Facultad 2

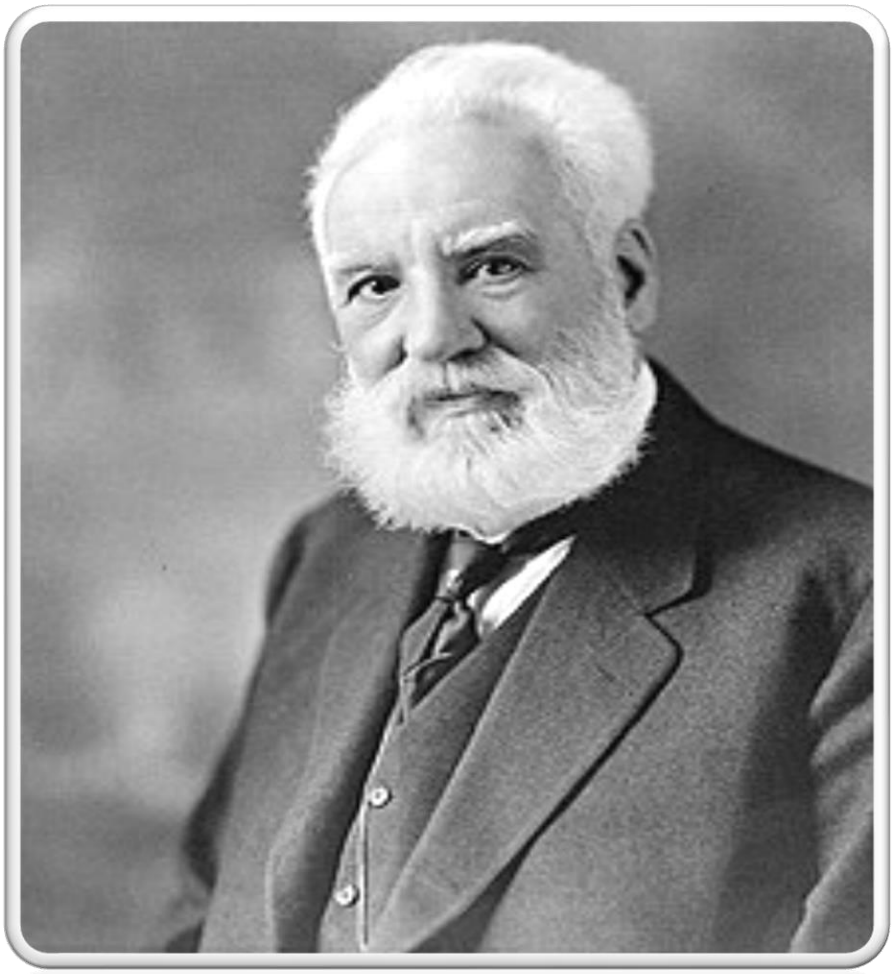
Aplicación móvil para la gestión de capacitación en el Sistema Integral de Gestión Empresarial “DISTRA”

Trabajo de diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autor: Leandro Bolívar Smith
Tutores: Ing. Joiser Bruzón Estrada
Ing. Yosniel Montes de Oca Gonzáles
Esp. Juan Carlos Herrera Betancourt

La Habana, diciembre del 2023

Año 65 de la Revolución



Antes que cualquier otra cosa, la preparación es la llave del éxito.

Alexander Graham Bell

DECLARACIÓN DE AUTORÍA

Declaro por este medio que Leandro Bolívar Smith, es el autor del Trabajo de Diploma: “Gestión estratégica de capacitación empresarial y su incidencia en el Sistema Integral de Gestión Empresarial ´DISTRA´” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio, así como los derechos patrimoniales con carácter exclusivo. Y para que así conste, firmamos la presente declaración de autoría en La Habana a los ____ días del mes de _____ del año 2023.

_____

Firma del Autor

Leandro Bolívar Smith

_____ _____

Firma del Tutor Firma del Cotutor

Ing. Joiser Bruzón Estrada Ing. Yosniel Montes de Oca Gonzáles

_____

Firma del Tutor

Esp. Juan Carlos Herrera Betancourt

RESUMEN

La capacitación constituye una herramienta que responde a las necesidades de una organización, es un elemento importante en la formación y el desarrollo de los recursos humanos. En la Empresa de Tecnología de la Información para la Defensa, la capacitación tiene un papel esencial en el despliegue del Sistema Integral de Gestión Empresarial “DISTRA”. En el centro de implantación de esta empresa para llevar a cabo estos procesos de formación se registran en las fuentes bibliográficas, modelos, procedimientos, normas y manuales a nivel nacional e internacional, las cuales son importantes para apoyar una capacitación virtual y a distancia con calidad. En la actualidad este proceso se lleva a cabo de manera presencial, haciendo que una parte del equipo se traslade hasta las diferentes entidades donde se encuentra desplegado el sistema. En algunos momentos no se cuenta con acceso en línea de los medios tecnológicos presentes en la empresa y en ocasiones no hay existencia de medios tecnológicos, por lo que el equipo debe trasladar sus propios medios de trabajo. De ahí que el objetivo de esta investigación radica en el desarrollo de una aplicación Android para la capacitación virtual del cliente, posibilitando el intercambio de información entre los usuarios. Para llevar a cabo la investigación se utiliza la plataforma Android, desarrollada sobre el marco de trabajo Flutter y el lenguaje de programación Dart. Con el fin de entregar al cliente una solución libre y confiable, se muestran los resultados de un conjunto de pruebas realizadas al producto resultante.

PALABRAS CLAVE: aplicación Android, capacitación, manual de usuario, sistema de gestión

ABSTRACT

…

Keywords: Android application, training, user manual, management system.

ÍNDICE

INTRODUCCIÓN 1

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA 6

1.1 La capacitación del cliente como herramienta para el manejo de sistemas informáticos 6

1.2 Caracterización del Sistema de Gestión empresarial DISTRA 12

1.3 Estudio de sistemas homólogos 14

1.3.1 Estudios de sistemas homólogos internacionales 15

1.3.2 Estudios de sistemas homólogos nacionales 16

1.3.3 Análisis de las soluciones encontradas 16

1.4 Metodología de desarrollo de software 17

1.4.1 *Scrum* 18

1.4.2 Kanban 18

1.4.3 Extreme Programming 18

1.4.4 Crystal 18

1.4.5 AUP-UCI 19

1.5 Lenguaje de modelado y herramientas 21

1.5.1 Lenguaje de modelado 21

1.5.2 Entorno de desarrollo 22

1.5.3 Lenguaje de programación 24

1.6 Conclusiones del capítulo 26

CAPÍTULO 2: ANÁLISIS Y DISEÑO 27

2.1 Características de la propuesta de solución 27

2.2 Requisitos funcionales 27

2.3 Requisitos no funcionales 31

2.4 Historias de usuario 32

Historia de Usuario 33

Historia de Usuario 34

2.5 Arquitectura de software 35

2.6 Patrones de diseño 39

2.6.1 Patrones Generales de Software para la Asignación de Responsabilidades (GRASP) 40

2.6.2 Patrones de Diseño GOF (The Gang of Four) 40

CAPÍTULO 3: VALIDACIÓN DE LA PROPUESTA DE SOLUCIÓN 46

3.1 Validación de requisitos 46
3.2 Validación del diseño 47
3.3.1 Métrica relación entre clases (RC) 47
3.3.2 Métrica tamaño operacional de clases (TOC) 51
3.3 Pruebas de software 53
3.3.1 Pruebas Internas 55
3.3.2 Pruebas Funcionales 59
CONCLUSIONES FINALES 63
RECOMENDACIONES 64

ÍNDICE DE TABLAS

TABLA 1. RELACIÓN DE REQUISITOS FUNCIONALES 28

TABLA 2. HU_ MOSTRAR SUBSISTEMAS 33

TABLA 3. SUBSISTEMAS DE RECURSOS HUMANOS 34

TABLA 4. MÉTRICA RC. CATEGORÍA POR ATRIBUTO Y CRITERIO DE EVALUACIÓN 48

TABLA 5. MÉTRICA TOC. CATEGORÍA POR ATRIBUTOS Y CRITERIO DE EVALUACIÓN 51

TABLA 6. CASO DE PRUEBA PARA LA RUTA BÁSICA 1 58

TABLA 7. CASO DE PRUEBA PARA LA RUTA BÁSICA 2 58

TABLA 8. CASO DE PRUEBA PARA LA RUTA BÁSICA 3 58

TABLA 9. CASO DE PRUEBA PARA LA RUTA BÁSICA 4 ..... **¡ERROR! MARCADOR NO DEFINIDO.**

ÍNDICE DE FIGURAS

Ilustración 1. Interfaz de Inicio de DISTRA 12

Ilustración 2. Herramientas de DISTRA 13

Ilustración 3. Funcionalidades de DISTRA 14

Ilustración 4. Patrón arquitectónico Modelo-Vista-Presentador..... 37

Ilustración 5. Diagrama de Paquete de la Arquitectura Modelo-Vista-Presentador 38

Ilustración 6. Diagrama de Componentes de la Arquitectura Modelo-Vista-Presentador..... 39

Ilustración 7. Clase "AppTheme" evidenciando el patrón “Builder” 41

Ilustración 8. Clase "CustomBottomNavigaton" evidenciando el patrón “State" .. 42

Ilustración 9. Clase "BottomCRM" evidenciando el patrón “Decorador" 43

Ilustración 10. Clase "Subsistema Screen" evidenciando el patrón “Composite" ... 44

Ilustración 11. Clases y cantidad de dependencias..... 49

Ilustración 12. Acoplamiento, complejidad de mantenimiento, cantidad de pruebas y reutilización de clases 50

Ilustración 13. Representación de la cantidad de clases por cantidad de procedimientos..... 52

Ilustración 14. Responsabilidad, complejidad y reutilización de las clases 53

Ilustración 15. Código del método HomeScreen() 56

ILUSTRACIÓN 16. GRAFO DEL FLUJO GETCURRENTINDEX() 57

ILUSTRACIÓN 17. REPRESENTACIÓN DE LA CANTIDAD DE NC POR ITERACIONES..... 60

INTRODUCCIÓN

Una empresa es una organización que se dedica a la producción o comercialización de bienes o servicios con el fin de obtener beneficios económicos. Los procesos que se llevan a cabo en la misma pueden variar dependiendo del tipo de negocio, entre los más comunes incluyen la planificación estratégica, la gestión de recursos humanos, la producción o prestación de servicios, la contabilidad y finanzas, el marketing y ventas, la gestión de la cadena de suministros y el proceso de capacitación (Errázuriz, 2023).

La capacitación, constituye un proceso mediante el cual se adquieren conocimientos y actitudes para interactuar en el entorno laboral (Maria, 2018). El proceso de capacitación tiene como objetivo mejorar las habilidades, conocimientos y competencias de los empleados para que puedan desempeñar mejor sus funciones y contribuir al éxito de la empresa. Debe incluir una definición clara de objetivos, conocimientos específicos de trabajo, puede ser ofrecida en diferentes formatos y en variadas estrategias de formación.

Desde la visión empresarial, la capacitación es asumida como una inversión que la empresa realiza esperando una mejora en sus procesos. Se establece como un elemento importante en la formación y el desarrollo de los recursos humanos. En el área de ventas y atención al cliente es fundamental, contribuyendo al desarrollo y a la innovación de las organizaciones, aumentando su competitividad en el mercado.

En la Empresa de Tecnología de la Información para la Defensa (XETID), la capacitación tiene un papel esencial en el despliegue del Sistema Integral de Gestión Empresarial “DISTRA”. Este proceso permite a sus especialistas adquirir los conocimientos, las herramientas, las habilidades y las actitudes para interactuar en el entorno laboral y cumplir con el trabajo que se les asigna, capacitando a los clientes que trabajan con el sistema.

En el centro de implantación de XETID para llevar a cabo estos procesos de formación se registran en las fuentes bibliográficas, modelos, procedimientos, normas y manuales a nivel nacional e internacional, las cuales son importantes para apoyar una capacitación virtual y a distancia con calidad.

En la actualidad este proceso se lleva a cabo de manera presencial, haciendo que una parte del equipo se traslade hasta las diferentes entidades donde se encuentra

desplegado el sistema; no siempre se cuenta con acceso en línea de los medios tecnológicos presentes en la empresa y en ocasiones no hay existencia de medios tecnológicos, por lo que el equipo debe trasladar sus propios medios de trabajo. Al no contar con acceso a la información de manera offline, cuando hay baja conectividad se dificulta la consulta de la información sobre el sistema en la web.

Se han identificado dificultades generales en el centro de implantación de DISTRA en cuanto a:

- Limitados materiales tecnológicos para el desarrollo de la capacitación y evaluación en cuanto al conocimiento sobre el sistema DISTRA.
- Limitaciones en la práctica desde lo virtual durante el trabajo para el desarrollo de la capacitación y adquisición de los conocimientos sobre DISTRA.
- Dificultades para facilitar el aprendizaje abarcando todas las áreas de DISTRA.

Se demanda la creación e instrumentación de estrategias formativas virtuales que doten de un carácter más operativo, contextual y dinámico al proceso de capacitación de los especialistas que despliegan DISTRA. La ampliación del campo de conocimientos frente al ritmo acelerado de la vida cotidiana impone la imperiosa necesidad de optimizar tiempo, los recursos y además hacerles frente a retos y barreras con soluciones innovadoras que garanticen un proceso de capacitación virtual en la empresa que tenga un impacto significativo en los servicios que esta brinda.

Existe la necesidad de fortalecer y favorecer la preparación de los especialistas que implantan DISTRA de forma tal que logren desarrollar de forma virtual, novedosa y pertinente el proceso formativo de capacitación. Además, se debe proyectar de forma eficaz el proceso de enseñanza-aprendizaje en función de las necesidades de los clientes, con la participación de todos los especialistas del sistema DISTRA.

(Luiso, 2019) Declara que la eficacia del proceso de capacitación es perceptible en la medida en que el personal capacitado pone en práctica el conocimiento adquirido. En ese sentido es aconsejable la utilización de indicadores que faciliten medir la efectividad de forma numérica. Por ejemplo, si se quisiera saber la eficacia de la capacitación dada al área de atención al cliente, el número de quejas y el tiempo de resolución, son dos

indicadores que miden la eficacia de la capacitación. Estos pueden ser comparados con resultados anteriores al momento de la capacitación y sería una manera efectiva para medir los verdaderos logros de la superación.

A partir de la situación descrita anteriormente se determina el siguiente **problema de investigación**: ¿Cómo contribuir a la capacitación virtual de los clientes que deseen utilizar el Sistema Integral de Gestión Empresarial “DISTRA”?

Se delimita como **objeto de estudio:** el proceso de capacitación en la modalidad virtual y a distancia.

Para resolver el problema planteado anteriormente se define como **objetivo general:** desarrollar una aplicación Android que contribuya a una eficiente capacitación a distancia sobre el Sistema Integral de Gestión Empresarial “DISTRA”.

Enmarcándose en el **campo de acción:** el proceso de capacitación virtual y a distancia a través de aplicaciones móviles para el Sistema de Gestión Empresarial “DISTRA”.

Para dar solución al problema identificado y guiar el desarrollo de la investigación se definen los siguientes **objetivos específicos:**

1. Definir el marco teórico de la investigación orientado al desarrollo de aplicaciones móviles destinadas a la capacitación.
2. Determinar la metodología a utilizar en la solución.
3. Desarrollar la planificación y el diseño de la solución para obtener los componentes de software asociados al acceso de la información de “DISTRA”.
4. Implementar la solución y valorar su nivel de aceptación y efectividad mediante las pruebas de software.
5. Aplicar métricas de validación y pruebas de software para evaluar la calidad de los resultados obtenidos.

Como **posibles resultados** se tiene**:**

1. Desarrollo de una aplicación Android para el Sistema Integral de Gestión Empresarial “DISTRA”.
2. Mediante estrategias de pruebas de software, realizar una evaluación a la aplicación Android para la capacitación de los clientes del Sistema Integral de Gestión Empresarial “DISTRA”.

3. Contribución a la gestión estratégica de capacitación virtual y a distancia sustentada en las nuevas tecnologías.

Para darle soporte teórico metodológico a la presente investigación se utilizan los siguientes **métodos científicos**

Métodos teóricos:

El método analítico-sintético: se empleó para analizar en profundidad el estado del arte sobre el tema de investigación y, posteriormente, sintetizar la información obtenida de manera concisa y precisa. Permitió identificar las principales tendencias, enfoques y hallazgos relevantes en el campo de estudio, facilitando así una comprensión más clara y estructurada de la temática investigada.

El método histórico-lógico: posibilitó la recopilación de datos sobre las características de los sistemas de acceso a la información, los diferentes paradigmas en el desarrollo de aplicaciones móviles y los aspectos que conectan a ambos.

El método modelación: permitió realizar las actividades encaminadas a la adaptación o construcción de modelos, a través de los cuales se representan parcial o totalmente las cualidades esenciales del fenómeno objeto de estudio.

Métodos empíricos:

El método entrevista: brindó apoyo a la incorporación de conocimientos mediante las entrevistas no estructuradas efectuadas a los especialistas con conocimiento en el área.

La presente investigación está **estructurada** de la siguiente forma:

Capítulo 1: **Fundamentación teórica:** en este capítulo se exponen los conceptos asociados al dominio de la investigación. Se realiza un análisis sobre las tendencias actuales de la capacitación virtual y a distancia, así como los manuales de usuario. Además, se analizan las herramientas para la formación de capacitación reconocidos nacional e internacionalmente, definiéndose el entorno de desarrollo que se utilizará para dar solución al problema planteado en la investigación.

Capítulo 2: Análisis y diseño de la aplicación Android: se realiza una descripción general de la solución propuesta y su funcionamiento, a partir de la cual se derivan las principales funcionalidades que debe contener. Se detallan los principales aspectos relacionados con su diseño y la arquitectura que organiza la lógica del módulo. Se

especifican los patrones del diseño aplicados y los artefactos derivados de la metodología de desarrollo de software seleccionada.

Capítulo 3: Pruebas de la propuesta de solución: se detalla la propuesta de solución al problema planteado, además de los métodos y medios utilizados para cumplir con el objetivo general, resaltando los principales logros alcanzados.

Posteriormente se presentan las conclusiones, recomendaciones y referencias utilizadas en la investigación, así como los anexos que complementan el contenido esencial.

CAPÍTULO 1: Fundamentación teórica

Diversas empresas reconocen la capacitación como una inversión estratégica en su capital humano. Mientras más sólida sea la formación de sus empleados será más factible que se cumplan los objetivos que han trazado. La visión que tienen las empresas sobre la capacitación empresarial al cliente para el manejo de su sistema informático se basa en empoderar a los usuarios para que puedan utilizar el sistema de manera efectiva, maximizando su productividad y satisfacción.

Las empresas están adoptando un enfoque proactivo al capacitar a sus clientes para que utilicen sus sistemas informáticos, ya que esto fortalece la relación cliente-empresa. Las organizaciones reconocen que la capacitación es fundamental para mejorar la eficiencia operativa y garantizar una experiencia positiva para el cliente (Morgan, 2019).

La capacitación consta de varias etapas que interactúan con acciones específicas (Douglas da Silva, 2022):

- Diagnosticar
- Priorizar
- Definir objetivos
- Elaborar el programa de capacitación
- Ejecutar
- Evaluar
- Certificar y cerrar

Al concluir la certificación y el cierre de las acciones de capacitación, la entidad estará en condiciones de ejecutar una nueva planeación y acciones de capacitación, permitiendo mejorar la experiencia del cliente, reducir los costos de soporte técnico y generar una base de usuarios satisfechos.

Se puede afirmar que la capacitación en las organizaciones es una actividad planeada para el aprendizaje de los clientes. Se enfoca en enseñar los contenidos necesarios para sus trabajos de manera eficiente y efectiva, basada en las necesidades reales de la organización, además del aprendizaje diseñado para la mejora de actitudes y conocimientos de todos.

Por su parte (Vargas, 2021), define que una estrategia de capacitación consiste en trazar el camino que se recorrerá para incrementar la competitividad de una persona o equipo. Establece una visión a largo plazo, define los objetivos, crea estrategias para la formación y desarrollo de los conocimientos de trabajadores o clientes de la empresa. La estrategia planifica una ruta del trabajo formativo, que detallará los pasos que hay que dar para pasar de la situación presente al escenario deseado.

También conocida como capacitación estratégica, este plan debe estar en línea con los objetivos y estrategias de la empresa, debe ser coherente con el plan de desarrollo del personal. Se debe tener presente que el plan estratégico de capacitación debe ser un documento vivo y en constante actualización, que se ajuste a las necesidades cambiantes de la empresa.

Para (Garcia, 2022) algunas ventajas que ofrece la capacitación son:

- Reducción en tiempos y supervisión,
- Solución de problemas con diferente visión,
- Sensibilización ante nuevos retos,
- Tiene la dualidad de ser un derecho, pero también una obligación: ofrece desarrollo y demanda compromiso,
- Mayor rendimiento y disminución de tiempos de atención.

La capacitación virtual y a distancia es un enfoque de formación que utiliza tecnologías de comunicación en línea para impartir conocimientos y habilidades a través de medios digitales (Martínez, 2020). Son dos modalidades de educación que no requieren la

presencia física de los participantes, sino que utilizan herramientas tecnológicas para facilitar el aprendizaje. Como características independientes:

- La capacitación a distancia es una alternativa que permite a los participantes tener control sobre el tiempo, el espacio y el ritmo del aprendizaje, sin necesidad de una conexión a internet (Sumit Dutta, 2021). Los materiales pueden ser físicos o multimedia, y se entregan por diversos medios, como correspondencia, correo electrónico, memorias USB o CD (Henandez, 2020).
- La capacitación virtual es una modalidad que requiere una conexión a internet, ya que el contacto con el instructor se hace mediante una plataforma multimedia, donde los participantes pueden acceder a diversos materiales, como documentos, presentaciones o videos (Henandez, 2020).

La unión de estas modalidades de aprendizaje proporciona un enfoque de formación que utiliza tecnologías en línea para impartir conocimientos y habilidades. Ofrece flexibilidad, supera barreras geográficas y utiliza herramientas interactivas para facilitar el aprendizaje. Aunque presenta desafíos, es una opción cada vez más popular para la capacitación en diversos campos y sectores.

La capacitación ha experimentado diversas tendencias debido a los avances tecnológicos, los cambios en el ámbito laboral, las demandas de los empleados y las organizaciones. Se basa cada vez más en el desarrollo de habilidades específicas y relevantes para el entorno laboral, que cambia rápidamente con la tecnología y el mercado (Ana, 2023). Se personaliza y se adapta a las necesidades, intereses y estilos de aprendizaje de cada profesional, ofreciendo contenido moderno y seleccionado. Es realizada de forma continua y permanente, siguiendo el principio del *lifelong learning*, implicando la actualización y la adaptación constante de los conocimientos (Rodríguez, 2023).

Existen diversas formas de considerar la capacitación virtual y a distancia, autores como (Rodríguez, 2013) define algunas tendencias de este proceso en la actualidad:

- Aprendizaje permanente: por su nombre en inglés *lifelong learning*, se trata de un estilo de aprendizaje que considera al desarrollo profesional e individual como algo voluntario, proactivo y constante. Pretende acompañar a los profesionales a través del avance de la tecnología y de la ciencia en general, el cual se da con mucha rapidez. Por ello, sostiene que las formaciones de los clientes deben ser constantes y continuas y se centra en formar un profesional eficiente que trabaje al ritmo de los cambios que ocurren en el mundo.

- Entrenamiento personalizado: se enfoca en la necesidad de formación individual para maximizar los resultados de la misma. Da cuenta que personalizar el aprendizaje y el recorrido del entrenamiento permite que cada individuo crezca exponencialmente. Un entrenamiento personalizado brinda la oportunidad de estudiar a los profesionales y sus estilos de aprendizaje.

- Habilidades para el trabajo a distancia: con el auge del trabajo remoto, las tendencias de capacitación muestran que es imprescindible potenciar las habilidades esenciales para que, al trabajar a distancia, los resultados no se vean perjudicados. Hay habilidades específicas que son valiosas para los colaboradores que se dedican a trabajar de forma remota: autogestión, comunicación efectiva, tecnología y competencias digitales, adaptabilidad y resiliencia.

- Diversidad e inclusión: la diversidad es un componente necesario para potenciar el rendimiento y la productividad. Un estudio de *McKinsey & Company*, entregó que las empresas con mayor porcentaje de diversidad de género tienen un 21% más de posibilidades de superar el promedio de movimientos del mercado que empresas que no apuestan por la diversidad interna. Por otro lado, es importante resaltar que no se habla solamente de género, sino también de la etnia, de los conocimientos y del rango etario.

- Micro aprendizaje: El micro aprendizaje o *micro learning,* es un método de aprendizaje que resulta muy amigable para el talento humano. Consiste en repartir las unidades de estudio en pequeñas sesiones para que la información sea más

fácil de procesar. Cada una de estas cápsulas de aprendizaje responde a un objetivo concreto que se espera profundizar una vez que se haya culminado el programa educativo. En ese sentido, el objetivo de esta tendencia de capacitación es evitar la sobrecarga de conocimientos en los profesionales.

- Acceso a contenido en línea: Las aplicaciones móviles facilitan el acceso a contenido educativo en línea, como cursos, conferencias y materiales de estudio. Los usuarios pueden descargar materiales para acceder a ellos sin conexión, lo que les permite aprender en cualquier momento y lugar, incluso en áreas con conectividad limitada.

La tendencia del acceso a contenido en línea para la capacitación virtual y a distancia ha experimentado un crecimiento significativo en los últimos años y se espera que continúe en aumento. Algunos de los factores que la impulsan son:

- La necesidad de desarrollar nuevas capacidades para afrontar los desafíos y oportunidades del entorno actual y futuro, como la fluidez digital, el aprovechamiento de datos y los métodos de trabajo ágiles.
- La disponibilidad y el avance de las tecnologías y plataformas digitales que facilitan la creación, distribución y consumo de contenido en línea, como los sistemas de gestión de aprendizaje (LMS).
- La preferencia y la demanda de los trabajadores por formas de aprendizaje más flexibles, autónomas, personalizadas y centradas en el usuario, que les permitan aprender a su propio ritmo, en el momento y lugar que elijan, y de acuerdo con sus intereses y necesidades (Melo, 2023).

El acceso a contenido educativo en línea ha revolucionado la forma en que las personas aprenden y se capacitan, brindando una amplia gama de oportunidades para adquirir conocimientos y habilidades en diferentes áreas. El mismo ofrece varias ventajas para la capacitación a distancia:

- Presupuesto: actualmente la capacitación virtual permite que la empresa optimice sus recursos en relación al presupuesto de capacitación empresarial.
- Flexibilidad: los clientes que tomen un curso tienen la posibilidad de elegir el lugar y el horario para hacerlo. Este además puede ser fuera de las horas laborales o en momentos en que no afecte la productividad.

.

- Adaptabilidad: permite adaptarse a las necesidades del cliente, porque deja que cada persona aprenda a su propio ritmo, tome la parte del curso de su interés o pueda enfocarse con mayor detalle en las secciones que fortalecerán sus habilidades.
- Aprendizaje continuo: tener un catálogo de cursos seleccionado por el gerente de recursos humanos o por los líderes de la organización, podrá asegurarte que tus empleados reciban el mismo entrenamiento de una manera consistente y precisa, con la ventaja de que podrán recibir todos los materiales vía electrónica, archivarlos en sus dispositivos personales y tenerlos a su disposición en cualquier momento.

Si bien el acceso a contenido en línea ofrece muchas ventajas para la capacitación a distancia, también existen ventajas significativas en tener acceso a la información de forma offline:

- Permite ahorrar datos y recursos de conexión, lo que puede ser útil en zonas con poca cobertura o con tarifas elevadas.
- Al tener acceso a la información de forma offline, no se necesita utilizar datos móviles o una conexión Wi-Fi para acceder al contenido. Esto puede resultar en un ahorro significativo de costos si se tiene una limitación de datos o si no se tiene acceso a una conexión Wi-Fi gratuita.
- Facilita el control y la verificación de la calidad y la fiabilidad de la información, al provenir de fuentes seleccionadas previamente.
- Al acceder a la información de forma offline, se tiene mayor control sobre la seguridad de los datos. No se necesita compartir información personal o realizar transacciones en línea, lo que reduce los riesgos de seguridad asociados con el acceso en línea (Melo, 2023).

El acceso a la información offline garantiza que los clientes puedan acceder a recursos educativos incluso en áreas donde la conectividad en línea es limitada o inexistente. La disponibilidad de información fuera de línea amplía las oportunidades de capacitación al permitir a los clientes aprendan en cualquier momento y lugar, sin depender de una conexión a Internet.

Las aplicaciones Android son herramientas importantes en el proceso de capacitación debido a su capacidad para proporcionar un acceso conveniente y personalizado al contenido educativo. Estas aplicaciones permiten a los usuarios aprender a su propio ritmo, acceder a materiales de capacitación interactivos y recibir retroalimentación inmediata. Además, facilitan la integración de diferentes formatos de contenido, como videos, simulaciones y documentos, lo que mejora la experiencia de aprendizaje.

DISTRA es un Sistema de Planificación de Recursos Empresariales (ERP), que le permite integrar y optimizar procesos de negocio, gestionar eficazmente la información y reducir los costos operativos.

Ilustración 1. *Interfaz de Inicio de DISTRA (Elaboración propia)*

Con su uso se puede gestionar, de forma centralizada, la información relevante de las estructuras organizativas e integrar todos los procesos de trabajo. Está compuesto por diversos subsistemas y módulos, que cubren las principales áreas de gestión de cualquier entidad:

1. Módulo de Seguridad
2. Módulo de Datos Maestros
3. Módulo de Gestión Organizacional
4. Módulo Capital Humano
5. Módulo de Gestión de Inventario

.

6. Módulo de Punto de venta
7. Módulo de Facturación
8. Módulo de Producción
9. Módulo de Gestión Contable y Financiera
10. Módulo para la Gestión de los Activos Fijos Tangibles (AFT)
11. Módulo de Planificación
12. Módulo de Gestión Comercial

Se trata de una oferta muy útil para medianas y grandes empresas, que tengan varias entidades subordinadas y poder aplicar el teletrabajo, en cualquiera de las áreas de gestión de la entidad. DISTRA trabaja en conjunto con otros productos de XETID, lo que garantiza una mayor funcionalidad.

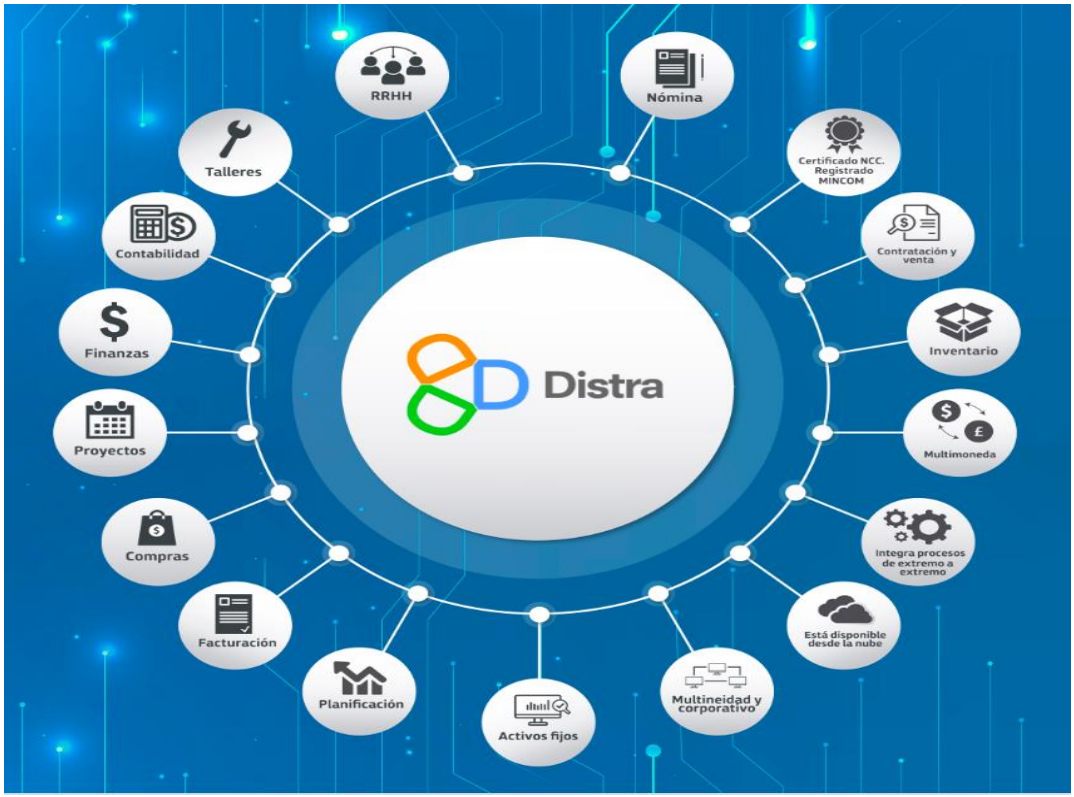


***Ilustración 2**. Herramientas de DISTRA (Elaboración propia)*

Está asociado a la Plataforma de Comercio Electrónico ENZONA, para que los procesos de pago sean más sencillos y seguros. También cuenta con herramientas de gestión de procesos, las cuales tienen disponibles procesos como solicitud de materiales, solicitud de viáticos, solicitud de pagos menores, solicitud de compras y pagos, solicitud de

.

vacaciones, solicitud de licencias no remuneradas, entre otros, según las necesidades que se identifican. Trabaja en conjunto con plataformas de inteligencia de negocios; que hace posible disponer de un amplio y configurable catálogo de informes que facilitan el trabajo.

DISTRA permite controlar la entidad de una forma más práctica, fácil y ágil, ya que está diseñada para hacerlo y puedes encontrar la información de forma precisa y sin deambular por la aplicación.

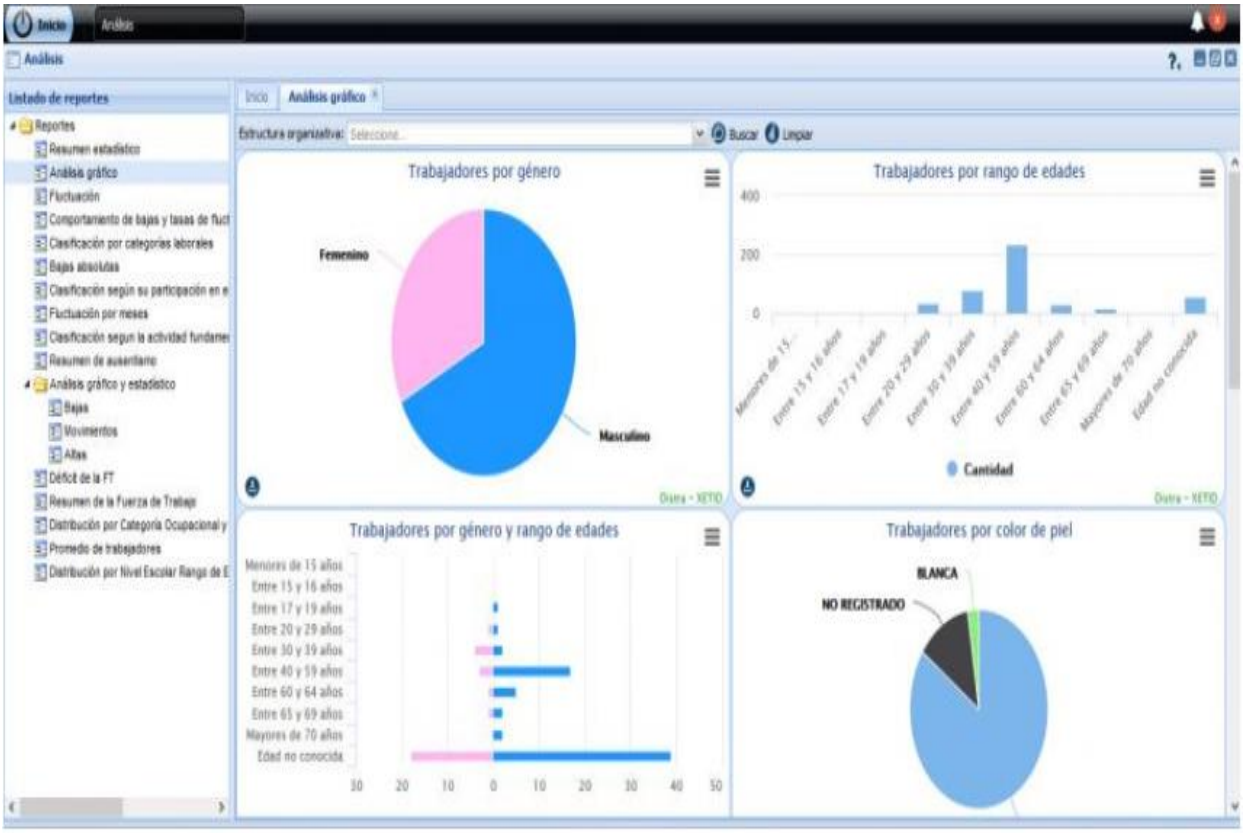


Ilustración 3. *Funcionalidades de DISTRA (Elaboración propia)*

DISTRA proporciona propuestas de servicios adaptadas a todas las necesidades de la entidad; porque antes de implementarlo en la empresa o negocio, se realizará un estudio previo por parte de los especialistas.

La capacitación del personal es esencial para el éxito de cualquier empresa, y las aplicaciones móviles se están convirtiendo en una herramienta popular para ello y el desarrollo de habilidades en el lugar de trabajo. A continuación, se presenta un estudio

sobre las aplicaciones Android *e-learning* que apoyan el proceso de capacitación (Julian, 2022) .

1.3.1 Estudios de sistemas homólogos internacionales

- ***Udemy Mobile***

Es una aplicación en la cual se encuentran muchos cursos online, tanto gratuitos como de pago. Se puede acceder a la aplicación Udemy en versiones de iOS y Android. Sobre todo, les permite a sus empleados tomar cualquiera de los miles de cursos disponibles a través de la plataforma Udemy. Tener en cuenta que hay una tarifa para muchos de los cursos (Gutiérrez, 2023).

- ***Skill Pill***

Esta aplicación gratuita, sirve videos de entrenamiento rápido (“píldoras”) para los empleados que usan iOS, Android, teléfonos Windows y BlackBerry. Ofrece una buena selección de temas para capacitación, incluidos administración, ventas y marketing, y otros temas específicos. Aunque la aplicación es gratuita, hay una tarifa para los cursos. También desarrollan cursos personalizados para satisfacer las necesidades específicas de capacitación (Silla, 2021).

- ***DesingJot***

Creada por Allen Comunication Learning Services Inc., esta es la primera aplicación creada para ayudar a los diseñadores de instrucción, formadores y consultores de rendimiento a desarrollar una mejor capacitación. Con una inversión de $ 4.99, la aplicación ayuda a diseñadores de instrucción nuevos y experimentados a colaborar con sus clientes para crear nuevos cursos (designjet, n.d.).

- ***BoostHQ***

BoostHQ es un concepto diferente de las aplicaciones mencionadas anteriormente, ya que es una aplicación gratuita para compartir contenido que le permite enviar información útil para seleccionar personas a su alrededor. Permite compartir su contenido con un público más amplio o usar BoostHQ en colaboración en círculos más pequeños. Autoriza a elegir el contenido que se desea enviar a los empleados e incluso permite que los mismos envíen contenido a otros empleados, si se les da el permiso. El contenido se puede obtener de cualquier recurso y luego se organiza en canales. Los análisis de

BoostHQ le brindan retroalimentación y permite saber qué es efectivo para los empleados y qué no (López, 2020).

1.3.2 Estudios de sistemas homólogos nacionales

- **Habana Trans**

La aplicación HabanaTrans es un software diseñado para Android, el cual ofrece a los usuarios la información actualizada sobre las rutas de transporte en ciudad de La Habana. Emplea mapas de la ciudad para localizar las paradas de ómnibus y señalizar los recorridos de los medios de transporte, mientras ofrece la ubicación mediante el sistema GPS. En su descripción en el sitio cubano para aplicaciones *Apklis.cu*, los desarrolladores publicaron: “surge con el objetivo de que la población en la capital, o el que venga, tenga pleno conocimiento de cuáles son las vías para transportarse de un lugar a otro, ya sea ómnibus, trenes y demás” (Gonzales, 2019).

- **ARTEX**

La Empresa de Promociones Artísticas y Literarias por sus siglas ARTEX, cuenta con una aplicación móvil que contiene un directorio de todas las tiendas y centros culturales en el país, así como de las Academias de Arte y Cultura, proyecto de Paradiso y su agencia de turismo cultural. La guía informa además sobre las opcionales culturales brindadas por Paradiso, diseñadas para acercar al visitante a lo más autóctono de la cultura cubana, y los Burós de Información en los que se pueden reservar. Incluye también una galería de las líneas de productos que habitualmente se encuentran en las tiendas de la empresa (Issel, 2016).

1.3.3 Análisis de las soluciones encontradas

Luego de realizar un análisis en los sistemas seleccionados, observando sus características y contribuciones, se resume que los elementos encontrados no constituyen una solución debido a que no satisfacen las necesidades de la situación problemática:

- No emplean en sus soluciones el acceso a la información de manera offline.
- No realizan un seguimiento del proceso de capacitación.
- La información que contienen no está disponible en múltiples formatos.

.

A pesar de las limitaciones identificadas, estas aplicaciones, aportaron elementos importantes para el desarrollo de la solución a la problemática anteriormente plateada, como, por ejemplo:

- La aplicación Udemy Mobile ofrece muchos cursos tanto gratuitos como de pago y les permite a sus usuarios acceder a cualquier curso disponible en la plataforma.
- La aplicación Skill Pill utiliza videos como forma de entrenamiento del conocimiento, es decir, mucha de la información que utilizan todos los cursos para capacitar de los que dispone, utilizan el formato audiovisual para cumplir con este objetivo.
- La aplicación Artex ofrece una amplia gama de información en diversos formatos de todo lo relacionado en su ámbito y además de los servicios que presta la organización. Brinda una guía que informa sobre todas las opciones accesibles que este prestando la entidad Paradiso en tiempo real.

Las metodologías de desarrollo de software son técnicas y prácticas que se utilizan para planificar, diseñar, construir, probar y mantener aplicaciones de software. Estas se dividen en dos categorías principales: metodologías ágiles y metodologías pesadas (también conocidas como metodologías tradicionales). Las metodologías pesadas son lineales y secuenciales, y se dividen en fases específicas que se completan antes de pasar a la siguiente fase. Estas metodologías se centran en la planificación detallada y en la documentación exhaustiva y, las metodologías ágiles se caracterizan por su enfoque iterativo e incremental, lo que permite a los equipos de desarrollo adaptarse rápidamente a los cambios en los requisitos del usuario. Las metodologías ágiles se enfocan en la entrega continua de software funcional y en la colaboración entre los miembros del equipo de desarrollo (AdmItsqmet, 2022).

Las metodologías ágiles son ideales para proyectos en los que los requisitos del usuario pueden cambiar rápidamente y se necesita una entrega rápida de software funcional. Las metodologías ágiles se enfocan en la colaboración entre los miembros del equipo de desarrollo y con el cliente. Estas metodologías permiten a los equipos de desarrollo adaptarse rápidamente a los cambios en los requisitos del usuario y entregar software

funcional de alta calidad de manera más eficiente. La elección de una metodología de software depende del proyecto y de las necesidades del equipo de desarrollo, por lo cual se ha elegido el uso de una metodología ágil para el desarrollo de esta aplicación Android. A continuación, se ofrece una breve descripción de algunas de estas metodologías:

1.4.1 *Scrum*

Scrum es un marco de trabajo ágil que se utiliza para gestionar proyectos de software. En *Scrum*, el equipo de desarrollo trabaja en ciclos cortos llamados Sprint, generalmente de dos a cuatro semanas. Al comienzo de cada sprint, el equipo de desarrollo se reúne para planificar el trabajo que se realizará en ese sprint y al final del sprint, se realiza una revisión para demostrar el trabajo completado (Julian, 2022).

1.4.2 Kanban

Kanban es un sistema de gestión de proyectos ágil que se centra en la visualización del flujo de trabajo. En Kanban, el trabajo se visualiza en un tablero Kanban, que muestra las tareas pendientes, en progreso y completadas. Cada tarea se mueve a través del tablero Kanban a medida que se realiza, lo que permite al equipo de desarrollo visualizar el flujo de trabajo y gestionar eficazmente el tiempo y los recursos (Martins, what-is-kanban, 2022).

1.4.3 Extreme Programming

Extreme Programming es una metodología ágil que se enfoca en la calidad del software y en la satisfacción del cliente. En XP, el equipo de desarrollo trabaja en ciclos cortos de desarrollo (generalmente de una a dos semanas) y se enfoca en prácticas de desarrollo de software como pruebas automatizadas, integración continua y refactorización del código (Raeburn, 2022).

1.4.4 Crystal

Crystal es una familia de metodologías ágiles que se adaptan a diferentes tamaños de equipos y proyectos. Las metodologías Crystal se enfocan en la comunicación efectiva entre los miembros del equipo de desarrollo y en la entrega de software funcional de alta calidad (RINCÓNPM, 2022).

1.4.5 AUP-UCI

La metodología AUP-UCI (Agile Unified Process-UC Irvine) es una metodología de desarrollo de software que combina los principios ágiles con una estructura y disciplina más formal. Esta metodología se enfoca en la entrega continua de software funcional y en la colaboración entre los miembros del equipo de desarrollo y con el cliente, al mismo tiempo que proporciona una estructura para la gestión de proyectos de software.

La metodología AUP-UCI se divide en cuatro fases principales:

Inicio: En esta fase, se definen los objetivos del proyecto, se identifican los requisitos del usuario y se elabora un plan de proyecto preliminar.

Elaboración: En esta fase, se elaboran los requisitos del usuario y se realiza un análisis detallado del alcance del proyecto. También se elabora un plan de proyecto detallado y se identifican los riesgos del proyecto.

Construcción: En esta fase, se desarrolla el software de acuerdo con los requisitos y el plan de proyecto. El equipo de desarrollo trabaja en ciclos cortos de desarrollo (Sprint) y se enfoca en la entrega continua de software funcional.

Transición: En esta fase, se realiza la prueba final del software y se prepara para su lanzamiento. También se proporciona capacitación y soporte al usuario y se realiza la entrega final del software.

La metodología AUP-UCI también se enfoca en la documentación adecuada del proyecto y en la gestión eficaz del cambio. La metodología se adapta a las necesidades específicas del proyecto y del equipo de desarrollo, lo que permite una mayor flexibilidad en la planificación y el desarrollo del proyecto.

En resumen, la metodología AUP-UCI se enfoca en la entrega continua de software funcional y en la colaboración entre los miembros del equipo de desarrollo y con el cliente, al mismo tiempo que proporciona una estructura para la gestión de proyectos de software. La metodología se divide en cuatro fases principales y se adapta a las necesidades específicas del proyecto y del equipo de desarrollo.

La metodología AUP-UCI consta de 4 escenarios para modelar el sistema aplicando diferentes formas de encapsular los requisitos las cuales son Caso de Uso del Sistema, Historias de Usuario y Descripción de Requisitos por Proceso (CUS, HU y DRP por sus siglas) y variantes de modelado de negocio tales como Caso de Uso del Negocio,

Descripción de Proceso de Negocio o Modelo Conceptual (CUN, DPN y MC). Escogiendo el escenario #4 para la propuesta de solución (Sánchez, 2014):

•**Escenario No 4**: proyectos que no modelen negocio solo pueden modelar el sistema con HU.

Se elige este escenario ya que se aplica a proyectos que tienen un negocio bien definido. El cliente estará acompañando al equipo de desarrollo para convenir los detalles de los requisitos encontrados y poder implementarlos, probarlos y validarlos. Se recomienda en proyectos no extensos, ya que las Historias de usuarios (HU) no deben poseer demasiada información. Además, esta metodología en su escenario #4 cuenta con:

- Adaptabilidad a los cambios: Se basa en principios ágiles y permite adaptarse a los cambios de manera eficiente. En un escenario en el que los requisitos y las circunstancias pueden cambiar con frecuencia, esta metodología facilita la incorporación de cambios y ajustes en el proceso de desarrollo.
- Entrega incremental: Promueve la entrega incremental de software funcional. Esto significa que se pueden obtener resultados tangibles en etapas tempranas del proyecto, lo que permite una retroalimentación más temprana por parte de los interesados y una mayor visibilidad del progreso.
- Gestión de riesgos: Incorpora la gestión de riesgos como parte integral del proceso de desarrollo. En el escenario 4, donde puede haber riesgos técnicos o de infraestructura, esta metodología ayuda a identificar, mitigar y controlar los riesgos de manera proactiva
- Mejora de la calidad: Fomenta la adopción de buenas prácticas de desarrollo de software, como pruebas automatizadas, revisión de código y aseguramiento de la calidad. Esto contribuye a mejorar la calidad del software entregado en el escenario 4.
- Escalabilidad: Se puede adaptar a proyectos de diferentes tamaños y complejidades. En el escenario 4, donde puede haber un crecimiento o expansión de los requisitos y alcance del proyecto, esta metodología permite una gestión más efectiva de los recursos y una adaptación flexible a medida que el proyecto evoluciona.

- Estandarización de la infraestructura: En el escenario 4, donde se requiere una infraestructura tecnológica sólida, la metodología AUP-UCI proporciona una guía para estandarizar y optimizar la infraestructura de desarrollo. Esto garantiza que todos los miembros del equipo trabajen con herramientas y entornos consistentes, lo que facilita la colaboración y reduce posibles problemas de compatibilidad.

En conclusión, se utilizó la metodología AUP-UCI en su escenario 4 por: su estandarización y optimización de la infraestructura tecnológica, lo que facilita la colaboración y reduce problemas de compatibilidad; la gestión eficiente de los recursos de desarrollo, maximizando la productividad y minimizando desperdicios; el control y seguimiento del progreso del proyecto, permitiendo una visibilidad clara y la toma de acciones correctivas oportunas y su gestión efectiva de riesgos, identificando y abordando los riesgos de manera proactiva para reducir problemas y mitigar impactos negativos.

Las herramientas de software utilizadas cumplen con el objetivo de facilitar, optimizar y mejorar el desempeño de nuestro trabajo. Constituyen un conjunto de programas y ayudas a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software, permitiendo la creación de todo tipo de diagramas. En la metodología AUP, se hace necesario el uso de una herramienta de Ingeniería de Software asistida por computadora en las disciplinas de Análisis, Diseño e Implementación, donde se genera la mayor cantidad de artefactos.

1.5.1 Lenguaje de modelado

El Lenguaje Unificado de Modelado (UML), permite visualizar, especificar, construir y documentar los artefactos de un sistema que involucra el desarrollo de un software. Un modelo UML describe lo que supuestamente hará un sistema, no cómo implementarlo. Este lenguaje dispone de reglas para combinar tales elementos y permite la modelación de sistemas con tecnología orientada a objetos. Los diagramas son entes importantes de UML, cuya finalidad es representar diversas perspectivas de un sistema, a las cuales se les conoce como modelo. El modelo gráfico de UML tiene un vocabulario en el que se identifican: elementos, relaciones y diagramas (Grady Booch, 2005).

Herramienta CASE Visual Paradigm 16.3

Visual Paradigm es una herramienta UML CASE compatible con UML 2, SysML y la notación de modelado de procesos empresariales del grupo de gestión de objetos. Además del soporte de modelado, proporciona capacidades de ingeniería de código y generación de informes, incluida la generación de código (Tec, 2023). Es una herramienta que se puede emplear en el uso de metodologías ágiles, permite la generación de gráficas y tablas, así como la creación de modelos UML con capacidad de diseñar UX. Además, es fácil de usar y se puede compartir en equipo

UML: UML son las siglas de "*Unified Modeling Language*" o "Lenguaje Unificado de Modelado". Se trata de un estándar que se ha adoptado a nivel internacional por numerosos organismos y empresas para crear esquemas, diagramas y documentación relativa a los desarrollos de software (programas informáticos). Es una herramienta propia de personas que tienen conocimientos relativamente avanzados de programación y es frecuentemente usada por analistas funcionales (aquellos que definen qué debe hacer un programa sin entrar a escribir el código) y analistas-programadores (aquellos que, dado un problema, lo estudian y escriben el código informático para resolverlo en un lenguaje como Java, C#, Python o cualquier otro) (Krall, 2023). Se decide la utilización de este lenguaje de modelado, pues posibilita estandarizar la documentación referente al proceso de desarrollo de software; es uno de los lenguajes de modelados más utilizados y difundidos.

1.5.2 Entorno de desarrollo

Un entorno de desarrollo es un espacio de trabajo que permite a los desarrolladores crear una aplicación o realizar cambios en ella sin afectar a la versión real del producto de software. Estos cambios pueden incluir el mantenimiento, la depuración y la aplicación de parches (tokioschool, 2023). En general, el término entorno de desarrollo incluye todo el entorno, desde el desarrollo y la puesta en escena hasta los servidores de producción (V., 2023).

Un entorno de desarrollo integrado (IDE), es un paquete de software que combina todas las herramientas de desarrollo en una única interfaz gráfica de usuario (GUI). Como resultado, esto hace que el proceso de desarrollo sea más eficiente y rápido.

Algunos ejemplos de entornos de desarrollo integrados populares son:

.

-NetBeans
-Microsoft Visual Studio
-Adobe Flex Builder
-Eclipse

Los IDE poseen características que pueden mejorar la productividad general:

-Editor de código
-Finalización de código
-Compilador
-Depurador
-Herramientas de automatización de creación
-Compatibilidad con el lenguaje de programación
-Control de versiones

Además, algunos IDEs pueden incluir también un navegador de clases, un navegador de objetos y un diagrama de clases (V., 2023)

Visual Studio Code 1.79.2

Visual Studio Code (VS Code) es un editor de código fuente desarrollado por Microsoft. Es una herramienta muy popular y ampliamente utilizada por desarrolladores de software. VS Code es un editor de texto con resaltado de sintaxis, autocompletado de código, depuración integrada, control de versiones y una amplia gama de extensiones que permiten personalizar y ampliar su funcionalidad (Visual Studio Code, 2023).

VS Code es una herramienta muy versátil y popular para el desarrollo de aplicaciones Android. Algunas de sus ventajas que ofrece para este propósito son:

-Extensibilidad: VS Code cuenta con una amplia gama de extensiones disponibles que permiten a los desarrolladores personalizar y ampliar la funcionalidad del editor según sus necesidades específicas.

-Autocompletado de código: VS Code ofrece una función de autocompletado inteligente que ayuda a los desarrolladores a escribir código más rápido y reducir errores. Esto es especialmente útil al trabajar con el SDK de Android y las bibliotecas asociadas.

-Depuración integrada: VS Code ofrece una interfaz de depuración intuitiva y fácil de usar, lo que facilita la identificación y corrección de errores en las aplicaciones Android.

-Control de versiones: VS Code tiene una integración sólida con sistemas de control de versiones como Git, lo que permite a los desarrolladores gestionar fácilmente el control de versiones de su código fuente (Visual Studio Code - Code Editing, 2023).

1.5.3 Lenguaje de programación

Un lenguaje de programación es un conjunto de reglas y estructuras utilizadas para escribir instrucciones que una computadora puedes entender y ejecutar. Proporciona una forma de comunicación entre los programadores y las máquinas, permitiendo la creación de programas y aplicaciones informáticas.

Cada lenguaje tiene su sintaxis y semántica, lo que determina cómo se deben organizar las instrucciones. Estos lenguajes se utilizan para desarrollar una amplia variedad de software, desde aplicaciones de escritorio y móviles, hasta sitios web y sistemas embebidos (Culwin & Lancaster, 2016).

Flutter 3

Flutter es un framework que permite el desarrollo de un proyecto de programación. Es gratuito y de código abierto, y fue creado por Google en mayo de 2017. Permite diseñar rápidamente funcionalidades con el foco en la experiencia de usuario nativa. “La arquitectura en capas permite una completa personalización, que resulta en un renderizado increíblemente rápido y diseños expresivos y flexibles” explican en el sitio web oficial del framework.

Flutter consta de dos funciones, o partes, importantes:

SDK: se trata de una colección de herramientas que permite desarrollar aplicaciones. Esto incluye elementos para compilar código para iOS y Android.

Framework: una colección de elementos de interfaz de usuario reutilizables (por ejemplo: botones, entradas de texto, controles deslizantes) que pueden personalizarse según lo que requiera el proyecto. El SDK de Flutter se basa en el lenguaje de programación Dart, que también fue creado por Google con el objetivo de convertirse en un sucesor de JavaScript. Al igual que este, Dart se ejecuta directamente en el navegador (Cristancho, 2022).

Dart 3.1.2

Dart es un lenguaje de programación con carácter *open source* (de código abierto o con acceso al público mediante este tipo de *software*), establecido con Google para lograr en

conjunto con los desarrolladores de programación un lenguaje que permita estar centralizado en los objetos, así como también logrando un análisis estático en estos mismos. Es destacable mencionar que el lenguaje ha sido modificado tanto en su estructura como en su objetivo desde su primera versión en 2011.

Básicamente, el lenguaje de programación Dart consiste y se sostiene a la vez en la medida de su facilidad y sencillez al momento de establecerse de manera más rápida y cómoda. Ofrece un conjunto de herramientas que logran en el usuario la utilización de estas, que son de tipo de compiladores, analizadores y de formatear. Asimismo, una gran ventaja que ofrece Dart es la ejecución del código en el momento en el que se va desarrollando. También es importante mencionar las características o conocimientos similares que comparte con JavaScript o C ++, además de que ofrece su utilización a partir de aplicaciones web, en servidores, aplicaciones de consola y aplicaciones móviles gracias a Flutter (KeepCoding, que es el lenguaje de programacion dart, 2022).

Android SDK 1.6.4

Android SDK, hace referencia a un kit de desarrollo de software, que en inglés se conoce como *Software Development Kit*, el cual fue diseñado por Google con el propósito de proporcionarle a los desarrolladores elementos necesarios para las apps destinadas a dispositivos Android. El mismo es un conjunto de herramientas enfocadas en el desarrollo de software que está destinado al desarrollo de aplicaciones para distintos sistemas operativos, en este caso, para móviles con sistema operativo Android (KeepCoding, que es android sdk, 2022).

Gradle 7.5

El fichero Gradle en Android Studio es el resultado del uso de la herramienta que utiliza el sistema de Android para compilar nuestra app. También por medio del fichero de Gradle en Android Studio, tenemos la posibilidad de definir las diferentes dependencias que estamos solicitando para llegar a usarlas en el momento de compilar nuestra aplicación. El fichero Gradle en Android Studio utiliza el lenguaje Groovy, aspecto que nos permite realizar diversas tareas sobre la Máquina Virtual de JAVA del inglés JVM (Java Virtual Machine) (KeepCoding, fichero gradle en android studio, 2023).

El estudio del proceso de capacitación a clientes, con el apoyo de las tecnologías móviles, permitió vislumbrar los principales conceptos asociados al desarrollo de una aplicación Android como apoyo a la capacitación virtual para los clientes. Se presta atención a las etapas interactuantes, las tendencias, ventajas y el análisis de los principales elementos sobre la capacitación para que puedan utilizar la aplicación con mayor facilidad. La selección de las tecnologías Visual Paradigm y Visual Studio Code posibilitó obtener una base tecnológica enfocada en las políticas de desarrollo definidas en la XETID.

CAPÍTULO 2: Análisis y diseño

Con el empleo del modelo de investigación, método empírico; la entrevista permitió identificar la necesidad de acceder a toda la información de DISTRA, mediante un medio portátil que garantiza al centro de implantación la eficacia en el proceso de capacitación virtual y a distancia de sus clientes, así como la calidad de este proceso, el acceso a la información de forma portable y offline.

Como solución a la problemática planteada, se pretende desarrollar una aplicación móvil con el objetivo de, contribuir al proceso de capacitación virtual y a distancia de los clientes que emplean DISTRA. Que contribuya a la gestión estratégica de preparación no presencial sustentada en las nuevas tecnologías, así como la celeridad del proceso, el acceso a la información offline y su calidad. Materializando todo su contenido en diferentes formatos, desde texto y documentas hasta videos e imágenes.

La aplicación se convertirá en una de las herramientas para contribuir el proceso de capacitación virtual y a distancia que empleará el equipo encargado de desplegar DISTRA.

Los requisitos funcionales constituyen el proceso mediante el cual se especifican y validan los servicios que debe proporcionar el sistema, así como las restricciones sobre las que se deberá operar (M.J. Escalona, 2002). Los requisitos funcionales son declaraciones de los servicios que prestará el sistema, en la forma en que reaccionará a determinados insumos. En algunos casos, los requisitos funcionales de los sistemas también establecen explícitamente lo que el sistema no debe hacer (requerimientos funcionales y no funcionales ejemplos y tips , 2018).

Después del encuentro con el cliente, y a través de una encuesta aplicada al mismo, se llegó a la conclusión que la aplicación no cuenta con gestión de usuarios, pues se basa en un diseño netamente publicitario. Se obtuvo un total de 6 requisitos funcionales, a los cuales se les asignó una prioridad teniendo en cuenta la importancia fijada por el cliente a partir de sus necesidades. Los mismos se especifican en la siguiente tabla:

Tabla 1. *Relación de requisitos funcionales (Elaboración propia)*

No. HU	No. RF	Nombre	Descripción	Complejidad
HU 1	RF 1	Mostrar una barra inferior de navegación con 3 botones.	La aplicación contendrá una barra de navegación inferior, la cual permitirá navegar por 3 pantallas principales.	Alta
HU 2	RF 2	Mostrar un botón inicio en la barra inferior de navegación.	Al presionar este botón se navega a la pantalla de inicio de la aplicación, pantalla que constituye la primera vista mostrada al usuario.	Alta
HU 3	RF 3	Mostrar un botón subsistema en la barra inferior de navegación.	Al presionar este botón se navegará a la pantalla subsistema. Dicho panel contendrá la información de todos los subsistemas de DISTRA.	Alta
HU 4	RF 4	Mostrar un botón DISTRA en la barra inferior de navegación.	Al presionar este botón se navegará a una interfaz la cual de manera general y abarcadora mostrara al usuario que es DISTRA.	Alta
HU 5	RF5	Mostrar subsistema seguridad.	La aplicación permitirá una vez seleccionado el subsistema de Seguridad, dentro de la vista subsistema, mostrar toda la información referente a este subsistema.	Alta

HU 6	RF 6	Mostrar subsistema Datos Maestros.	La aplicación permitirá una vez seleccionado el subsistema Datos Maestros, dentro de la vista subsistema, mostrar toda la información referente a este subsistema.	Alta
HU 8	RF 8	Mostrar subsistema Recursos Humanos.	La aplicación permitirá una vez seleccionado el subsistema Recursos Humanos, dentro de la vista subsistema, mostrar toda la información referente a este subsistema, y los subsistemas que este compone: • Capital Humano • Gestión Organizacional • Registro de personas • Gestión de movimientos laborales • Registro y control de asistencia del personal • Retribución salarial • Selección e integración • Evaluación del desempeño	Alta
HU 9	RF 9	Mostrar subsistema Comercial.	La aplicación permitirá una vez seleccionado el subsistema Comercial, dentro de la vista subsistema, mostrar toda la información referente a este	Alta

			subsistema y el subsistema que este compone. • Gestión de relación con los clientes (CRM)	
HU 10	RF 10	Mostrar subsistema Planificación.	La aplicación permitirá una vez seleccionado el subsistema Planificación, dentro de la vista subsistema, mostrar toda la información referente a este subsistema y los subsistemas que este compone: • Planificación de material • Planificación de actividades • Planificación de portadores energéticos	Alta
HU 11	RF 11	Mostrar subsistema de Contabilidad.	La aplicación permitirá una vez seleccionado el subsistema Contabilidad, dentro de la vista subsistema, mostrar toda la información referente a este subsistema y los subsistemas que este compone: • Contabilidad general • Operaciones • Contabilidad de costo • Finanzas	Alta

HU 12	RF 12	Mostrar subsistema de Logística.	La aplicación permitirá una vez seleccionado el subsistema Logística, dentro de la vista subsistema, mostrar toda la información referente a este subsistema y los subsistemas que este compone: • Inventario • Activos • Facturación • Talleres	Alta
HU 13	RF 13	Mostrar subsistema Punto de venta.	La aplicación permitirá una vez seleccionado el subsistema Punto de Venta, dentro de la vista subsistema, mostrar toda la información referente a este subsistema.	Alta
HU 14	RF 14	Mostrar subsistema Producción.	La aplicación permitirá una vez seleccionado el subsistema Producción, dentro de la vista subsistema, mostrar toda la información referente a este subsistema.	Alta

Se trata de requisitos que no se refieren directamente a las funciones específicas suministradas por el sistema (características de usuario), sino a las propiedades del sistema: rendimiento, seguridad, disponibilidad. En palabras más sencillas, no hablan de "*lo que*" hace el sistema, sino de "*cómo*" lo hace. Alternativamente, definen restricciones del sistema tales como la capacidad de los dispositivos de entrada/salida y la representación de los datos utilizados en la interfaz del sistema (requerimientos funcionales y no funcionales ejemplos y tips , 2018).

.

Distribuidos en especificaciones de seguridad, se obtuvo un total de 7 requisitos no funcionales, los cuales se relacionan a continuación:

- **Usabilidad**

RnF 1: La aplicación móvil debe de permitir acceder a sus diferentes interfaces con una profundidad máxima de 2 clics.

RnF 2: El sistema debe poseer interfaces amigables, con botones que tengan nombres sugerentes para que usuarios inexpertos puedan interactuar fácilmente con el software.

- **Eficiencia**

RnF 3: La aplicación debe responder en un tiempo menor de 3 segundos a las solicitudes de los usuarios.

- **Escalabilidad**

RnF 4: El sistema debe ser escalable permitiendo que se le puedan agregar nuevas funcionalidades sin afectar las anteriores implementadas.

- **Hardware**

RnF 5: El dispositivo móvil debe contar como mínimo con 3 megas de almacenamiento que la aplicación sea instalada y funcione correctamente.

- **Software**

RnF6: El dispositivo móvil debe tener sistema operativo Android 11 o superior para poder utilizar la aplicación móvil.

- **Diseño o Implementación**

RnF 7: Como lenguaje de programación para el desarrollo de la aplicación se deberá utilizar Dart.

Las historias de usuario se usan como una herramienta de comunicación que combina las fortalezas: escrito y verbal. Describen, en una o dos frases, una funcionalidad de software desde el punto de vista del usuario, con el lenguaje que éste emplearía. El foco está puesto en qué necesidades o problemas soluciona lo que se va a construir. Son una herramienta que agiliza la administración de requisitos, reduciendo la cantidad de documentos formales y tiempo necesarios. Forman parte de la fórmula de captura de funcionalidades definida en 2001 por Ron Jeffries de las tres Cs:

- Card: cada historia de usuario se reduce hasta hacerla fácil de memorizar y de sintetizar en una tarjeta o post-it.

- Conversation: el equipo de desarrollo y el propietario del producto añaden criterios de aceptación a cada historia poco antes de su implementación.

- Confirmation: el propietario del producto o usuario de negocio confirma que el equipo de desarrollo ha entendido y recogido correctamente sus requisitos revisando los criterios de aceptación (Menzinsky, 2018).

Una vez identificados los requisitos se establecieron las prioridades del negocio: alta cuando las funcionalidades son esenciales para el negocio, medias cuando son importantes, pero no influyen en el desarrollo del negocio y baja cuando su ausencia no perjudica al sistema.

Tabla 2. *Hu_ Mostrar subsistemas (Elaboración propia)*

Historia de Usuario	
Número: HU_2	**Nombre:** Mostrar subsistemas
Programador responsable: Leandro Bolívar Smith	**Iteración asignada: 1**
Prioridad: Alta	
Descripción: Muestra todos los subsistemas con los que cuenta el sistema	
Observaciones: Ninguna	

Prototipo:

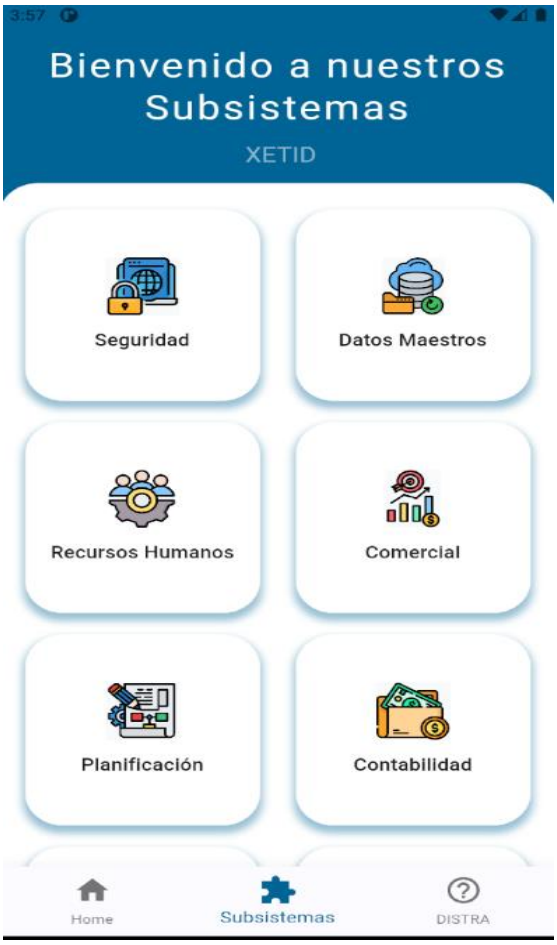


Tabla 3. *Subsistemas de Recursos humanos (Elaboración propia)*

Historia de Usuario	
Número: HU_2	**Nombre:** Mostrar interfaz de autenticación de usuario
Programador responsable: Leandro Bolívar Smith	**Iteración asignada: 3**
Prioridad: Alta	
Descripción: Mostrar los subsistemas con los que cuenta el subsistema de recursos Humanos	
Observaciones: Ninguna	

Prototipo:

La arquitectura del software es una disciplina muy relevante en la actualidad. Constituye un conjunto de patrones y abstracciones coherentes que proporcionan un marco definido y claro para interactuar con el código fuente del software. Los patrones pertenecientes a estas arquitecturas guían a los desarrolladores, analistas y todos los cargos relacionados para lograr cumplir con los requerimientos de la aplicación. La arquitectura de software se selecciona y diseña con base en objetivos (requisitos) y restricciones (Zúñiga, 2020). Existen diversos tipos de arquitecturas de software, cada uno con sus propias características, adaptabilidad a ciertos tipos de problemas y requisitos. Algunos ejemplos de estas(Nazarevich, 2022):

- Arquitectura cliente-servidor

.

- Arquitectura peer-to-peer
- Arquitectura orientada a servicios
- Arquitectura basada en componentes

Los patrones de arquitectura de software son aquellos que permiten diseñar y desarrollar aplicaciones que se basan en la comunicación entre un componente servidor que ofrece servicios o recursos y uno o más componentes clientes que los solicitan a través de una red. Dentro de la arquitectura cliente-servidor, se puede encontrar algunos de estos patrones (Barriento, 2023):

- Modelo-vista-controlador (MVC)
- Arquitectura de microservicios
- Arquitectura de bus de evento

A lo largo del tiempo, han surgido variantes y evoluciones del patrón MVC, que se adaptan a diferentes necesidades y contextos. Algunos de estos:

- Modelo-vista-presentador (MVP)
- Modelo-vista-modelo de vista (MVVM)
- Modelo-Vista-Servicio-Controlador (MVSC)

El patrón MVP es un formato de distribución para la puesta en práctica de la interfaz de usuario (UI) de una aplicación. Como esquema, reduce la complejidad de uso en módulos. En síntesis, MVP persigue la estructuración de lo que los clientes digitales ven

y con lo que interactúan en web o en una aplicación. Es una derivación del Modelo Vista Controlador (MVC).

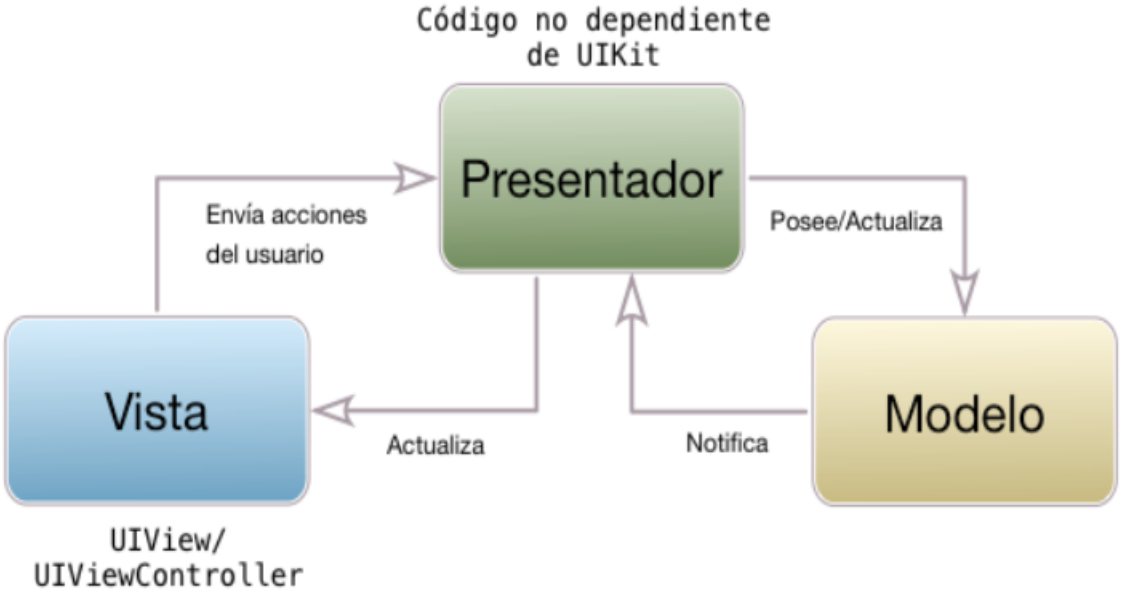


***Ilustración 4**. Patrón arquitectónico Modelo-Vista-Presentador (Elaboración propia)*

Este de diseño consta, como indica su nombre, de 3 aspectos o componentes esenciales:

- Modelo: Es el componente del sistema del que depende el acceso a base de datos. Puede variar, pero lo más común es una memoria caché o API rest. Este elemento abarca, asimismo, la naturaleza de la conexión BD (SQLite).
- Vista: Hace referencia a la visualización del diseño de la interfaz. Es, de cierta forma, lo que los individuos presionan en la pantalla de un teléfono o lo que hace al clicar en el ordenador. Esta capa externa del producto digital se encarga de mostrar propiedades según las peticiones. Se trata de una acción simple para exhibir fragmentos.
- Presentador: Este elemento es el puente entre la vista y el modelo. Gracias al presentador, se da la interacción entre petición y almacenamiento de información (Team, 2023).

Se presenta a continuación una vista lógica de la arquitectura Modelo Vista Presentador (MVP) utilizando el diagrama de paquetes proporcionado por UML.

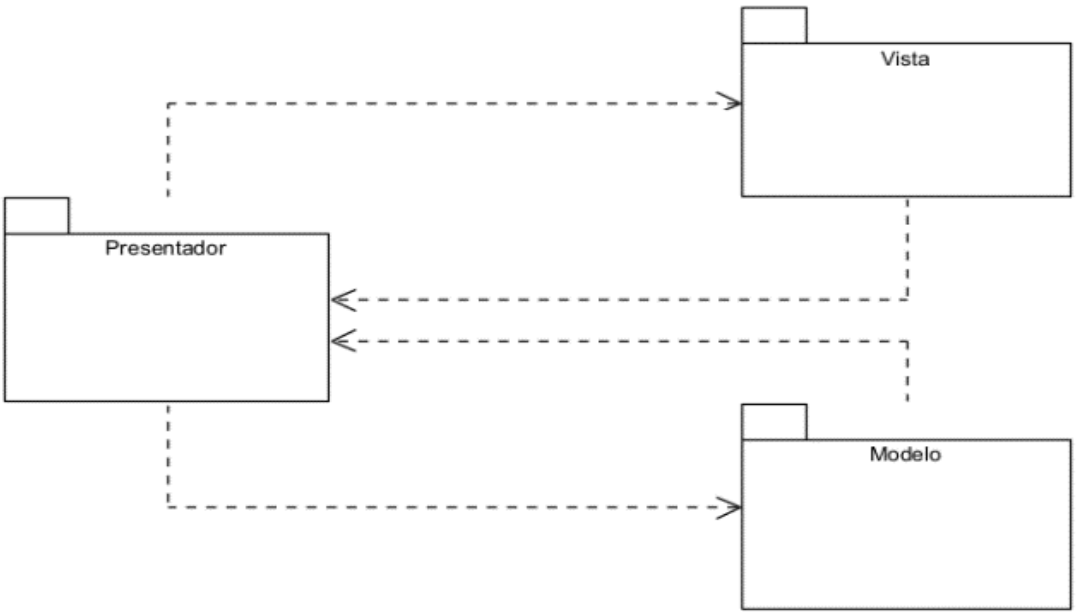


Ilustración 5. *Diagrama de Paquete de la Arquitectura Modelo-Vista-Presentador (Elaboración propia)*

MVP es una forma de organizar el código de una aplicación Android que permite separar la lógica de negocio de la interfaz de usuario. Facilita el mantenimiento y la escalabilidad del código, ya que se evita el acoplamiento entre las clases y se favorece la reutilización de los componentes. Facilita la realización de pruebas unitarias y de integración, ya que se puede aislar el comportamiento de cada capa (modelo, vista y presentador) y simular sus interacciones y mejora el rendimiento y la eficiencia de la aplicación, ya que se reduce la carga de trabajo de la vista y se evita la creación innecesaria de objetos.

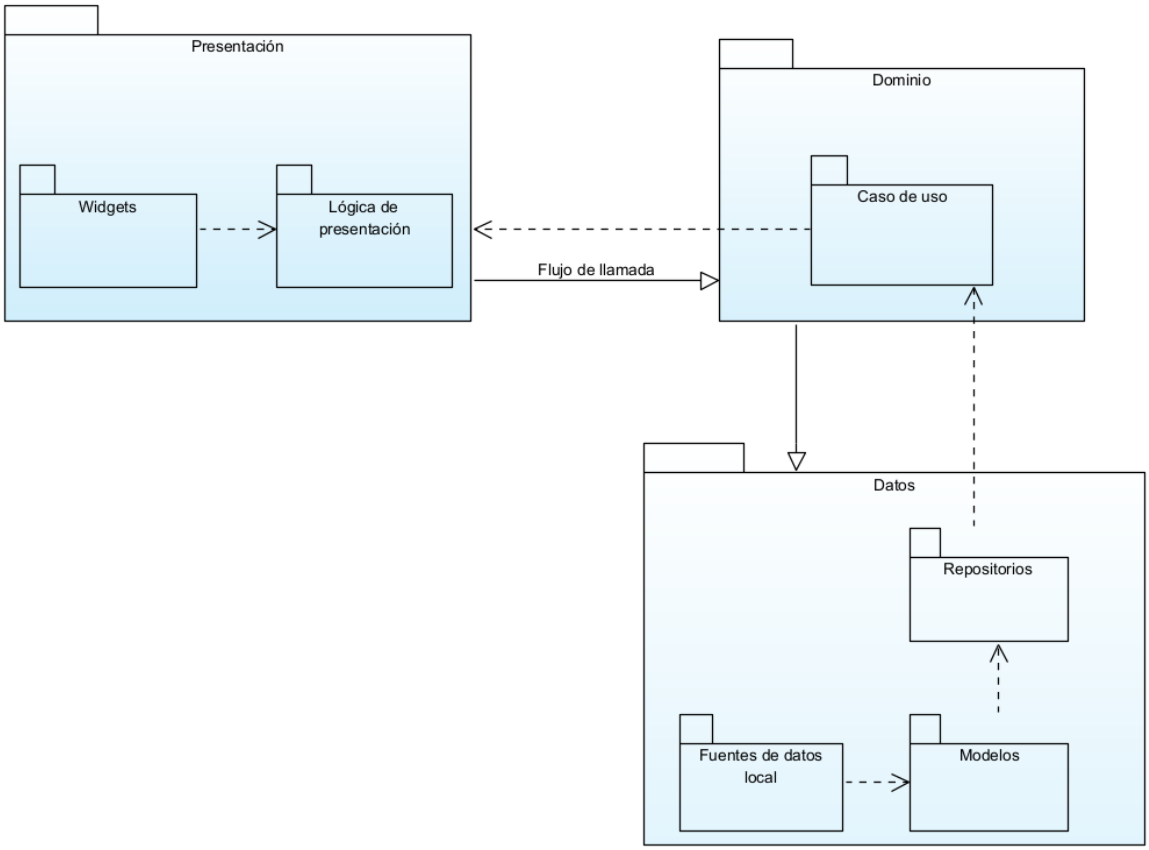


Ilustración 6. *Diagrama de Componentes de la Arquitectura Modelo-Vista-Presentador (Elaboración propia)*

"Un patrón arquitectónico es un conjunto de decisiones de diseño que se repiten en la práctica, que tienen características bien definidas y que pueden reutilizarse, describiendo así características de una arquitectura bien diferenciada (Ramirez, 2023)."

La experiencia que nos avala, nos ha hecho ver la necesidad de aprender a establecer esos buenos patrones y componentes dentro de cada Sistema de Diseño.

Un patrón es un conjunto de:

- Problema
- Solución
- Entorno

Un patrón de diseño resulta ser una solución a un problema de diseño. Describen soluciones adecuadas a problemas que ocurren repetidamente, haciendo al diseño más

.

flexible, elegante y principalmente reusable (Torresburriel, 2022). A continuación, se presentan los patrones utilizados en el desarrollo de la aplicación Android:

2.6.1 Patrones Generales de Software para la Asignación de Responsabilidades (GRASP)

Es un sistema orientado a objetos que se compone de objetos que envían mensajes a otros objetos para que lleven a cabo las operaciones requeridas. Los diagramas de interacción describen gráficamente estas operaciones, a partir de los objetos en interacción, que se responsabilizan de una actividad determinada. Los patrones GRASP describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Ayudan a entender el diseño de objeto esencial y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable (savedra, 2007).

A continuación, se muestran los patrones utilizados y un ejemplo de cómo se evidencia en la aplicación:

- **Patrón Creador:** este patrón indica que la responsabilidad de crear o instanciar nuevos objetos debe recaer sobre la clase que tiene la información necesaria para hacerlo, o que usa directamente, almacena o contiene a los objetos creados. (Mora, 2022).
- **Bajo Acoplamiento:** es la medida en que los cambios de un componente tiende a necesitar cambios de otro componente. Un bajo acoplamiento permite: entender una clase sin leer otras, cambiar una clase sin afectar a otras, mejora la mantenibilidad del código (Mora, 2022).
- **Alta Cohesión:** la cohesión es la medida en la que un componente se dedica a realizar solo la tarea para la cual fue creado, delegando las tareas complementarias a otros componentes. Una alta cohesión hace más fácil: entender qué hace una clase o método, usar nombres descriptivos y reutilizar clases o método (Mora, 2022).

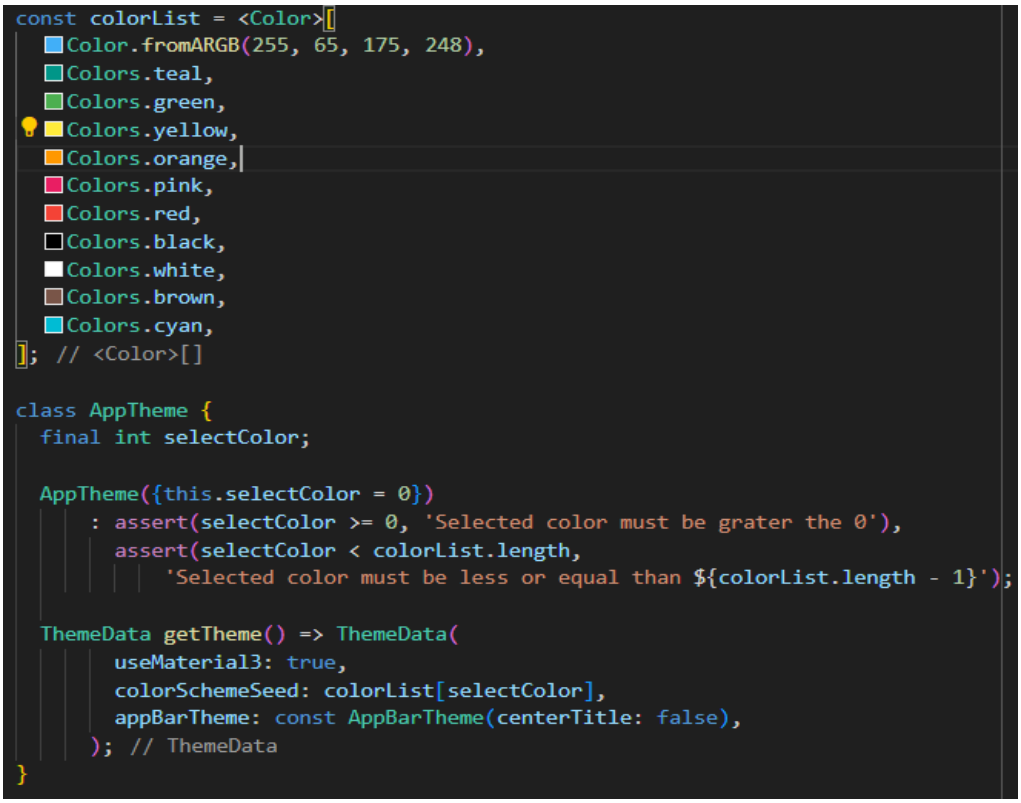
2.6.2 Patrones de Diseño GOF (The Gang of Four)

Los patrones del Grupo de los Cuatro (GOF por sus siglas en inglés) se dividen en tres categorías: patrones de creación, patrones estructurales y patrones de comportamiento:

A continuación, se describe el patrón de comportamiento utilizado en la solución:

.

- **Patrón Builder:** se encarga de la creación de objetos, abstrayendo la forma en que se crean y permitiendo tratar las clases a crear de forma genérica. Este proceso se realiza mediante un constructor virtual, que incluye como patrón de programación las características de varios objetos complejos, que llaman a esta clase para obtener la información que necesitan en el momento en que se realizan llamadas sobre ellos (Lago, 2023).



```
const colorList = <Color>[
  Color.fromARGB(255, 65, 175, 248),
  Colors.teal,
  Colors.green,
  Colors.yellow,
  Colors.orange,
  Colors.pink,
  Colors.red,
  Colors.black,
  Colors.white,
  Colors.brown,
  Colors.cyan,
]; // <Color>[]

class AppTheme {
  final int selectColor;

  AppTheme({this.selectColor = 0})
      : assert(selectColor >= 0, 'Selected color must be grater the 0'),
        assert(selectColor < colorList.length,
            'Selected color must be less or equal than ${colorList.length - 1}');

  ThemeData getTheme() => ThemeData(
        useMaterial3: true,
        colorSchemeSeed: colorList[selectColor],
        appBarTheme: const AppBarTheme(centerTitle: false),
      ); // ThemeData
}
```

***Ilustración 7**. Clase "AppTheme" evidenciando el patrón "Builder" (Elaboración propia)*

- **State:** Permite que un objeto cambie su estado de cara a otros objetos o interfaces. La programación de este patrón favorece que el cambio de estado se ejecute, se notifique y se visualice (Lago, 2023).

```
lib > config > widgets > custom_bottom_navigation.dart > ...
1   import 'package:flutter/material.dart';
2   import 'package:go_router/go_router.dart';
3
4   class CustomBottomNavigation extends StatelessWidget {
5     const CustomBottomNavigation({super.key});
6
7     int getCurrentIndex(BuildContext context) {···
24
25    void onItemTapped(BuildContext context, int index) {···
40
41    @override
42    Widget build(BuildContext context) {
43      // ignore: unused_local_variable
44      final colors = Theme.of(context).colorScheme;
45
46      return BottomNavigationBar(
47          currentIndex: getCurrentIndex(context),
48          elevation: 30,
49          iconSize: 30,
50          selectedFontSize: 15,
51          onTap: (value) => onItemTapped(context, value),
52          items: const [
53            BottomNavigationBarItem(icon: Icon(Icons.home_sharp), label: 'Home'),
54            BottomNavigationBarItem(icon: Icon(Icons.extension_sharp), label: 'Subsistemas'),
55            BottomNavigationBarItem(icon: Icon(Icons.help_outline), label: 'DISTRA'),
56          ]); // BottomNavigationBar
57    }
58  }
59
```

Ilustración 8. *Clase "CustomBottomNavigaton" evidenciando el patrón "State" (Elaboración propia)*

- **Decorador:** este tipo de patrón se enfoca más en la creación de funcionalidades de manera dinámica, evitando que se deban crear nuevas clases hijas de una clase padre cada vez que surja una nueva función (Lago, 2023).

```dart
class BottomCRM extends StatelessWidget {
  final String title;
  const BottomCRM({super.key, required this.title});

  @override
  Widget build(BuildContext context) {
    return InkWell(
      onTap: () {
        Navigator.of(context).push(MaterialPageRoute(
          builder: (context) => const CRMScreen())); // MaterialPageRout
      },
      child: Container(decoration: BoxDecoration(color: Colors.white,
            borderRadius: BorderRadius.circular(30),
            boxShadow: [BoxShadow(offset: const Offset(0, 5),
                color: Theme.of(context).colorScheme.primary.withOpaci
                spreadRadius: 2,blurRadius: 5) ], // BoxShadow // Box
        child: Column(mainAxisAlignment: MainAxisAlignment.center,
          children: [const SizedBox(height: 12),
            Text(title, style: const TextStyle(fontSize: 19),
              textAlign: TextAlign.center,
            ) // Text
          ],
        ), // Column
      ), // Container
    ); // InkWell
  }
}
```

Ilustración 9. *Clase "BottomCRM" evidenciando el patrón "Decorador" (Elaboración propia)*

- **Composite:** este patrón aplica al problema que soluciona la recursividad. Es decir, permite que un objeto se componga de otros objetos o los incluya, selecciona para cada caso cuales de ellos se fusionarán. Así un macroObjeto, se divide en partes más pequeñas e individuales, que se juntan indistintamente según la función que deben realizar.

```dart
class SubsistemasScreen extends StatelessWidget {
  static const String name = 'subsistemas_screen';
  const SubsistemasScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Theme.of(context).colorScheme.primary,
      extendBodyBehindAppBar: true,
      body: Column(
        children: [
          const SizedBox( // SizedBox ···
          Text( // Text ···
          const SizedBox(height: 10),
          Row( // Row ···
          const SizedBox(height: 12),
          Expanded(
              child: Container(
            padding: const EdgeInsets.only(top: 10),
            decoration: const BoxDecoration( // BoxDecoration ···
            child: GridView.count(
              padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 15),
              crossAxisCount: 2,
              mainAxisSpacing: 20,
              crossAxisSpacing: 28,
              children: const [···
            ), // GridView.count
          )) // Container // Expanded
        ],
      ), // Column
    ); // Scaffold
  }
}
```

Ilustración 10. *Clase "Subsistema Screen" evidenciando el patrón "Composite" (Elaboración propia)*

Conclusiones del capítulo

- Se realizó la descripción de la propuesta para obtener una solución de acuerdo con los requerimientos del cliente empleando la metodología AUP en su variación para la UCI.
- Con la descripción de la propuesta de solución se obtuvo una visión más clara y detallada del sistema que se desea desarrollar.
- Se obtuvieron también los requisitos funcionales y no funcionales, y se desarrollaron las historias de usuario

- Los patrones de diseño garantizaron la obtención de una aplicación con pocas dependencias entre las clases y la estandarización del código fuente del software, permitiendo que sea fácil el mantenimiento y evolución del mismo.
- Se definió la arquitectura a emplear, los estándares de codificación en el lenguaje seleccionado que garantizaron pautas para una legibilidad y escalabilidad mejor y el modelo de datos

Capítulo 3: Validación de la propuesta de solución

"La validación de requerimientos es un proceso continuo en el proyecto de desarrollo de software con el fin de asegurar que los requerimientos solicitados sean representaciones exactas de las necesidades y expectativas de los usuarios. Esta actividad contribuye a mejorar la calidad de los requerimientos, a reducir costos, tiempos y riesgos en el desarrollo de software (Santana, 2020)*."*

Con el objetivo de validar los requisitos se emplearon técnicas que se mencionan a continuación:

- Construcción de prototipos: es utilizado principalmente en el desarrollo de software para ofrecer al usuario una visión previa de cómo será el programa o sistema (Felipe, 2021). Se mostró al cliente un modelo ejecutable que permitió tener una visión preliminar de cómo se verían estos componentes en el sistema y a través de la interacción con los mismos se comprobó la satisfacción y aprobación del cliente, los mismos fueron aprobados satisfactoriamente.
- Aplicación de la métrica Calidad de especificación: es importante para asegurar que el producto o servicio cumpla con los estándares de calidad establecidos y que satisfaga al cliente. Con ciertas métricas se puede determinar la calidad del modelo de requisitos y la especificidad (o ambigüedad) y obtener una comprensión cuantitativa (Benjamin, 2021).

Nr: Total de requisitos de *software*

Nf: Cantidad de requisitos funcionales

Nnf: Cantidad de requisitos no funcionales

$$Nr = Nf + Nnf$$

$$Nr = 14 + 7$$

$$Nr = 21$$

Finalmente, para determinar la Especificidad de los Requisitos **(ER)** se tuvo en cuenta la ausencia de ambigüedad en los mismos (mientras más cerca de 1 esté el valor de ER, menor será la ambigüedad), para ello se realiza la siguiente operación:

ER = Nui / Nr

.

Nui: número de requisitos para los cuales todos los revisores tuvieron interpretaciones idénticas.

Para sustituir los valores de las variables en la ecuación se tuvo en cuenta que, de los requisitos especificados para el desarrollo de la propuesta de solución, dos de ellos causaron contradicción (RF12, RF13) en sus interpretaciones. Por tanto, la variable ER obtiene el siguiente valor:

ER = 19 / 21

ER = 0.90

Teniendo en cuenta el valor anterior, se arroja a un resultado positivo donde el grado de ambigüedad de los requisitos es bajo (10%) ya que el 90% son entendibles. Los requisitos ambiguos fueron modificados y validados para garantizar una correcta interpretación, llegando al resultado ideal.

Con el objetivo de validar el diseño se aplicaron métricas del diseño, las cuales permiten evaluar la calidad, la eficiencia y la efectividad de un diseño estructural. Estas métricas pueden servir para comparar distintas alternativas de diseño, optimizar los recursos empleados, verificar el cumplimiento de los requisitos normativos y mejorar el rendimiento y la seguridad de la estructura. Además, permiten medir de forma cuantitativa la calidad de los atributos internos del software, permitiendo evaluar a la vez la calidad durante el desarrollo de este (Pascual, 2016).

Las métricas de diseño que se aplicadas:

- Relación entre clases (RC)
- Tamaño Operacional de Clases (TOC).

3.3.1 Métrica relación entre clases (RC)

El resultado de la aplicación de la métrica Relación entre Clases (RC) está dado por el número de relaciones de uso que se establecen entre una clase y las demás clases existentes. Se evalúa a partir de los siguientes atributos de calidad:

- Acoplamiento: un aumento del RC implica un aumento del acoplamiento de la clase.

- Complejidad de Mantenimiento: un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
- Reutilización: un aumento del RC implica una disminución en el grado de reutilización de la clase.
- Cantidad de Pruebas: un aumento del RC implica un aumento de la cantidad de pruebas necesarias para probar una clase.

Para obtener un nivel óptimo de relación entre clases, el valor obtenido al aplicar dicha métrica debe ser directamente proporcional al acoplamiento y a la complejidad de mantenimiento. Además, debe ser inversamente proporcional al nivel de reutilización del código. Para aplicar la métrica RC es necesario categorizar cada una de las clases según la cantidad de relaciones que esta contenga.

A continuación, se muestra una tabla con las categorías para clasificar cada uno de los atributos de calidad anteriormente mencionados, así como el criterio de evaluación. En la misma se emplea la abreviatura Prom (promedio).

Tabla 4. *Métrica RC. Categoría por atributo y criterio de evaluación (Elaboración propia)*

Atributo	Categoría	Criterio
Acoplamiento	Ninguno	RC = 0
	Bajo	RC = 1
	Medio	RC = 2
	Alto	RC > 2
Complejidad de Mantenimiento	Bajo	RC <= Prom
	Medio	Prom <= RC <= 2* Prom
	Alto	RC >2* Prom
Reutilización	Bajo	RC >2* Prom
	Medio	Promedio< RC <= 2 *Promedio
	Alto	RC <= Promedio
Cantidad de pruebas	Bajo	RC <= Promedio

	Medio	Promedio<= RC <2*Promedio
	Alto	RC >= 2*Promedio

(Lorenz & Kidd, 2010)

La ilustración que se muestra a continuación representa la cantidad de clases por cantidad de dependencias que poseen.

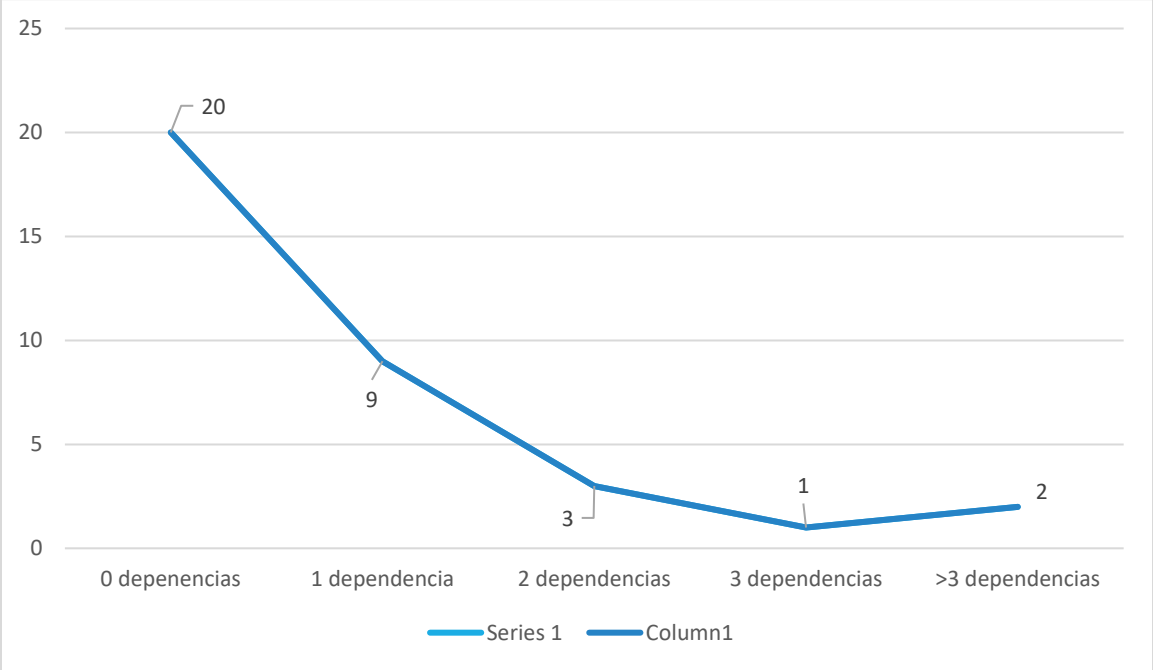


***Ilustración 11**. Clases y cantidad de dependencias (Elaboración propia)*

En la gráfica se observa que 37 del total de clases, 1 no posee ninguna dependencia, 1 clase presenta 1 sola dependencia, 3 clases presentan 2 dependencias, 9 clases con 3 dependencias y existen 20 clases con más de tres dependencias.

A continuación, se muestra en porciento (%) el nivel de acoplamiento, complejidad de mantenimiento, cantidad de pruebas y reutilización de las clases al aplicar la métrica RC.

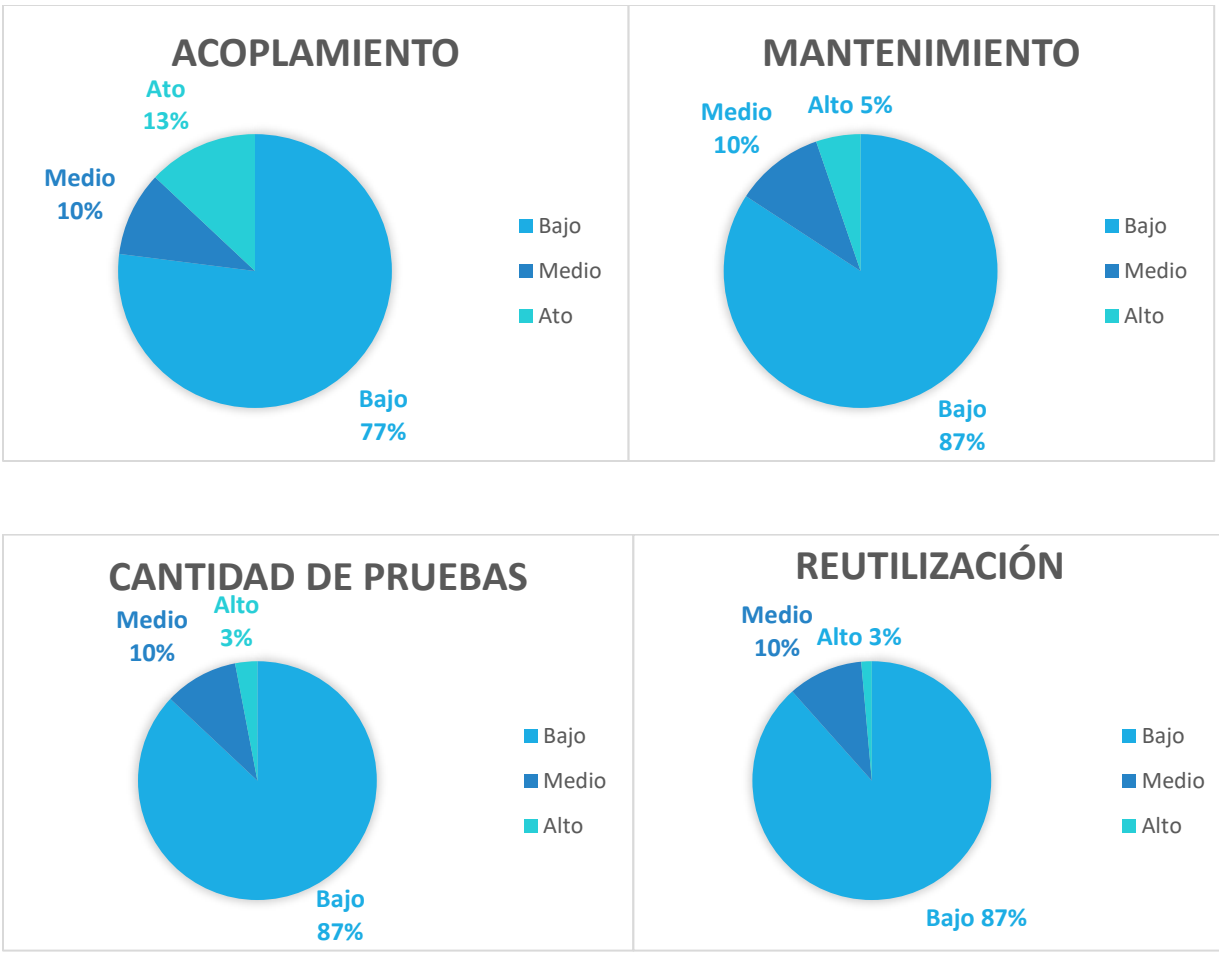


Ilustración 12. *Acoplamiento, complejidad de mantenimiento, cantidad de pruebas y reutilización de clases (Elaboración propia)*

Después de evaluar los resultados obtenidos de cada uno de los atributos de calidad que mide esta métrica, se obtuvieron los siguientes resultados:

- El 77% de las clases poseen un bajo acoplamiento.
- El 87% de las clases poseen una baja complejidad de mantenimiento y un bajo porciento de cantidad de pruebas.
- El 87% de las clases poseen una alta reutilización.

Como se puede observar en la figura existe un bajo nivel de acoplamiento por lo que hay poca dependencia entre las clases trayendo como consecuencia una alta probabilidad de reutilización. También se puede apreciar que existe un bajo nivel de complejidad de mantenimiento por lo que a la hora de optimizar métodos y demás operaciones no es necesario realizar una gran cantidad de pruebas, lo que minimiza el tiempo de implementación y pruebas de los componentes propuestos.

3.3.2 Métrica tamaño operacional de clases (TOC)

Las métricas de tamaño operacional de clases se basan en la medición de las características y atributos de las clases en un sistema de software. Estas métricas proporcionan una manera cuantitativa de evaluar el tamaño relativo y la complejidad de las clases, lo que puede ser útil para comprender la estructura y el diseño del sistema, así como para identificar posibles problemas de mantenibilidad y calidad del código (Ramírez, 2023).

Se evalúa a partir de los siguientes atributos de calidad:

- Responsabilidad
- Complejidad y de implementación
- Reutilización

Una vez analizado el indicador tamaño de clase, si el valor resultante tiende al crecimiento, es probable que la clase posea un alto grado de responsabilidad; en consecuencia, el nivel de reutilización sería mínimo y la implementación altamente compleja.

Para medir los atributos de calidad se definieron los umbrales que se muestran en la Tabla 8.

En la misma se emplean la abreviatura Prom (promedio).

Tabla 5. *Métrica TOC. Categoría por atributos y criterio de evaluación (Elaboración propia)*

Atributo	Categoría	Criterio
Responsabilidad	Bajo	TOC < =Promedio (Prom)
	Medio	TOC Entre Prom. y 2* Prom
	Alto	TOC >2*Prom
Complejidad de Implementación	Bajo	TOC < =Prom
	Medio	Entre Prom. y 2* Prom
	Alto	TOC >2*Prom
Reutilización	Bajo	TOC >2*Prom

	Medio	TOC Entre Prom. y 2* Prom
	Alto	TOC < =Prom

(Lorenz, Mark ; Kidd, Jeef, 2010)

La ilustración que se muestra a continuación representa la cantidad de procedimientos de cada una de las clases del diseño:

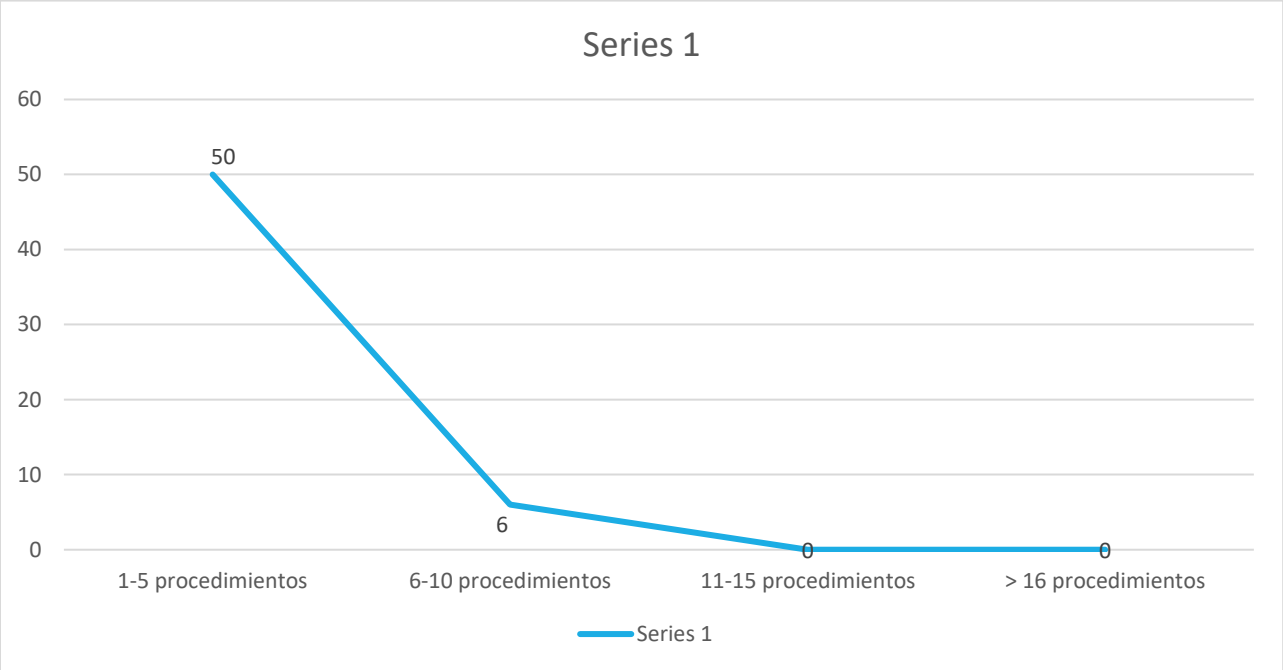


Ilustración 13. *Representación de la cantidad de clases por cantidad de procedimientos (Elaboración propia)*

Como se muestra en la figura anterior, cincuenta y ocho (50) del total de las clases poseen entre 1 y 5 procedimientos, y por último siete (6) clases del total poseen entre 6 y 10 procedimientos. A continuación, se muestra en porciento (%) el nivel de complejidad,

responsabilidad y reutilización de las clases al aplicar la métrica TOC.

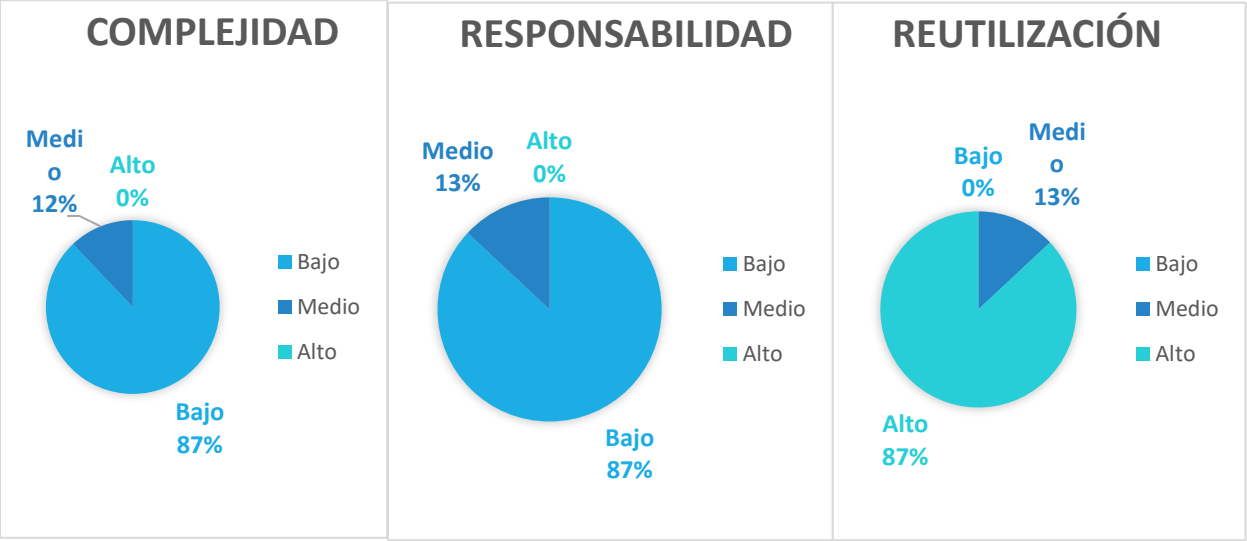


Ilustración 14. *Responsabilidad, complejidad y reutilización de las clases (Elaboración propia)*

Después de una evaluación de los resultados obtenidos de cada uno de los atributos de calidad que mide esta métrica, los resultados obtenidos:

- El 87% de las clases poseen una baja responsabilidad y complejidad de implementación.
- El 80% de las clases poseen una alta reutilización.

Se demuestra con estos resultados la calidad del diseño de la solución propuesta. Al obtener bajos índices de complejidad y responsabilidad, conjunto a un elevado grado de reutilización de las clases, se proporciona en gran medida la implementación de los componentes como resultados de la propuesta de solución.

Las pruebas de software son un conjunto de procesos con los que se pretende probar un sistema o aplicación en diferentes momentos para comprobar su correcto funcionamiento. Este tipo de pruebas abarca cualquier estadio del desarrollo del sistema, desde su creación hasta su puesta en producción. La organización del proceso de pruebas se realiza a través de cuatro niveles: unitarias, integración, sistema y aceptación (Turrado, 2022).

"Las pruebas de software son el mejor método para confirmar la funcionalidad del producto que se está desarrollando. Deben aplicarse en todas las fases del desarrollo, desde el inicio del proyecto hasta el final, de manera que puedan descubrirse fallos aislados en un primer momento o detectar un comportamiento erróneo al final (Herrera, 2022)*."* Este proceso de pruebas conduce a dos metas diferentes:

- Demostrar al desarrollador y al cliente que el software cumple con lo pactado (requisitos funcionales y no funcionales) (Verificación).
- Encontrar situaciones donde el comportamiento del software no sea el correcto (Validación).

Se pretende desarrollar dos tipos de prueba fundamentalmente en la fase de pruebas internas que propone AUP-UCI sustentando ambas en los diseños de casos de prueba, pruebas de caja blanca y pruebas de caja negra que serán descritas a continuación siguiendo los principios de pruebas (Ramiro, 2023):

- El probador debe tener acceso al código fuente y al diseño del software.
- El probador debe utilizar herramientas de análisis estático y dinámico para medir la cobertura del código y detectar errores.
- El probador debe verificar que el software cumple con los requisitos funcionales y no funcionales especificados.
- El probador no necesita tener acceso al código fuente ni al diseño del software.
- El probador debe diseñar casos de prueba que cubran todas las posibles entradas, salidas y escenarios del software.
- El probador debe verificar que el software se comporta según lo esperado y que no presenta defectos ni vulnerabilidades.

Para la validación de la solución propuesta, se define una estrategia de pruebas de softwares aplicadas a partir de las disciplinas establecidas para el desarrollo de las pruebas por la metodología AUP variación UCI. En la estrategia se definen las disciplinas de Pruebas internas a nivel de unidad y de sistema, con la aplicación de sus respectivos métodos y técnicas, y la disciplina de Pruebas de aceptación que plantea dicha metodología, para verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido.

.

A continuación, se describen los resultados obtenidos durante la disciplina de Pruebas internas en los niveles descritos anteriormente.

3.3.1 Pruebas Internas

Las pruebas unitarias de software son el instrumento utilizado para validar un fragmento de código fuente. Los desarrolladores aíslan una línea del lenguaje codificado para saber si el sistema está operando correctamente en una función, proceso o actividad específica.

Son aplicadas acorde a las etapas de desarrollo del software. Por lo regular, los test unitarios se utilizan en fases iniciales, antes de integrar fragmentos de mayor tamaño en el sistema (Tomoshi, 2022).

Para la ejecución de estas pruebas se deben desarrollar artefactos de apoyo, tales como: diseños de casos de prueba, listas de chequeo y de ser posible componentes de prueba ejecutables para automatizar el proceso. Para la validación de la aplicación, las pruebas internas se ejecutan a nivel de unidad y sistema. A continuación, se detalla cómo se llevaron a cabo dichas pruebas.

Pruebas Unitarias

Las pruebas unitarias consisten en aislar una parte del código y comprobar que funciona a la perfección. Son pequeños tests que validan el comportamiento de un objeto y la lógica. Estas pruebas son una manera de asegurarse de que el código funcione correctamente (Uber, 2023).

Método Caja blanca: para la aplicación de las pruebas unitarias, se decide utilizar este método ya que permite examinar el código fuente y la estructura interna del programa, para verificar que cumple con los requisitos y funciones esperados. Posibilita desarrollar casos de prueba que garanticen la ejecución, al menos una vez, de las rutas independientes (Fernandez, 2023). Para aplicar este método se define la técnica de ruta básica.

Técnica de ruta básica: permite obtener una medida de la complejidad lógica de un diseño procedimental, y usar esa medida como guía para la definición de un conjunto básico de diseños de casos de pruebas de caminos de ejecución. Los casos de prueba derivados del conjunto garantizan que durante la prueba se ejecuta por lo menos una

vez cada sentencia del programa. Tiene como objetivo comprobar que cada camino se ejecute independientemente de un componente o programa (Pérez, 2015).

Para la aplicación de la ruta básica, Pressman propone realizar un análisis de la complejidad ciclomática de cada procedimiento que componen las clases del sistema; así como la selección del método con mayor valor de complejidad ciclomática, que es el que tiene más probabilidad de contener errores. Además, ofrece una medida del número de pruebas que deben diseñarse para validar la correcta implementación de una determinada función (Silva, 2023).

A partir de esta técnica, para la obtener los casos de prueba se debe construir el grafo de flujo para cada uno de los procedimientos desarrollados, a continuación, se muestra un ejemplo de la aplicación de la técnica correspondiente al código del método: *getCurrentIndex()* que es el encargado de dar la información acerca de cuál es el índice seleccionado para luego mostrarlo en el bottomNavigatonBar().

```
9   int getCurrentIndex(BuildContext context) {
10    final String location = GoRouterState.of(context).matchedLocation;
11    int caso = 0;
12
13    if (location == '/')
14      caso = 1;
15    else if (location == '/subsistemas')
16      caso = 2;
17    else if (location == '/distra')
18     caso = 3;
19
20    return caso;
21  }
22
```

Ilustración 15. Código del método HomeScreen() (Elaboración propia)

La métrica propuesta se fundamenta con teoría de grafos y se puede realizar mediante tres formas distintas:

- V(G) = R
- V(G) = E – N + 2
- V (G) = P + 1

.

Donde:

G: Grafo de flujo (grafo)

V(G): Complejidad ciclomática para un grafo G

R: El número de regiones

E: Número de aristas

N: Número de nodos del grafo

P: Número de nodos predicados incluidos en el grafo

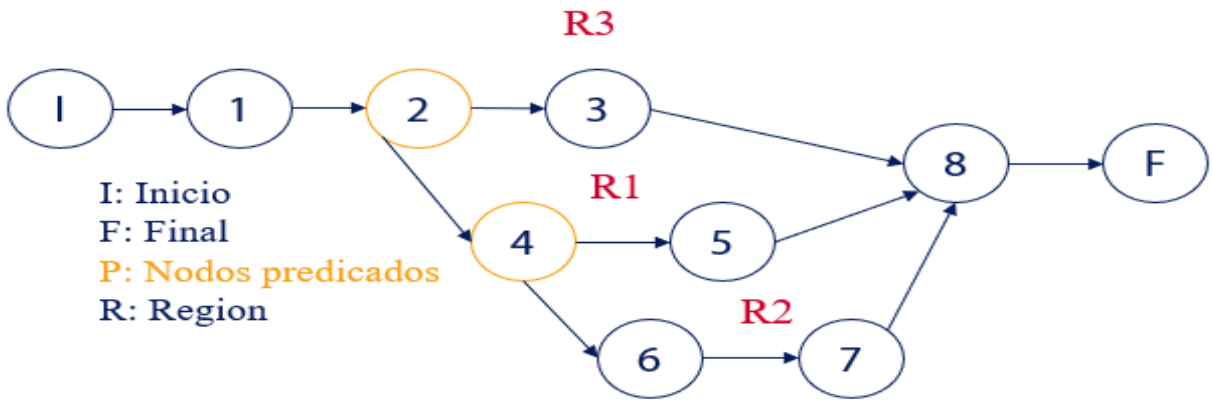


***Ilustración 16**. Grafo del flujo getCurrentIndex() (Elaboración propia)*

Realizando los cálculos correspondientes se obtiene por cualquiera de las vías conocidas el siguiente resultado:

V(G) = R	V(G) = E – N + 2	V(G) = P + 1
V(G) = 3	V(G) = 11 – 10 + 2	V(G) = 2 + 1
	V(G) = 3	**V(G) = 3**

Después de calcular la complejidad del grafo, se pudo comprobar que el resultado obtenido es igual a 3; los conjuntos de caminos básicos son:

Camino 1: I →1 → 2 → 3 → 8 → F

- Este camino cubre el caso en el que location es igual a '/'.

Camino 2: I →1 → 2 → 4 → 5 → 8 → F

- Este camino cubre el caso en el que location es igual a '/subsistemas'.

Camino 3: I →1 → 2 → 4 → 6 → 7 → 8 → F

- Este camino cubre el caso en el que location es igual a '/distra'.

.

Una vez definidas las rutas, se procede a diseñar los casos de prueba para cada una de las rutas básicas obtenidas. A continuación, se presentan los casos de prueba de cada ruta.

Tabla 6. *Caso de prueba para la ruta básica 1 (Elaboración propia)*

Caso de prueba para la ruta básica 1	
Descripción: Prueba para el caso en el que location es igual a '/'. Cuando el usuario presione el botón de home, se hará una navegación a la pantalla de inicio, definida con el path: '/'. Visualizando toda la información contenida en la pantalla.	
Entradas	location = '/'.
Condición de ejecución	El valor de location coincide con el caso case '/'.
Resultados esperados	El método getCurrentIndex debe devolver el valor 1.

Tabla 7. *Caso de prueba para la ruta básica 2 (Elaboración propia)*

Caso de prueba para la ruta básica 2	
Descripción: Prueba para el caso en el que location es igual a '/subsistemas'. Cuando el usuario presiona el botón de subsistema se le muestran las HU definidas en la tabla 1. A través de esta interfaz el usuario podrá visualizar las diferentes HU y navegar al módulo seleccionado, recibiendo una previsualización de lo que contendrá ese modulo en un mensaje emergente que se lanza al presionar dicho elemento.	
Entradas	location = '/subsistemas'.
Condición de ejecución	El valor de location coincide con el caso case '/subsistemas'.
Resultados esperados	El método getCurrentIndex debe devolver el valor 2.

Tabla 8. *Caso de prueba para la ruta básica 3 (Elaboración propia)*

Caso de prueba para la ruta básica 3
Descripción: Prueba para el caso en el que location es igual a '/distra'. Al presionar el botón DISTRA, el usuario tendrá acceso a una página donde podrá documentarse sobre todo lo referente a este sistema de manera general. Podrá saber desde que

componentes tiene el sistema hasta cuales son todas sus herramientas. Además de visualizar todas las vías disponibles para poder establecer contacto con la oficina que se encarga del despliegue de DISTRA.	
Entradas	location = '/distra'.
Condición de ejecución	El valor de location coincide con el caso case '/distra'.
Resultados esperados	El método getCurrentIndex debe devolver el valor 3.

Se puede concluir que luego de realizar la prueba de caja blanca y aplicar los casos de prueba expuestos anteriormente se evidenció que el flujo de trabajo de las funcionalidades es correcto. Se logró asegurar el cumplimiento del proceso de mejora del código, se comprobó que cada sentencia fuese ejecutada al menos una vez, se pudo mejorar el código fuente y reducir la complejidad ciclomática del sistema.

3.3.2 Pruebas Funcionales

Las pruebas funcionales son una práctica beneficiosa cuando se refiere al proceso del desarrollo. De esta manera se puede tener el progreso del proyecto para la administración de las pruebas funcionales aprobadas (Vargas C. , 2023). Estas pruebas componen un proceso, donde las mismas se derivan a partir de la especificación del sistema. El sistema se trata como una “caja negra” cuyo comportamiento sólo se puede determinar por el estudio de entradas y salidas (Pressman, 2014).

Método caja negra: el método de caja negra es una técnica de prueba que se utiliza para evaluar el funcionamiento de un sistema o componente sin tener en cuenta su estructura interna o detalles de implementación. El objetivo es probar el sistema desde una perspectiva externa, centrándose en validar la funcionalidad y el comportamiento del sistema sin conocer los detalles internos de su diseño o implementación (Pressman, 2014).

Partición de equivalencia: es una técnica utilizada en pruebas de software para reducir el número de casos de prueba necesarios mientras se mantiene una cobertura adecuada. Se basa en dividir el conjunto de datos de entrada en grupos o particiones que se consideran equivalentes, lo que implica que, si un caso de prueba dentro de una partición funciona correctamente, se espera que los demás casos de prueba en la misma

partición también funcionen correctamente. El objetivo principal es identificar conjuntos representativos de casos de prueba que cubran diferentes escenarios de entrada y que sean, en cierto sentido, "equivalentes" en términos de comportamiento (Myers, 2011).
Para aplicar esta técnica, se deben primeramente realizar el diseño de casos prueba (DCP) con el objetivo de obtener un conjunto de pruebas que tengan la mayor probabilidad de descubrir los defectos del software.
Un diseño de caso de prueba se refiere al proceso de identificar y definir los casos de prueba que se utilizarán para probar un sistema o componente de software. Garantiza que se cubran adecuadamente los diferentes escenarios y funcionalidades del sistema. (Mullin, 2021). Como resultado de aplicar la técnica partición equivalente se realizaron tres iteraciones de pruebas, para poder alcanzar resultados satisfactorios, atendiendo al correcto comportamiento del sistema ante diferentes situaciones y obteniéndose los resultados que se muestran en la siguiente figura:

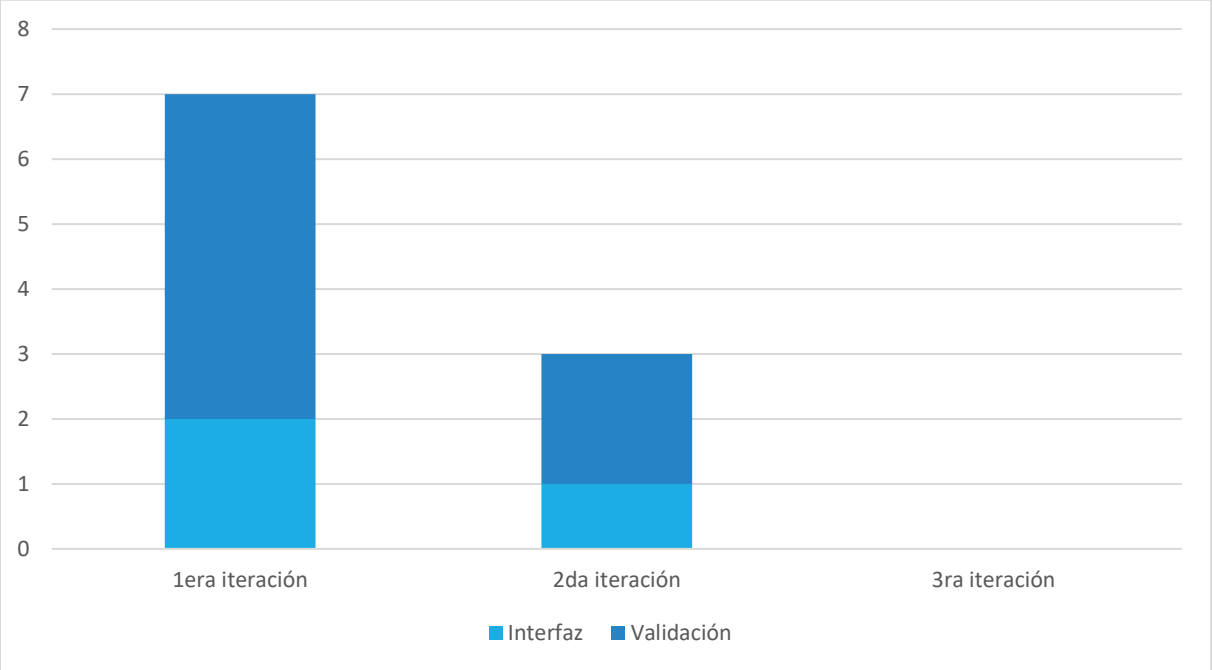


Ilustración 17. *Representación de la cantidad de NC por iteraciones (Elaboración propia)*

En una primera iteración se detectaron un total de siete No Conformidades (5 de

validación y 2 de interfaz). En la segunda iteración se manifestaron tres NC (2 de validación y 1 de interfaz). Finalmente, en la tercera iteración se obtuvieron resultados satisfactorios al no mostrar NC.

Conclusiones del capítulo

- En el este capítulo se utilizaron las técnicas de validación de requisitos, permitiendo corroborar la correspondencia de los mismos con las solicitudes del cliente.
- En la validación del diseño se utilizaron las métricas RC y TOC, donde se obtuvo el grado de complejidad de implementación, mantenimiento, responsabilidad, reutilización, acoplamiento y la cantidad de pruebas necesarias a realizar.
- A través del método de caja blanca, se comprobaron internamente las funciones de los componentes de la aplicación, proporcionando la detección de fallos para su corrección.
- El método de caja negra posibilitó evidenciar que las funciones son operativas a través de la interfaz del software.
- La realización de la encuesta a los funcionarios que trabajan en el despliegue del sistema, aportó datos significativos sobre el grado de satisfacción de los clientes.
- La validación de las variables de la investigación arrojó resultados satisfactorios de la aceptación de la aplicación.

Capítulo 3: Validación de la propuesta de solución

Conclusiones finales

La investigación llevada a cabo y los resultados obtenidos permiten llegar a las conclusiones:

- La definición del marco teórico de la investigación orientado al desarrollo de aplicaciones móviles destinadas a la capacitación, proporcionó los fundamentos necesarios para abordar la solución propuesta, lo cual facilitó la adquisición de los conocimientos requeridos.
- La selección de la metodología permitió estructurar, planificar y controlar el proceso de desarrollo de la aplicación móvil que se propone como solución.
- La especificación clara y precisa de los requisitos, en conjunto con el análisis y diseño detallado del sistema, permitió obtener una aproximación concreta y bien definida de los elementos esenciales que se necesitan para la implementación de la aplicación móvil para DISTRA.
- La aplicación de las métricas de validación y pruebas de software brindaron una medida cuantitativa que ha permitido comprobar la calidad de los artefactos generados durante el desarrollo de la solución propuesta.

Recomendaciones

Se recomienda para investigaciones futuras:

- Agregar una autenticación a la aplicación. De manera tal que se tenga un control sobre todos los usuarios que emplean el sistema a través de un registro dado por la aplicación.
- Se propone la implementación de un botón en la aplicación que facilite a los usuarios el acceso inmediato al sitio oficial del sistema.

Referencias bibliográficas

Ana. (18 de 04 de 2023). *las tendencias actuales en capacitacion que debemos saber.* Obtenido de TeachPlace: https://teach.place/2023/04/18/las-tendencias-actuales-en-capacitacion-que-debemos-saber/#:~:text=Las%20tendencias%20actuales%2C%20como%20el%20aprendizaje%20basado%20en,el%20aprendizaje%20y%20el%20desarrollo%20de%20los%20empleados.

Barriento, C. (28 de 09 de 2023). *patrones arquitectonicos mvc mvp y mvvm.* Obtenido de appmaster: https://appmaster.io/es/blog/patrones-arquitectonicos-mvc-mvp-y-mvvm

Benjamin. (15 de 07 de 2021). *tolerancias y especificaciones.* Obtenido de IngienreriaOnline: https://www.ingenieriaonline.com/tolerancias-y-especificaciones/

Carlos. (27 de 03 de 2023). *pruebas de software servicios de informatica profesion.* Obtenido de LinKedIn: https://es.linkedin.com/pulse/pruebas-de-software-servicios-de-informatica-profesion

creately. (18 de 10 de 2022). *tutorial de diagrama de despliegue*. Obtenido de creately: https://creately.com/blog/es/diagramas/tutorial-de-diagrama-de-despliegue/

Cristancho, F. (26 de 07 de 2022). *que es flutter*. Obtenido de TALENTLY BLOG: https://talently.tech/blog/que-es-flutter/

Edraw. (16 de 03 de 2023). *crc card software*. Obtenido de wondershare: https://www.edrawsoft.com/es/crc-card-software.html

Errázuriz, J. G. (2023). *6 caracteristicas esenciales de una capacitacion de calidad*. Obtenido de Capacitación y Desarrollo UC: https://capacitacion.uc.cl/noticias/168-6-caracteristicas-esenciales-de-una-capacitacion-de-calidad

Felipe. (6 de 07 de 2021). *modelo de prototipos que es y cuales son sus etapas.* Obtenido de hostingplus: https://www.hostingplus.pe/blog/modelo-de-prototipos-que-es-y-cuales-son-sus-etapas/

Fernandez, E. (11 de 04 de 2023). *que son pruebas de caja blanca.* Obtenido de KeepCoding: https://keepcoding.io/blog/que-son-pruebas-de-caja-blanca/

Garcia, J. (6 de 12 de 2021). *que es un modelo de datos base de datos.* Obtenido de Tecno-Simple: https://tecno-simple.com/que-es-un-modelo-de-datos-base-de-datos/

Garcia, N. (15 de 07 de 2022). *porque es importante la capacitacion para el beneficio de los empleados*. Obtenido de Time.Ly: https://time.ly/es/blog/%C2%BFPor-qu%C3%A9-es-importante-la-capacitaci%C3%B3n-para-los-beneficios-de-los-empleados%3F/

Gomez, C. (26 de 10 de 2022). *pruebas de caja blanca.* Obtenido de Diario de QA: https://www.diariodeqa.com/post/pruebas-de-caja-blanca

Referencias bibliográficas

Gonzales. (3 de 05 de 2023). *que son las pruebas unitarias de software.* Obtenido de KEECODING: https://keepcoding.io/blog/que-son-las-pruebas-unitarias-de-software/

Gonzales, P. (29 de 04 de 2019). *habanatrans aplicacion funciona a pesar de trabas*. Obtenido de CUBANET: https://www.cubanet.org/destacados/habanatrans-aplicacion-funciona-a-pesar-de-trabas/#:~:text=Se%20trata%20de%20un%20software,ubicaci%C3%B3n%20por%20medio%20de%20GPS.

Henandez, J. A. (13 de 08 de 2020). *educacion a distancia virtual y en linea cual es la diferencia.* Obtenido de Docentes Al Dia: https://docentesaldia.com/2020/08/09/educacion-a-distancia-virtual-y-en-linea-cual-es-la-diferencia/

Hernandez, R. D. (18 de 06 de 2021). *el modelo de arquitectura view controller pattern.* Obtenido de freecodecamp: https://www.freecodecamp.org/espanol/news/el-modelo-de-arquitectura-view-controller-pattern/

Herrera, C. (26 de 08 de 2022). *pruebas software.* Obtenido de unir: https://www.unir.net/ingenieria/revista/pruebas-software/

Issel. (18 de 08 de 2016). *artex en tu elefono*. Obtenido de Arte por Excelencias : https://www.arteporexcelencias.com/es/noticias/2016-08-18/artex-en-tu-telefono.html

JBenaQuir. (2012). *Diagrama_de_despliegue*. Obtenido de Ecured: https://www.ecured.cu/Diagrama_de_despliegue

Julian. (1 de 06 de 2022). *apps-e learning para empresas*. Obtenido de UBITS: https://www.ubits.com/contenidos/apps-e-learning-para-empresas

Kenguan, C. A. (2021). *QUE ES MAPA DE NAVEGACION*. Obtenido de academia: https://www.academia.edu/8402805/QUE_ES_MAPA_DE_NAVEGACION

Kidd, M. L. (2010). *Object Orientd Software Metrics.* Prentice Hall.

Lago, N. (25 de 09 de 2023). *patrones diseno de software.* Obtenido de saaSRadar: https://saasradar.net/patrones-diseno-de-software/

Lorenz, M., & Kidd, J. (2010). Object - Oriented Software Metrics. *Prentice-Hall*.

Luiso. (9 de 01 de 2019). *como medir la eficacia de la capacitacion*. Obtenido de Wilsoft: https://wilsoft-la.com/como-medir-la-eficacia-de-la-capacitacion/#:~:text=La%20eficacia%20del%20proceso%20de,la%20efectividad%20de%20forma%20num%C3%A9rica.

Marcello Visconti, H. A. (3 de 03 de 2012). *browse?type=subject&order=ASC&rpp=20&value*. Obtenido de Repositorio institucional de la Universidad de Las Tunas: http://roa.ult.edu.cu/jspui/handle/123456789/284/browse?type=subject&order=ASC&rpp=20&value

Referencias bibliográficas

Marcos. (8 de 12 de 2018). *tarjetas crc clase responsabilidad colaborador*. Obtenido de Medium: https://medium.com/@marcosrrg9813/tarjetas-crc-clase-responsabilidad-colaborador-81924cec3af0#id_token=eyJhbGciOiJSUzI1NiIsImtpZCI6IjdkMzM0NDk3NTA2YWNiNzRjZGVlZGFhNjYxODRkMTU1NDdmODM2OTMiLCJ0eXAiOiJKV1QifQ.eyJpc3MiOiJodHRwczovL2FjY291bnRzLmdvb2dsZS5jb20iL

Maria, D. J. (13 de 06 de 2018). *la-importancia-de-la-capacitacion-para-las-y-los-trabajadores*. Obtenido de Gobierno de Mexico: https://www.gob.mx/profedet/es/articulos/la-importancia-de-la-capacitacion-para-las-y-los-trabajadores?idiom=es#:~:text=La%20capacitaci%C3%B3n%20juega%20un%20papel,trabajo%20que%20se%20les%20encomienda.

Martínez, O. (14 de 07 de 2020). *capacitacion virtual a distancia.* Obtenido de educaline: https://www.educaline.com/blog/capacitacion-virtual-a-distancia/

Melo. (04 de 2023). *ventajas-capacitacion-virtual-empresas.* Obtenido de UBITS: https://www.ubits.com/contenidos/ventajas-capacitacion-virtual-empresas

Melo, J. (4 de 05 de 2023). *ventajas-capacitacion-virtual-empresas*. Obtenido de UBITS: https://www.ubits.com/contenidos/ventajas-capacitacion-virtual-empresas

Melo, S. (18 de 01 de 2022). *como-recolectar-informacion-offline*. Obtenido de DataScope: https://datascope.io/es/blog/como-recolectar-informacion-offline/

Mora, R. C. (03 de 07 de 2022). *grasp.* Obtenido de Adictos al trabajo: https://www.adictosaltrabajo.com/2003/12/22/grasp/

Mullin, W. M. (21 de 05 de 2021). *how to write test cases for software examples tutorial*. Obtenido de PARASOFT: https://es.parasoft.com/blog/how-to-write-test-cases-for-software-examples-tutorial/

Myers, G. J. (2011). *The Art of Software Testing .* wiley.

Nazarevich, D. (16 de 11 de 2022). *best software architecture patterns.* Obtenido de innowise: https://innowise-group.com/es/blog/best-software-architecture-patterns/

Pascual, R. V. (2016). Evolución de los parámetros geométricos de diseño en puentes construidos con voladizos sucesivos in situ. *UIS Ingienería*.

Peláez, J. (18 de 04 de 2009). *arquitectura basada en componentes.* Obtenido de geeks: https://geeks.ms/jkpelaez/2009/04/18/arquitectura-basada-en-componentes/

Pérez, C. J. (2015). *ingenieria software prueba caja blanca y camino basico.* Obtenido de Monografias: https://www.monografias.com/docs113/ingenieria-software-prueba-caja-blanca-y-camino-basico/ingenieria-software-prueba-caja-blanca-y-camino-basico

Pressman. (2014). *Ingeniería de software: el enfoque de un profesional.* New York: McGraw-Hill Education.

que-es-android-sdk. (15 de 07 de 2022). Obtenido de KeepCoding Tech School: https://keepcoding.io/blog/que-es-android-sdk/

Ramirez, E. R. (28 de 02 de 2023). *patrones en la arquitectura de software .* Obtenido de Linkedin: https://es.linkedin.com/pulse/patrones-en-la-arquitectura-de-software-elmo-renato-castro-ramirez

Ramírez, J. G. (25 de 10 de 2023). *metricas de evaluacion de modelos en el aprendizaje automatico*. Obtenido de DataSources.ai: https://www.datasource.ai/es/data-science-articles/metricas-de-evaluacion-de-modelos-en-el-aprendizaje-automatico

Ramiro, G. (6 de 01 de 2023). *que es black box testing o pruebas de caja negra.* Obtenido de OpenWebinars: https://openwebinars.net/blog/que-es-black-box-testing-o-pruebas-de-caja-negra/

Rodríguez, H. (16 de 06 de 2023). *tendencias de capacitacion.* Obtenido de Future of people: https://www.crehana.com/blog/gestion-talento/tendencias-de-capacitacion/

Santana, S. (7 de 09 de 2020). *SEDICI.* Obtenido de SEDICI: http://sedici.unlp.edu.ar/handle/10915/103994#:~:text=La%20validaci%C3%B3n%20de%20requerimientos%20es,y%20expectativas%20de%20los%20usuarios.

savedra, J. (8 de 05 de 2007). *Patrones_de_Asignaci%C3%B3n_de_Responsabilidades.* Obtenido de Ecured: https://www.ecured.cu/Patrones_de_Asignaci%C3%B3n_de_Responsabilidades

Schiaffarino, A. (12 de 03 de 2019). *modelo cliente servidor.* Obtenido de infranetworking: https://blog.infranetworking.com/modelo-cliente-servidor/

Silva, Y. M. (2023). Pruebas Internas. *Universidad de las Ciencias Informaticas*, 64-69.

Sumit Dutta, A. G. (8 de 04 de 2021). *everyone is within learning distance building skills remotely.* Obtenido de MCKinsey & Company: https://www.mckinsey.com/capabilities/operations/our-insights/everyone-is-within-learning-distance-building-skills-remotely/es-CL

Team, K. (10 de 04 de 2023). *que es el modelo vista presentador*. Obtenido de KeepCoding Tech School: https://keepcoding.io/blog/que-es-el-modelo-vista-presentador/

Tomoshi. (11 de 07 de 2022). *pruebas unitarias de software.* Obtenido de Testing It: https://www.testingit.com.mx/blog/pruebas-unitarias-de-software

Torresburriel. (14 de 03 de 2022). *que son los patrones de diseno y por que utilizarlos*. Obtenido de Torresburriel Estudio: https://torresburriel.com/weblog/que-son-los-patrones-de-diseno-y-por-que-utilizarlos/

Turrado, J. (26 de 09 de 2022). *que son las pruebas de software.* Obtenido de CampusMVP.es: https://www.campusmvp.es/recursos/post/que-son-las-pruebas-de-software.aspx

.

Referencias bibliográficas

Uber, C. (22 de 07 de 2023). *pruebas unitarias.* Obtenido de Asistencia472: https://www.asistencia472.com/tecnologia/pruebas-unitarias/

Vargas, C. (30 de 09 de 2023). *tipos de pruebas funcionales*. Obtenido de trycore: https://trycore.co/transformacion-digital/tipos-de-pruebas-funcionales/

Vargas, T. (2021). *define-un-plan-estrategico-de-capacitacion-en-tu-empresa*. Obtenido de TramitApp: https://www.tramitapp.com/blog/define-un-plan-estrategico-de-capacitacion-en-tu-empresa/#:~:text=Un%20plan%20estrat%C3%A9gico%20de%20capacitaci%C3%B3n%20es%20un%20documento%20que%20establece,los%20trabajadores%20de%20la%20empresa.

Zambrano, O. (2020). *que son y para que sirven las tarjetas crcs pdf free.* Obtenido de PDFECOFFEE: https://pdfcoffee.com/que-son-y-para-que-sirven-las-tarjetas-crcs-pdf-free.html

Zúñiga, F. G. (03 de 03 de 2020). *arquitectura software*. Obtenido de arsys: https://www.arsys.es/blog/arquitectura-software

Anexos

Anexo 1. Entrevista al cliente para conocer la necesidad del desarrollo de la propuesta de solución y definir los requisitos funcionales y no funcionales.

Guía de preguntas utilizadas en el desarrollo de la entrevista con los profesionales del centro de implantación de DISTRA.

1. ¿Considera que los clientes tienen el conocimiento necesario del Sistema de Gestión Empresarial DISTRA?
2. Respecto a la capacitación empresarial, ¿qué porciento de clientes se les brinda buena documentación para la capacitación de Sistema?
3. ¿Cuáles son las causas que podrían estar influyendo en el desarrollo de esta habilidad?
4. ¿Existe alguna herramienta en el centro para el diseño de una aplicación Android?
5. ¿Cuáles son sus características?
6. A partir de estas características, ¿considera que la herramienta posee lo necesario para brindar capacitación sobre el Sistema de Gestión Empresarial DISTRA?
7. ¿Considera que la aplicación Android debe tener toda la información para la capacitación de los clientes?
8. ¿Qué otras características, considera que deba presentar la aplicación Android, en cuanto a la usabilidad, seguridad, interfaz u otro aspecto que garantice su calidad?