. Observar que las palabras: “Telefono”, “Academicos”, y “Condicion” carecen de acento.
Prioridad	Baja
Información adicional	Ninguna

Capturas de pantalla	
Estado de la incidencia	Incidencia creada
Severidad	Media

Reporte de defectos	
ID	#Incidencia6
Nombre del probador	Raúl
Nombre del responsable	Integrante de desarrollo
Título	Al navegar dentro del módulo “Registrar usuario”. El sistema omite indicar la cantidad de caracteres que se pueden ingresar en los campos de tipo “Texto”.
Resumen	Al navegar dentro del módulo “Registrar usuario”. El sistema omite indicar la cantidad de caracteres que puede ingresar el usuario en los siguientes campos: "Nombre", "Apellido Paterno", "Apellido Materno", "Domicilio", "Telefono", "Celular", "RFC", "Correo", y "Número de cuenta". Se recomienda que el sistema presente la cantidad de caracteres que el usuario puede ingresar en dichos campos.

Pasos para reproducir el defecto	1. Ingresar al usuario con un perfil de administrador. 2. Dar clic al botón “Registrar Usuario”. 3. Observar que los campos: "Nombre", "Apellido Paterno", "Apellido Materno", "Domicilio", "Telefono", "Celular", "RFC", "Correo", y "Número de cuenta". Carecen de un contador de caracteres.
Prioridad	Baja
Información adicional	Ninguna
Capturas de pantalla	
Estado de la incidencia	Incidencia creada
Severidad	Media

Reporte de defectos	
ID	#Incidencia7
Nombre del probador	Raúl
Nombre del responsable	Integrante de desarrollo
Título	Al navegar dentro del módulo “Registrar usuario”, al no tener seleccionada ninguna opción en las listas desplegables. El sistema presenta en blanco la casilla.
Resumen	Al navegar dentro del módulo “Registrar usuario”. El sistema presenta los campos: “Carrera”, “Sala a la que pertenece”, “Condición”, y “Semestre actual”. Estos hacen uso de listas desplegables; sin embargo, al no tener seleccionada ninguna opción. El sistema presenta en blanco la casilla. Se recomienda que al no tener seleccionada ninguna opción dentro de las listas desplegables. El sistema muestre el texto “Selecciona una opción” o una frase similar. De esta manera el usuario entenderá que este campo no puede quedar vacío aún contando con la validación.
Pasos para reproducir el defecto	1. Ingresar al usuario con un perfil de administrador. 2. Dar clic al botón “Registrar Usuario”. 3. Observar que los campos: “Carrera”, “Sala a la que pertenece”, “Condición”, y “Semestre

	actual”. Carecen de una frase que indique la selección de alguna opción
Prioridad	Baja
Información adicional	Ninguna
Capturas de pantalla	
Estado de la incidencia	Incidencia creada
Severidad	Media

ETAPA 5. FINALIZACIÓN

Finalización de pruebas	
Número de ciclo de pruebas	#1
Objetivo de las pruebas	Evaluar la usabilidad del módulo registrar usuario Evaluar la funcionalidad del módulo registrar usuario, específicamente los campos: o Nombre o Apellido paterno o Apellido materno o Domicilio o Teléfono o Celular o RFC o Número de cuenta
Calendario de trabajo	Fecha de inicio: 10 de agosto de 2023 Fecha de fin: 19 de agosto de 2023
URL del sistema	Por motivos de confidencialidad se omite la URL
Equipo de trabajo y tareas	Rol: Becario de pruebas (Remoto) Tareas: • Generar el plan de pruebas • Diseñar casos de prueba.

		• Ejecutar casos de prueba. • Crear un esquema de gestión de incidencias • Generar el reporte de finalización		
Incidencias reportadas				
Id	Nombre del responsable	Estado	Prioridad	Severidad
#Incidencia1	Integrante de desarrollo	Incidencia creada	Alta	Crítico
#Incidencia2	Integrante de desarrollo	Incidencia creada	Alta	Crítico
#Incidencia3	Integrante de desarrollo	Incidencia creada	Alta	Crítico
#Incidencia4	Integrante de desarrollo	Incidencia creada	Alta	Crítico
#Incidencia5	Integrante de desarrollo	Incidencia creada	Baja	Media
#Incidencia6	Integrante de desarrollo	Incidencia creada	Baja	Media
#Incidencia7	Integrante de desarrollo	Incidencia creada	Baja	Media

RESULTADOS

RESULTADOS DEL CASO PRÁCTICO

Para medir la calidad del módulo “Registrar usuario”, se valoró la usabilidad de este, y la funcionalidad de los campos: "Apellido paterno", "Apellido materno", "Domicilio", "Teléfono", "Celular", "RFC", y "Número de cuenta". Para ello, se ejecutaron 64 casos de pruebas de los cuales se obtuvieron los siguientes resultados.

Como podemos observar en la Figura 9. Se presentan la cantidad de incidencias encontradas en el módulo “Registrar usuario” con severidad crítica y media.

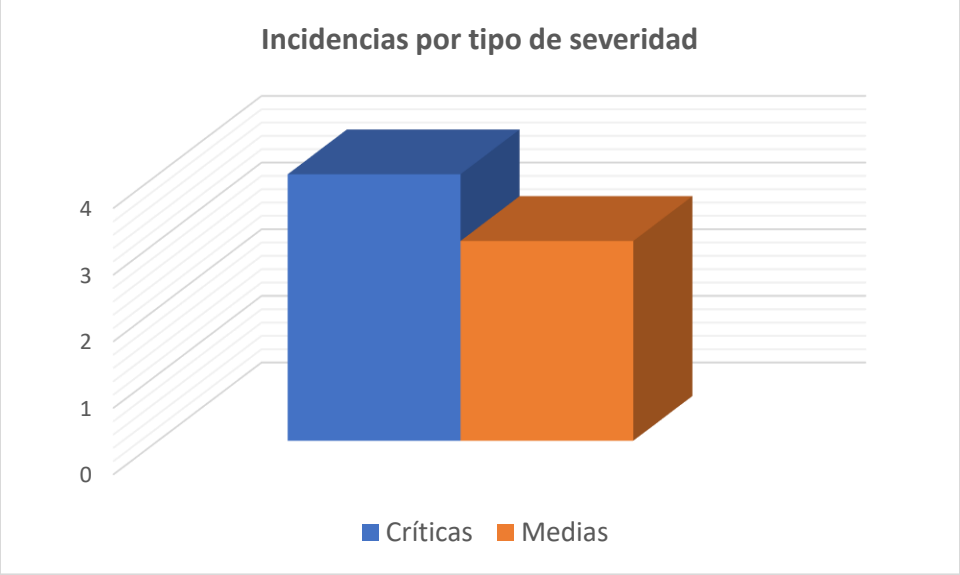


Figura 9

Gráfica de incidencias por tipo de severidad

Durante la ejecución de los casos de pruebas se identificaron 4 incidencias críticas relacionadas a la validación de los campos. Las cuales, impactan directamente en la funcionalidad del sistema y para que este funcione de manera correcta deberán de ser atendidas. Situación que no había sido identificada por el equipo de desarrollo. Como se menciona en el enfoque teórico, una vez solucionadas estas incidencias se pueden ejecutar los casos de pruebas para validar que estos defectos ya no se presenten. A esto se le conoce como pruebas de confirmación.

Es importante mencionar que el diseño de pruebas cumplió su objetivo. El cual fue mostrar la presencia de defectos, sin embargo, la reutilización de los casos de prueba dependerá de la composición del sistema. Podrán existir módulos que ocupen campos de tipo texto o numéricos, y si estos se acoplan a los casos de prueba previamente generados podrán ser utilizados, pero sí su contexto es diferentes se deberán de diseñar nuevos casos de pruebas. Que como su definición lo indica, deberán de encontrar defectos.

Para el caso de las pruebas de usabilidad, se identificaron 3 incidencias de severidad media, se catalogaron con esta severidad ya que los defectos están enfocados a la manera en que el usuario entiende el sistema, y no a la operación del sistema, sin embargo, para tener a una comprensión más precisa y menos complicada deberán de ser atendidos. Una vez corregidos, solo bastará con leer la redacción de estos y comprobar que se hayan solucionado.

RESULTADOS RELACIONADOS AL MARCO DE INVESTIGACIÓN

DE ACUERDO CON EL PROBLEMA DE INVESTIGACIÓN

"El no tener un proceso de pruebas de software basado en un enfoque teórico y práctico genera procedimientos ambiguos o confusos dentro de las áreas de desarrollo".

Y finalizado el caso práctico de la presente metodología, relumbraron algunos resultados interesantes, los cuales se mencionan a continuación.

En primer lugar, el departamento de desarrollo quedó conforme con el trabajo realizado. Explicando que la implementación de un proceso de pruebas permitió mejorar la calidad del proyecto. Con relación a sus procedimientos de calidad, relumbró el problema de investigación, mencionaron que el omitir las validaciones en los campos del formulario fue un error por falta de atención. Esto debido a los cortos periodos de tiempo de desarrollo. Situación que se presenta de manera constante durante los proyectos. También, señalaron que todo su personal está compuesto por desarrolladores, y ellos son los encargados de llevar a cabo las pruebas. Estas, son realizadas de manera esporádica, contemplando el funcionamiento general del sistema, y durante un breve periodo de tiempo. Esto es una manera de generar calidad, pero se omiten etapas importantes como lo son el análisis, y diseño de pruebas, las cuales, se enfocan en evaluar más de un escenario dentro de una funcionalidad.

CONFORME AL OBJETIVO DE LA INVESTIGACIÓN

"Generar una propuesta técnica acerca de un ciclo de pruebas de software. Que permita establecer un enfoque teórico, etapas, y actividades para su gestión. Podrá ser llevada a cabo, por las áreas de desarrollo que necesiten implementar un proceso de calidad de software."

Este se cumplió de manera exitosa. Ya que, la presente propuesta fue aprobada y puesta en marcha en una organización pública dedicada al desarrollo. Generando en ella, una estructura dividida en fases de desarrollo. En donde las pruebas de software se integran de manera constante, con base en elementos de la presente propuesta. Esto permitirá hacer revisiones periódicas a los sistemas desarrollados, con la finalidad de no esperar a encontrar defectos en las últimas etapas, sino en una fase temprana, en donde la solución a estos se de en momentos adecuados, y sin la necesidad de invertir tiempo del personal a cargo.

CON BASE EN LA PREGUNTA DE INVESTIGACIÓN

La falta de atención hacia las pruebas de software ¿Está relacionada con el desconocimiento del enfoque teórico y práctico de las mismas?

La respuesta es afirmativa, su resolución empieza a deslumbrar durante el análisis del estado del arte, y termina de materializarse al momento de llevar a cabo el caso práctico. Dando a notar, que para realizar un proceso eficiente de pruebas de software se debe de conocer su marco teórico, y el aporte que se ha realizado a través del tiempo.

Omitir el estudio teórico de las pruebas nos enseña que más allá de comprender que es el diseño, ejecución, y reporte de pruebas. También se ven afectados elementos organizacionales, como son la comunicación de los defectos. Elemento que puede generar distanciamiento entre los colaboradores. El enfoque teórico, también nos permite conocer los

principios que envuelven a las pruebas de software, nociones que ayudan a entender de mejor manera ¿Qué significan las pruebas de software?

La omisión de estos conceptos primordiales, también impactan en la calidad de los sistemas. Ya que, al no comprender ¿Que se está realizando? y ¿Por qué se realiza? Genera deducciones personales que derivan en pruebas sin sentido, repetitivas, empíricas, y sin valor para los sistemas de información.

Conforme al enfoque práctico. Practicar la realización de pruebas de software como anteriormente se mencionó debe partir del enfoque teórico. Ya que actualmente se han desarrollado diferentes tipos y técnicas de prueba, que pueden ayudar a la comprensión de estas. Ponerlas en práctica permitirá atacar los sistemas de múltiples maneras, generando distintos tipos de fallas. Sin embargo, si estas no son practicadas, la persona encargada se limitará a probar de la manera más básica, diseñando pruebas sin valor para la organización u omitiendo incidencias que pueden llegar a etapas finales.

CON RESPECTO A LA HIPÓTESIS DE ESTA INVESTIGACIÓN

“Fundamentar, enseñar, y establecer una metodología relacionada a las pruebas de software, permitirá su adopción dentro de las áreas de desarrollo, transformándose en una etapa primordial para el desarrollo de sistemas”.

Resultó verdadera. Esto, debido a que, al fundamentar la esencia de las pruebas de software, se comprende porque está actividad debe de ser esencial para las áreas de desarrollo.

La enseñanza de estas genera interés en los involucrados en el desarrollo de software, ya que genera conocimiento e ideas del porque comúnmente pueden fallar los sistemas de información.

Establecer una metodología que presente cómo llevar un proceso de pruebas de software, que comience desde una planeación hasta su finalización crea confianza en los involucrados, y permite su adopción en distintos proyectos de desarrollo.

CONCLUSIONES

- Actualmente, el desarrollo de software es una actividad que ha ido creciendo conforme ha avanzado el uso de la tecnología, pero aún encontramos sistemas o aplicaciones con fallas importantes. Siendo este un problema real de las empresas. El cual, pone en tela de juicio su reputación, provocando pérdidas económicas y proyectos futuros.
- Omitir los antecedentes de las pruebas de software como lo son: la ingeniería de software, las metodologías tradicionales y ágiles, Normas ISO, y otros estándares, impactan en la calidad de los sistemas. Ya que, al no comprender ¿Que se está realizando? y ¿Por qué se realiza? Se generan deducciones personales que derivan en pruebas sin sentido, repetitivas, empíricas, y sin valor para los sistemas de información.
- La realización de pruebas de software debe partir del enfoque teórico. Ya que actualmente se han desarrollado diferentes tipos y técnicas de prueba, que pueden ayudar a la comprensión de estas. Ponerlas en práctica permitirá atacar los sistemas de múltiples maneras, generando distintos tipos de fallas.
- Estudiar las diferentes investigaciones que se han realizado acerca de las pruebas de software permite descubrir sus diferentes enfoques (Educativos, sociales, económicos, organizacionales, etc.). Con ello, podemos realizar nuevas aportaciones que permitan reconocer su importancia y su valor en el desarrollo de software.

- La ejecución de pruebas: disminuye considerablemente las incidencias reportadas por los usuarios, aumenta la detección de defectos en etapas tempranas, permite obtener calidad en los sistemas, y generan confianza en el cliente final.

REFERENCIAS

Alegsa, L. (2002, 31 diciembre). *Definición de Sentencia de programación*. Alegsa.com.ar. Recuperado 19 de julio de 2022, de https://www.alegsa.com.ar/Dic/sentencia_de_programacion.php

Artehistoria. (s. f.). *Misiles SS-1b Scud A*. Recuperado 1 de febrero de 2021, de https://www.artehistoria.com/es/fuente/misiles-ss-1b-scud

BBC News Mundo. (2017, 8 octubre). *Sesgo confirmatorio: ¿cómo hacer que la gente vea la evidencia y no lo que confirma sus creencias y prejuicios?* Recuperado 18 de junio de 2022, de https://www.bbc.com/mundo/vert-fut-38887920

Blanquicett, L. A., Bonfante, M. C., & Acosta-Solano, J. (2018). Prácticas de Pruebas desde la Industria de Software. La Plataforma ASISTO como Caso de Estudio. *Información tecnológica*, *29*(1), 11–18. https://doi.org/10.4067/s0718-07642018000100011

Cadavid, A. N. (2013). Revisión de metodologías ágiles para el desarrollo de software. *Prospectiva*, *11*(2), 30–39. https://doi.org/10.15665/rp.v11i2.36

Capote García, T., Brito Rivero, Y., Yzquierdo Herrera, R., & Febles Estrada, A. (2014). Estrategia para desarrollar la perspectiva Procesos internos en un laboratorio de pruebas de software. *Revista Cubana de Ciencias Informáticas*, *8*(4), 145–156. https://rcci.uci.cu/index.php?journal=rcci&page=article&op=view&path[]=735&path[]=297

García, J. (2020, 13 enero). *El día que un misil mató a 28 soldados porque el sistema de defensa antimisiles ignoró un error de. . .* Xataka. Recuperado 1 de febrero de 2021, de https://www.xataka.com/historia-tecnologica/dia-que-misil-mato-a-28-soldados-porque-sistema-defensa-antimisiles-ignoro-error-0-000000095-segundos

González, E. (1996, 24 julio). *El Ariane 5 explota por un fallo de software afirma el informe oficial*. El País. Recuperado 1 de febrero de 2021, de https://elpais.com/diario/1996/07/24/sociedad/838159214_850215.html

IEC. (s. f.). *What we do*. Recuperado 1 de marzo de 2021, de https://iec.ch/what-we-do

IEEE. (2021, diciembre). *IEEE at a Glance*. Recuperado 1 de marzo de 2021, de https://www.ieee.org/about/at-a-glance.html

Instituto Politécnico Nacional, & Espinosa García, G. (2016). *Desarrollo de un modelo de pruebas y calidad de software para la empresa seguros ATLAS S.A* (TFG). https://tesis.ipn.mx/bitstream/handle/123456789/20210/Gabriela%20Espinosa%20Garcia.pdf?sequence=1&isAllowed=y

ISO. (2011, marzo). *ISO/CEI 25010:2011*. Recuperado 3 de septiembre de 2022, de https://www.iso.org/standard/35733.html

ISO. (2017, febrero). *ISO/CEI 20246:2017*. Recuperado 3 de septiembre de 2021, de https://www.iso.org/standard/67407.html

ISO. (2021, enero). *ISO/IEC/IEEE 29119–1:2022*. Recuperado 3 de septiembre de 2021, de https://www.iso.org/standard/81291.html

Isotools. (2022, 16 marzo). *¿Qué son las normas ISO y cuál es su finalidad?* Software ISO. Recuperado 1 de marzo de 2021, de https://www.isotools.org/2015/03/19/que-son-las-normas-iso-y-cual-es-su-finalidad/

ISTQB. (s. f.). *What we do*. Recuperado 1 de abril de 2021, de https://www.istqb.org/about-us/what-we-do/

ISTQB. (2018, abril). *Programa de Estudios de Probador Certificado de Nivel Básico*. https://isqi.org/es/index.php?controller=attachment&id_attachment=21

Jiménez, M. (1995, 26 enero). *El fallo del Pentium provoca unas pérdidas a Intel de 63.000 millones*. El País. Recuperado 1 de febrero de 2021, de https://elpais.com/diario/1995/01/26/economia/791074827_850215.html

Kaner, C., Padmanabhan, S., & Hoffman, D. (2013). *The Domain Testing Workbook* (1.ª ed.). Amsterdam University Press.

Mascheroni, M. A., Greiner, C. L., Dapozo, G. N., & Estayno, M. G. (2014). Ingeniería de Usabilidad. Una Propuesta Tecnológica para Contribuir a la Evaluación de la Usabilidad del Software. *Revista Latinoamericana de Ingenieria de Software*, *1*(4), 125. https://doi.org/10.18294/relais.2013.125-134

Mera Paz, J. (2016). Análisis del proceso de pruebas de calidad de software. *Ingeniería Solidaria*, *12*(20), 163–176. https://doi.org/10.16925/in.v12i20.1482

Ministerio De Defensa Del Gobierno de España. (s. f.). *Sistema de misiles Patriot - Ejército de tierra*. Recuperado 1 de febrero de 2021, de https://ejercito.defensa.gob.es/materiales/artilleria_antiaerea/PATRIOT.html

Pantaleo, G., & Rinaudo, L. (2015). *Ingeniería de Software* (1.ª ed.) [Libro electrónico]. Alfaomega. Recuperado 12 de marzo de 2021, de https://unam-bibliotecasdigitales-com.pbidi.unam.mx:2443/read/9786076222379/back

Ramirez G., I. (2020, 6 noviembre). *Manifiesto Agile*. iPMOGuide. Recuperado 12 de marzo de 2021, de https://ipmoguide.com/manifiesto-agile/

Rocha, C. I. M. (2016). *Metodología de la investigación* (1.ª ed.). Oxford University Press.

Roche, J. (2018, 12 diciembre). *Historia del movimiento Agile*. Deloitte España. Recuperado 12 de marzo de 2021, de

https://www2.deloitte.com/es/es/pages/technology/articles/historia-movimiento-agile.html

Roger S. Pressman. (2010). *Ingeniería del software. Un enfoque práctico* (Séptima edición). McGraw-Hill Education.

S. Brito, M. A., Rossi, J. L., S. De Souza, S. R., & V. Braga, R. T. (2012). An Experience on Applying Software Testing for Teaching Introductory Programming Courses. *CLEI Electronic Journal*, *15*(1). https://doi.org/10.19153/cleiej.15.1.4

Universidad Nacional Autónoma de México, & Campos Chiu, C. (2015, abril). *Las pruebas en el desarrollo de software* (TFG). http://132.248.9.195/ptd2015/abril/0728331/Index.html

Universidad Nacional Autónoma de México, & De las Nieves Sánchez Guerrero, G. (2007, septiembre). *Guía para la elaboración de un plan de pruebas para el desarrollo de sistemas de software: El caso del SIIF en la CNSF* (TFM). http://132.248.9.195/pd2007/0621283/Index.html

Universidad Nacional Autónoma de México, & González Fernández, C. U. (2015). *Propuesta para la implementación del proceso de pruebas de software en una institución educativa* (TFG). http://132.248.9.195/ptd2015/octubre/0736796/Index.html

Universidad Nacional Autónoma de México, & Herrera Carrera, D. (2001, marzo). *Visión preventiva de las pruebas en el software* (TFM). http://132.248.9.195/pd2001/290811/Index.html

Virender, S. (2021, 7 julio). *Crystal Method in Agile*. Toolsqa. Recuperado 12 de marzo de 2021, de https://www.toolsqa.com/agile/crystal-method/