Revista RITI
Investigación en Tecnologías de la Información

Revista de Investigación en Tecnologías de la Información
ISSN: 2387-0893
revista.riti@gmail.com
Universitat Politècnica de Catalunya
España

Basilio López, Mario Andrés; López Díaz, Samuel Erasto; Pérez Sibaja,
José Alejandro; Reyes Rodríguez, Guillermo Eduardo; Salvador Nolasco,
José Ricardo; Toledo Toledo, Guadalupe; Arellano Pimentel, J. Jesús

Aplicación móvil para el análisis de grafos bajo un enfoque de Ingeniería de Software

Revista de Investigación en Tecnologías de la Información,
vol. 10, núm. 22, 2022, Julio-Diciembre, pp. 103-117
Universitat Politècnica de Catalunya
España

DOI: https://doi.org/10.36825/RITI.10.22.008

Aplicación móvil para el análisis de grafos bajo un enfoque de Ingeniería de Software

Mobile application for graph analysis under a Software Engineering approach

Mario Andrés Basilio López
Universidad del Istmo, campus Tehuantepec, Santo Domingo Tehuantepec Oaxaca, México
andres.basilio@outlook.es

Samuel Erasto López Díaz
Universidad del Istmo, campus Tehuantepec, Santo Domingo Tehuantepec Oaxaca, México
lopezdiaz200028@gmail.com

José Alejandro Pérez Sibaja
Universidad del Istmo, campus Tehuantepec, Santo Domingo Tehuantepec Oaxaca, México
japsmisterio@gmail.com

Guillermo Eduardo Reyes Rodríguez
Universidad del Istmo, campus Tehuantepec, Santo Domingo Tehuantepec Oaxaca, México
eduar1299@hotmail.com

José Ricardo Salvador Nolasco
Universidad del Istmo, campus Tehuantepec, Santo Domingo Tehuantepec Oaxaca, México
jose_rivel@hotmail.com

Guadalupe Toledo Toledo
Universidad del Istmo, campus Tehuantepec, Santo Domingo Tehuantepec Oaxaca, México
gtoledo@sandunga.unistmo.edu.mx
ORCID: 0000-0002-8197-8865

J. Jesús Arellano Pimentel
Universidad del Istmo, campus Tehuantepec, Santo Domingo Tehuantepec Oaxaca, México
jjap@sandunga.unistmo.edu.mx
ORCID: 0000-0003-0609-9470

doi: https://doi.org/10.36825/RITI.10.22.008

Recibido: Julio 15, 2022
Aceptado: Octubre 06, 2022

Resumen: Los avances tecnológicos contribuyen a la mejora educativa, apoyando a los docentes y alumnos en el proceso de enseñanza-aprendizaje dentro y fuera del aula de clases para superar ciertas dificultades de accesibilidad, lugar y tiempo en el desarrollo de actividades de aprendizaje. Particularmente, en el área de las matemáticas discretas existe en los alumnos la necesidad por validar los resultados de sus ejercicios de clase sin la intervención directa del docente. Razón por la cual, en el presente artículo se tiene por objetivo describir el

proceso de creación de una aplicación móvil implementada en Android que permite resolver ejercicios de teoría de grafos a través de 7 algoritmos de optimización, su desarrollo se basó en el Modelo Orientado a Objetos bajo el ciclo de vida incremental con apoyo del Lenguaje de Modelado Unificado para la construcción de los artefactos que describen su funcionalidad. Los artefactos presentados en este artículo son el resultado del refinamiento obtenido a través de los incrementos efectuados. La aplicación resultante representa una mejora a las aplicaciones identificadas en los trabajos relacionados y se tienen amplias expectativas de su implantación y uso con fines educativos.

Palabras clave: *Matemáticas Discretas, Modelo Incremental, Teoría de Grafos.*

Abstract: Technological advances contribute to educational improvement, supporting teachers and students in the teaching-learning process inside and outside the classroom to overcome certain difficulties of accessibility, place, and time to carry out learning activities. Particularly, in discrete mathematics, there is a need for students to validate the results of their class exercises without the direct intervention of the teacher. Reason why this article aims to describe the process of creating a mobile application implemented in Android that allows solving graph theory exercises through 7 optimization algorithms, its development was based on the Object-oriented Model under an incremental life cycle with the support of the Unified Language Modeling for the construction of the artifacts that describe its functionality. The artifacts presented in this article are the result of the refinement obtained through the increments made. The resulting application represents an improvement to the applications identified in the related works and there are wide expectations of its implementation and use for educational purposes.

Keywords: *Discrete Mathematics, Incremental Model, Graph Theory.*

1. Introducción

La incorporación de tecnología educativa continúa ampliando los procesos de enseñanza-aprendizaje, facilitando las tareas del docente y del estudiante respecto a la comprensión y asimilación de contenidos de distintas materias [1]. Además, debido a que no todas las personas tienen la misma capacidad de aprendizaje en el aula, pues este es un proceso único y específico de cada individuo, algunos alumnos presentan más dificultades para asimilar el conocimiento en unas materias que en otras, siendo las del área de matemáticas las consideradas como difíciles por los universitarios [2]. Razón de peso para ver a la tecnología como proveedora de recursos que permitan superar los retos inherentes a materias consideradas difíciles.

Las matemáticas en particular tienen un carácter formativo para un ingeniero, sirven de instrumento en el desarrollo de habilidades fundamentales como son: escribir con claridad, formalizar, adquirir destrezas para enfrentar situaciones nuevas, precisión y constancia. Sin embargo, como ciencia, es vista con un enfoque pragmático y puramente técnico, lo que en ocasiones crea un rechazo entre las personas durante su asimilación, para evitar esto, se crean herramientas que faciliten el aprendizaje [2]. Una de las ramas de interés de las matemáticas en las ingenierías es la Matemática discreta, ésta, en palabras de [3] *"Estudia los objetos discretos, es decir, el estudio de la matemática limitada al conjunto de los enteros"*. Surge como una disciplina que unifica áreas tradicionales de la matemática (combinatoria, probabilidad, geometría de polígonos, aritmética, grafos) y su impartición destaca en diversas carreras ingenieriles, como son: ingeniería en computación, licenciatura en informática, matemáticas aplicadas, etc.

Esta área de estudios se ha presentado en diferentes programas académicos a nivel ingeniería tales como: [4, 5, 6, 7], considerándose una materia indispensable para los universitarios, tal como lo afirmó [4] *"...todo futuro ingeniero necesita adquirir conocimientos y competencias en esta área, ya que es una de las ramas de las Matemáticas que más ayudan a pensar"*. A partir de la revisión de los contenidos temáticos de esta asignatura, disponibles en las fuentes mencionadas, se identificó que en [4, 6, 7] destacan tópicos sobre algoritmos de optimización y sus aplicaciones, en [4] le dedican especial atención a los algoritmos Dijkstra y Floy Warshall, útiles en teoría de grafos y diseño de árboles para finalmente en [4, 6] estudiar la resolución de problemas donde se emplean caminos Eulerianos y hamiltonianos.

La naturaleza del estudio de estos temas requiere el apoyo de elementos gráficos que permitan representar la solución a un debido problema e inclusive como validador de la ejecución de algún algoritmo de optimización.

Razón por la cual, elegir a la tecnología como agente facilitador para el cálculo de los algoritmos de optimización y las representaciones gráficas de sus soluciones, tiene el potencial de favorecer el análisis y entendimiento de los casos de estudio en sí mismo, basándose en la elección apropiada del método más que en el tiempo requerido para desarrollar la validación de su solución en clase [8].

Hoy en día, uno de los recursos tecnológicos accesibles o al alcance de la gran mayoría de los estudiantes es el dispositivo móvil, conocido como *smartphone* o teléfono inteligente, el cual brinda una serie de ventajas como: i) el conocimiento está disponible en cualquier lugar y cualquier momento [9], ii) promueve a que el estudiante sea un agente activo en su aprendizaje [10]. iii) se ofrece variedad de contenido multimedia estimulando el aprendizaje, haciéndolo atractivo [11], iv) favorece la interacción social y el aprendizaje colaborativo, facilitando la comunicación de ideas y conocimientos entre docentes y alumnos, incluso a distancia [12], v) se adapta a distintas estrategias de aprendizaje [13], vi) el costo es menor en comparación a una computadora de escritorio teniendo igualmente una mayor portabilidad y funcionalidad [14].

El presente artículo se centra en una aplicación para teléfonos inteligentes debido a su popularidad y acceso por parte de usuarios; según estadísticas obtenidas por el INEGI por medio de la Encuesta Nacional sobre Disponibilidad y Uso de Tecnologías de la Información en los Hogares (ENDUTIH) 2021, el teléfono celular es la tecnología con mayor penetración en México, ya que el 93.9% cuenta con al menos un celular de los llamados *Smartphone* [15] y su popularidad es reconocida como uno de los recursos mejor utilizados en ambientes educativos con propósitos de ayudar y/o promover el aprendizaje [16].

Al momento de esta publicación, se han identificado aplicaciones informáticas de escritorio y para dispositivos móviles (véase Tabla 1) que cubren aspectos de la asignatura de Matemáticas discretas como herramientas tecnológicas para el aprendizaje. No obstante, la mayoría de ellas carece dentro de sus recursos la integración de algoritmos de optimización y sus generalidades, así como tampoco ofrecen alternativas gráficas para visualizar la solución; a su vez, omiten los fundamentos del proceso de desarrollo de software y la metodología aplicada en su construcción, impidiendo además tener acceso al código fuente de su implementación.

Tabla 1. Aplicaciones orientadas a las matemáticas discretas y sus características relevantes.

	MatDis [17]	**TuMiST** [18]	**Matemáticas Discretas App** [19]	**Graphynx, grafos y algoritmos** [20]	**Wolfram con el paquete Vilcretas** [21]	**SAG (Sistema de análisis de grafos) [Sistema propuesto]**
¿Contiene información de la materia? ¿Qué temas?	Recursión modular, Fibonacci, Euler, algoritmo de Euclides, teorema del Resto Chino, primalidad, etc.	No	Teoría de conjuntos, operaciones, propiedades y ejemplos etc.	No	Teoría de Grafos.	Conceptos y definiciones básicas, subgrafos, complementos e isomorfos, recorridos en un grafo, concepto de caminos y circuitos, conexidad, algoritmos de Prim, Kruskal, Dijkstra, etc.
¿Resuelve algoritmos de optimización?	No	Algoritmos Prim y Kruskal	No	Algoritmos Dijkstra, Prim Kruskal etc.	Algoritmos Burbuja, inserción, selección etc.	Algoritmo de Definición de grafos, Dijkstra, Prim, Kruskal,

						Euleriana Conexa, Floyd. Estrategia Divide y vencerás.
¿Es Online?	No	No	No	No	Si	No
¿Es para dispositivos móviles?	No	No	Si	Si	No	Si
¿Es de paga?	No	No	No	Si	Si	No
¿Se desarrollaron bajo alguna metodología de ingeniería de software? ¿cuál?	No Especificado	No Especificado	No Especificado	No Especificado	No Especificado	Desarrollo de Software Orientado a Objetos
¿Cuál fue su tecnología de desarrollo?	Visual C++ y Visual Basic.	Lenguaje de programación Java con plataforma no especificada.	Android con el lenguaje de programación Java y XML.	Android con el lenguaje de programación Java y con XML.	No Especificado	Android con el lenguaje de programación Java y XML.
¿Es de código libre?	No Especificado	No Especificado	No Especificado	No Especificado	No Especificado	Si
¿Realiza representaciones gráficas?	No	Si	No	No Especificado	No Especificado	Si

La Tabla 1 fue construida a partir de las variables en común que estos primeros esfuerzos han presentado en sus descripciones, al revisar el contenido teórico que pudiesen proveer, se identificó que [18] no lo contempla, ya que se comporta como un asistente interactivo para uso docente. Por su parte, [20] es una aplicación de paga para móvil, puede crear grafos (simples, ponderados, dirigidos y/o multigrafos) y ejecutar algoritmos paso a paso. Sin embargo, carece de material de lectura que complemente a los ejercicios prácticos que pueden desarrollarse en él, por lo que se considera que el usuario debe comenzar con cierto nivel de conocimiento en el área de teoría de grafos.

Respecto a los algoritmos de optimización que dan cobertura, se identificó que [17] es una aplicación de escritorio diseñada para los estudiantes de la universidad donde fue creado, con una interfaz por línea de comandos y sin un repositorio de acceso libre al cual disponer, ofrece opciones de menú para carga de datos y realizar análisis que no involucran algoritmos de optimización, por ende, no cuenta con una opción de graficación. Por otro lado, [19] es considerada exclusivamente un manual de estudio integrado en un móvil por lo que su contenido es únicamente con el propósito de brindar apuntes que el propio autor de la App ha recopilado sobre temas específicos sin considerar tópicos asociados a la teoría de grafos o a los algoritmos de optimización.

En [21] por su parte, se considera una extensión del software comercial Wolfram Mathematica de uso exclusivo para computadoras de escritorio, además, los usuarios requieren un conocimiento sólido en el manejo de sus comandos para darle uso apropiado, finalmente, se desconoce la tecnología de desarrollo empleada en su construcción.

Por último, no fue posible comprobar el acceso al código fuente de ninguna de las aplicaciones antes mencionadas, lo que imposibilita manipular o incrementar la funcionalidad si fuese el caso, y, pese al valor que en los planes de estudio de las universidades se ha señalado, el objetivo de graficar, solo [18] ofrece dicha funcionalidad.

A partir de la revisión de los trabajos relacionados, en el presente artículo se tiene el objetivo de describir el proceso de creación de una aplicación móvil desarrollada en Android que permita resolver ejercicios de teoría de grafos a partir de una colección de 7 algoritmos de optimización, además de proveer contenido teórico que refuerza la explicación de los algoritmos disponibles y con la alternativa de su visualización gráfica, empleando para ello fundamentos de ingeniería de software mediante un modelo de desarrollo incremental, conformado cada incremento en 5 fases (requisitos, análisis, diseño, implementación y pruebas) y utilizando la metodología de desarrollo orientado a objetos para la elaboración de los artefactos con apoyo del lenguaje de modelado unificado [22], dicha aplicación fue denominada Sistema de Análisis de Grafos (SAG), descrita brevemente en la última columna de la Tabla 1 para su contraste con los trabajos relacionados.

2. Proceso de Desarrollo del Sistema de Análisis de Grafos

Al crear un software se deben seguir los planteamientos enmarcados en ingeniería de software, definida como "*La aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software*" [23]. Por lo tanto, el enfoque sistemático se asocia al ciclo de vida (modelo incremental) que guía hacia la construcción de un software de calidad, para esto, fue crucial acompañarlo de un modelo de desarrollo (modelo orientado a objetos) que brindará los conceptos y artefactos involucrados en su documentación y desarrollo respectivamente, además de una notación apropiada (UML- Lenguaje de Modelado Unificado) para representar sus requisitos a través de modelos visuales con el fin de integrar la documentación que explique su definición.

2.1. Recolección de requisitos

Después de un análisis acerca de las funciones que debería realizar el SAG, los cuales se clasifican en requerimientos funcionales, que son los servicios que el sistema debe tener y los no funcionales, los cuales son restricciones sobre los requerimientos funcionales, estos se describen en la Tabla 2.

Tabla 2. Requisitos Funcionales y No Funcionales en el SAG.

Requisitos funcionales	
Algoritmos involucrados. El sistema presenta la implementación de algoritmos de análisis de grafos útiles para el área de las matemáticas discretas	**Algoritmo de Definición de Grafos.** Partiendo de una sucesión de enteros positivos o nulos se puede decidir si se trata de una gráfica o no, se basa en el Teorema de Havel-Hakimi [24].
	Algoritmo de Prim. Su propósito es encontrar un subgrafo donde la suma de los costos o pesos de sus aristas sea el de menor ponderación en comparación del grafo original [25].
	Algoritmo de Dijkstra. Utilizado para determinar un camino de costo mínimo entre dos vértices de un grafo ponderado [26].
	Algoritmo de Kruskal. Busca el mismo objetivo que el algoritmo de Prim empleando otra metodología [27].
	Algoritmo de Euleriana Conexa. Determina si un grafo es conexo (si existe un camino entre dos vértices distintos) y/o euleriano (si existe un camino que contenga cada una de las aristas del grafo) [28].
	Algoritmo de Floyd. Determina el camino más corto entre todos los pares de nodos o vértices de un grafo [27].
	Algoritmo Divide y Vencerás. Separa un problema en subproblemas resolviéndolos recursivamente y, por último, combina las soluciones de los subproblemas para resolver el original [28].
Validación de datos. Verifica si los datos ingresados en	**Número de vértices del grafo.** Debe ser numérico no negativo, para así saber el número de vértices del grafo y crear su matriz de adyacencia.
	Peso de la Arista. Deberán ser siempre positivos o iguales a 0.
	Validación de los vértices origen y destino en algoritmo Dijkstra. Los vértices de

cada campo son correctos.	origen y destino deben definirse en el intervalo de **0** a **n-1**, donde **n** es el número de vértices.
Consultar información. El usuario podrá encontrar conceptos básicos y términos relacionados a la teoría de grafos.	
Menú de algoritmos. Menú principal donde se podrá elegir alguno de los algoritmos disponibles en la aplicación.	
Comando de análisis. Permite al usuario ejecutar el algoritmo que haya seleccionado del menú después de haber guardado la información requerida para el análisis.	
Visualizar actividad. Deberá cambiar dependiendo de la elección que haya hecho el usuario en el menú de algoritmos.	
Mostrar datos. El sistema mostrará los resultados por cada algoritmo usado	
Graficación. Opción a visualizar los resultados obtenidos gráficamente.	
Exportar Resultados. Traslada los resultados del análisis a un archivo de texto (.txt) en el dispositivo móvil del usuario.	
Requisitos No Funcionales	
Interfaz del sistema. Debe ser fácil de manejar, de aspecto minimalista y con colores agradables a la vista del usuario.	
Ayuda del uso del software. El sistema hará uso de conexión a internet para redireccionar al canal de *YouTube* dedicado al uso del software.	
Seguridad de Intervalo. El intervalo de los datos siempre será restringido con respecto al número n de vértices indicado por el usuario, entonces el intervalo de vértices siempre será de 0 a n-1.	
Activar botón de análisis. Restricción importante, se usa para evitar que el usuario comience la ejecución del análisis correspondiente antes de completar el ingreso de todos los datos necesarios.	

2.2. Ciclo de vida del Software

Teniendo en cuenta los requerimientos que debe contener la aplicación móvil, se procedió a elegir un ciclo de vida para transformarlos a una realidad concreta, un ciclo de vida es: "*Un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso*" [29]. Entonces, con apoyo de un marco metodológico se pueden conseguir ciertas ventajas en el desarrollo, entre ellas: i) detectar tempranamente defectos, ii) control de tiempos y costes de desarrollo, iii) documentación formal y estandarizada en paralelo al proceso de desarrollo, facilitan comunicación con el equipo de desarrollo y estos con usuarios, etc. [30].

Así, el ciclo de vida seleccionado fue el modelo incremental, debido a que se basa en desarrollar una implementación inicial, exponer ésta al comentario del usuario, y luego desarrollarla en sus diversas versiones hasta producir un sistema adecuado. Los primeros incrementos son versiones sin terminar del producto final, pero que aportan valor funcional al usuario y también le brindan un acercamiento temprano al sistema en la espera del término de todos sus incrementos [30].

Las actividades de especificación, desarrollo y validación están entrelazadas, con rápida retroalimentación a través de las actividades. Se avanza en una serie de pasos hacia una solución y se retrocede cuando se detectan errores. Esto significa que se puede evaluar el desarrollo del sistema en una etapa temprana, para contrastar si se entrega lo que se requiere, resultando más barato y fácil al realizar cambios en el software conforme éste se diseña [31]. En la Figura 1, se puede observar que al inicio del desarrollo se comenzó por definir la interfaz del sistema, al igual que la codificación del primer algoritmo de definición de grafos correspondiente al teorema de Havel-Hakimi [24] que incluye la entrada de datos, y por último, la salida de información como resultado de efectuar el algoritmo, obteniendo así el primer incremento totalmente funcional.

Gracias al incremento 1, se entendió la manera de cómo pedir todos los datos, entonces se logró realizar la implementación completa para el resto de los algoritmos teniendo como resultado seis nuevos incrementos. Se tuvieron dos incrementos más, que corresponden a la implementación de redirección al canal de YouTube donde estarán los tutoriales de ayuda y la codificación de la pantalla de información acerca de la Teoría de Grafos y sus

algoritmos. Por último, se crearon otros dos incrementos finales que dieron lugar a la representación gráfica de los análisis de los algoritmos y a la exportación de los datos obtenidos. Con ayuda de este modelo incremental se logró un total de 11 incrementos con un tiempo promedio de una semana para cada uno de ellos, los cuales tuvieron la oportunidad de recibir retroalimentación adecuada y así poder reducir fallos en ellos.

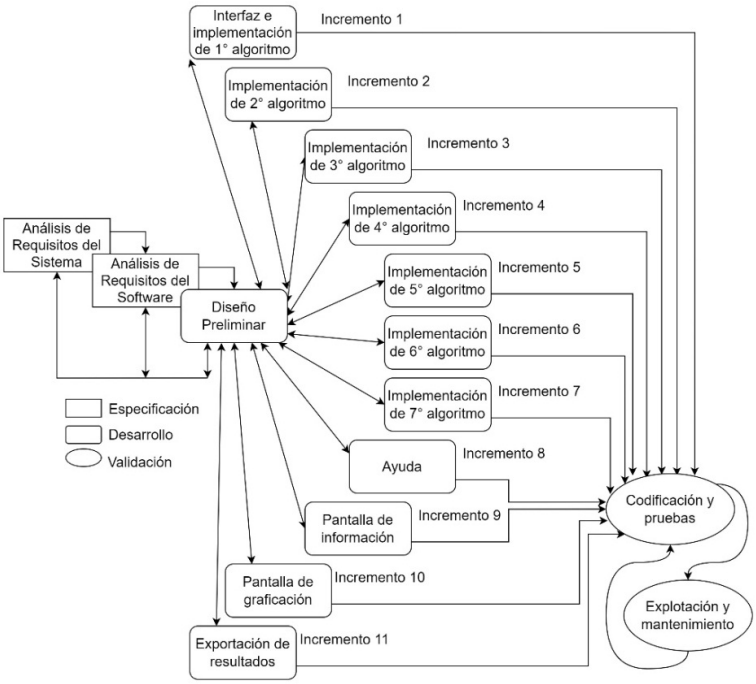

Figura 1. Incrementos propuestos en el SAG.

2.3. Etapa de Especificación

Iniciando con la fase de requerimientos, se creó el primer artefacto denominado modelo de casos de uso. Cada caso de uso debe ser guiado por el usuario de manera secuencial por eventos, que reparte la funcionalidad del sistema en transacciones para los actores/usuarios del sistema [32].

Para el SAG fueron detectados un total de 17 casos de uso y 5 actores, entre ellos, el usuario final, la base de datos, y la aplicación YouTube que interactúan entre sí para poder llevar a cabo las funcionalidades del sistema, la representación del diagrama puede verse en la Figura 2.

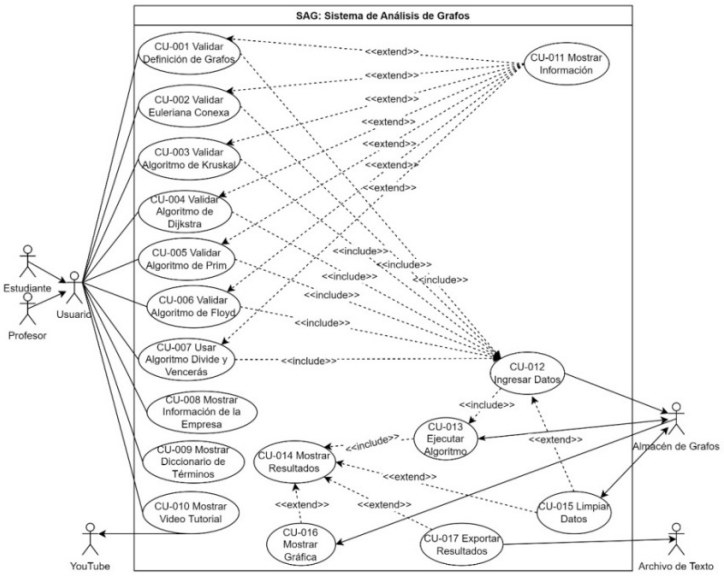

Figura 2. Modelo de casos de uso para el SAG.

Se creó un segundo artefacto denominado modelo de clases (véase Figura 3), el cual contiene aquellos objetos/términos del dominio del problema, los atributos y relaciones involucrados con la funcionalidad en la aplicación, esto para identificar elementos que pueden conformar la base de datos [22], obteniendo un total de 10 clases; éstos son tomados como punto de partida para la fase del diseño del sistema.

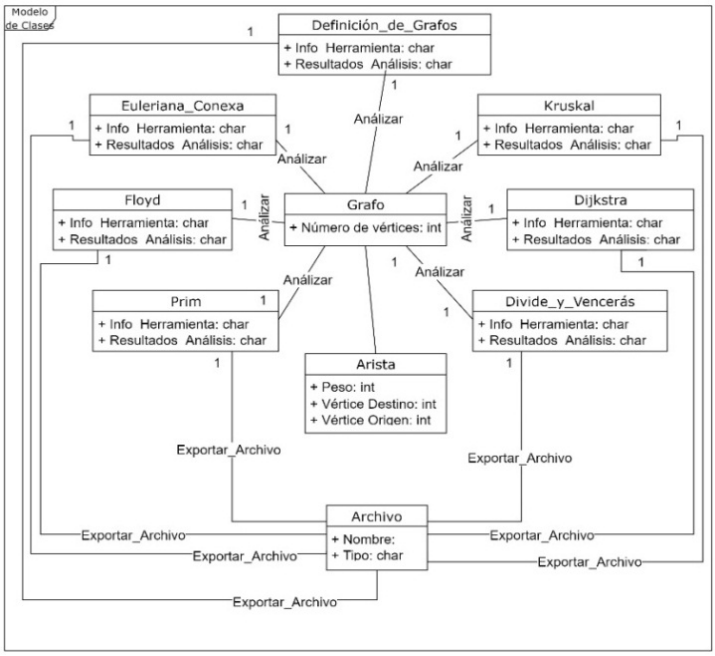

Figura 3. Modelo de clases perteneciente al SAG.

2.4. Etapa de Desarrollo

Posteriormente, comienza la fase de análisis donde se busca comprender y realizar una arquitectura de objetos que se ajuste a lo especificado en la etapa anterior, en esta etapa se realizó el artefacto denominado diagrama de secuencia, el cual describe el comportamiento dinámico del sistema enfatizado en la secuencia de los mensajes intercambiados por los objetos, siendo de 3 tipos: **entidad** son objetos que guardan información a corto y largo plazo, **borde** los que interactuarán con el usuario (interfaces) y **control** aquellos que manejan la lógica del sistema, éstos son identificados y clasificados a partir del modelo de casos de uso de la Figura 2 [22].

En el SAG se identificaron 14 objetos tipo borde, 16 objetos tipo control y 5 objetos tipo entidad, los cuales interactúan entre sí en 17 diagramas de secuencia diferentes, uno de los diagramas obtenidos se muestra en la Figura 4, el cual describe el comportamiento de SAG al seleccionar cualquiera de los algoritmos del menú principal.

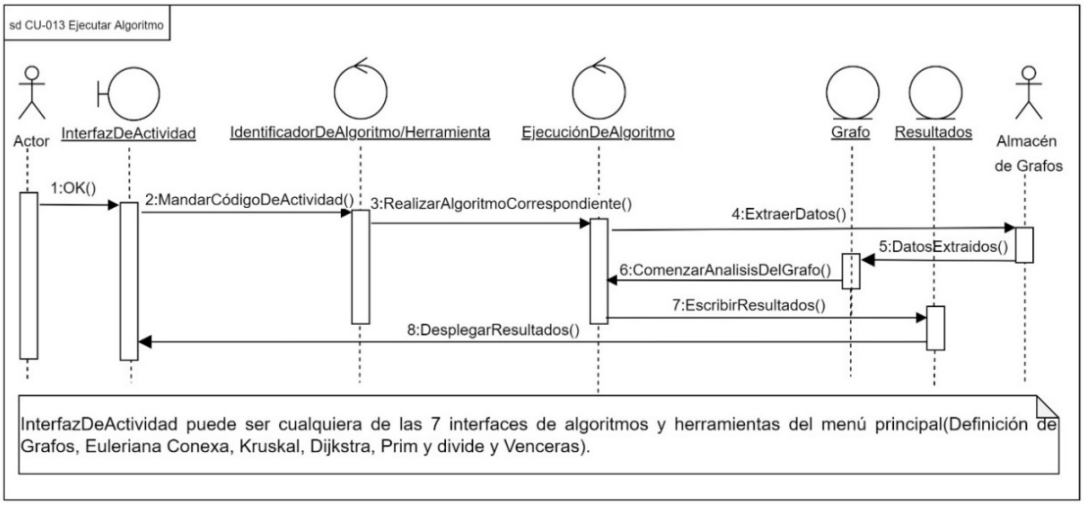

Figura 4. Diagrama de Secuencia para el Caso de Uso CU-013 “Ejecutar Algoritmo”.

Durante la fase de diseño, se busca refinar y formalizar lo hecho en la fase de análisis, tomando en cuenta el ambiente de implementación, obteniendo como resultado especificaciones detalladas de todos los objetos, con sus operaciones y atributos. Uno de los artefactos involucrados son las tarjetas CRC (Clase-Responsabilidad-Colaboración), las cuales presentan los objetos obtenidos por el modelo de casos de uso y describen el comportamiento de los diagramas de secuencia [22].

Estas tarjetas representan la guía para que el programador pueda implementar la aplicación a nivel de lenguaje de programación. El sistema SAG se constituye de 35 tarjetas CRC correspondientes a las diferentes clases de objetos que pueden conformar un diagrama de secuencia obtenido en la etapa de análisis, un ejemplo de ellas se muestra en la Tabla 3 que corresponde a la clase de tipo borde existente en la Figura 4.

El segundo artefacto en el diseño es el diagrama de componentes, éste presenta la parte física y lógica del sistema para tener una guía práctica al momento de llevar el software al mundo real [32]. En la Figura 5 se muestra el diagrama general del sistema.

Tabla 3. Tarjeta CRC correspondiente a clase "EjecuciónDeAlgoritmo" dentro del SAG.

Descripción Extrae información del almacén de datos correspondiente al grafo, para realizar la ejecución y enviar los resultados.	
Módulo: FuncionesAlgoritmos/Herramientas	
Estereotipo: Control	
Propiedades:	
Superclase:	
Subclase:	
Atributos:	
Contratos:	
Responsabilidades	**Colaboraciones**
Maneja evento "RealizarAlgoritmoCorrespondiente()"	**IdentificadorDeAlgoritmo/Herramienta**
Envía evento "ExtraerDatos()"	
Maneja evento "ComenzarAnálisisDelGrafo()"	**Grafo**
Envía evento "EscribirResultados()"	**Resultados**

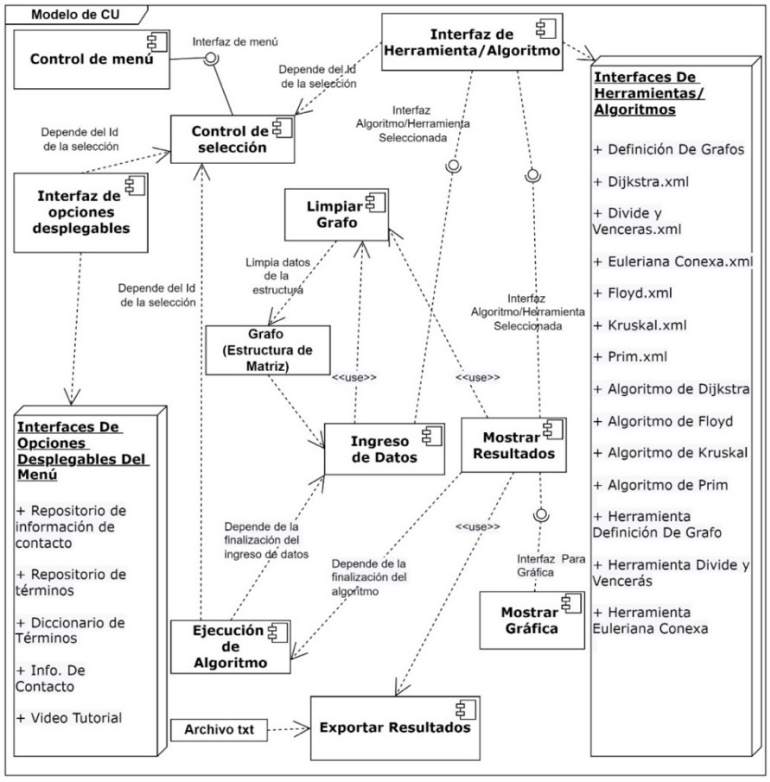

Figura 5. Diagrama de componentes del SAG.

Por último, está la fase de implementación en la cual se toma el resultado de la fase de diseño para generar el código final, esto debe de ser una traducción sencilla y directa, adaptándose al lenguaje de programación elegido y/o la base de datos [22].

Aunque es pensado que en esta fase se dedica solo a codificar, la realidad es que también se crean artefactos como el diagrama de paquetes, su objetivo es obtener una visión más clara del sistema, organizándolo en subsistemas, agrupando los elementos del análisis, diseño o construcción y detallando las relaciones de dependencia entre ellos [32]. En el sistema SAG se optó la implementación con el entorno Android Studio a través del lenguaje de programación Java, se agruparon las clases objetos mencionadas en la etapa de diseño en 13 diagramas de paquetes, mostrando el más general en la Figura 6.

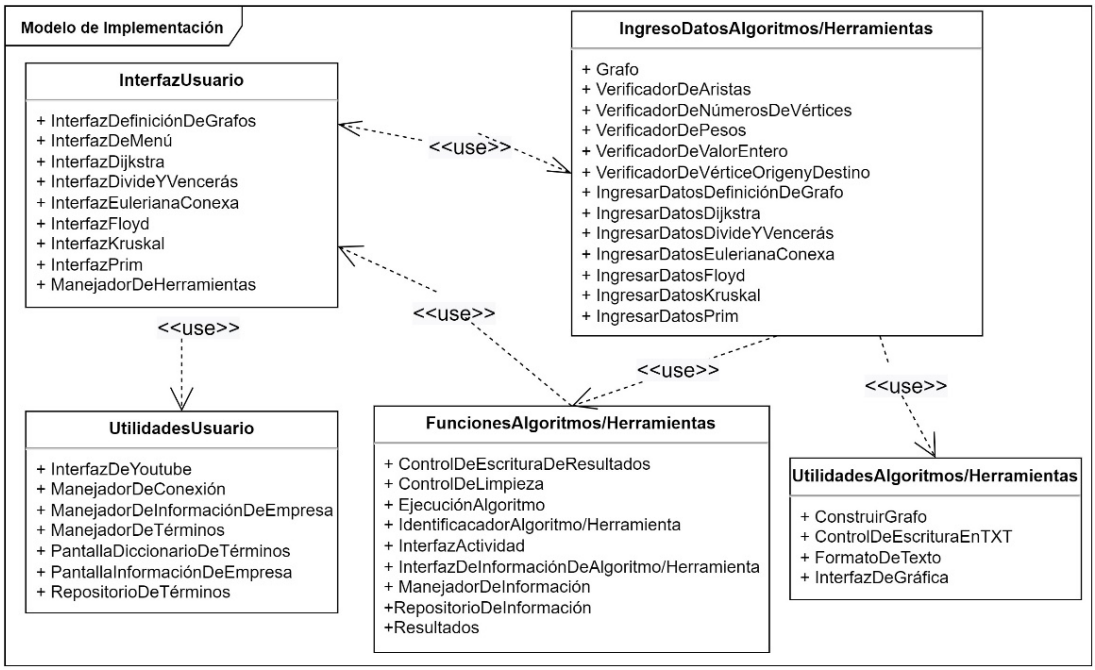

Figura 6. Modelo e Implementación del SAG.

3. Resultados

En etapas tempranas del desarrollo del software se decidió crear un prototipo de baja gama con la aplicación Balsamiq [33], a partir de los requisitos que se tenían definidos. En la Figura 7 se presenta el menú principal, además de un menú flotante que contiene el *Acerca de*, un diccionario de términos relevantes de la materia y *Tutorial* el cual redirige a YouTube que contendrá videos de cómo utilizar la aplicación, disponible en: https://youtu.be/93cuEIohovI.

Figura 7. Pantalla de inicio en el SAG con menú flotante.

Cada algoritmo disponible pide el número de vértices y el peso de cada arista, salvo los algoritmos Dijkstra y Floyd, que requieren el ingreso de un vértice origen y uno de destino para calcular los resultados, al finalizar el análisis se visualizan dos botones, uno para exportar los resultados en un archivo de texto y otro para mostrar una representación gráfica de los resultados del grafo. En la Figura 8 se puede apreciar el diseño de la pantalla para el caso del algoritmo *Dijkstra*.

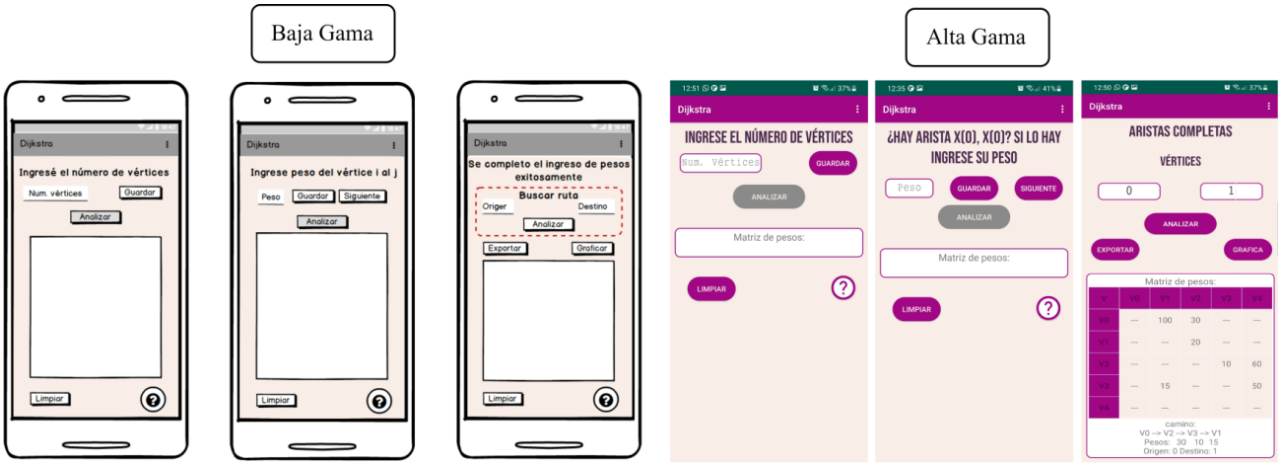

Figura 8. Interfaz del algoritmo de Dijkstra dentro del SAG.

3.1. Etapa de Validación

Para comprobar que la aplicación funciona de manera correcta se aplicaron pruebas, las cuales se encargan de la certificación final de la calidad del producto y de encontrar la mínima cantidad de errores que no afecten con el funcionamiento total del sistema para ser corregidos en futuras actualizaciones [22]. Se realizaron pruebas de unidad a cada módulo de manera individual y se corroboró que lo obtenido fuera lo esperado. Se implementó la primera prueba de unidad al algoritmo Floyd (Figura 9), éste recibe dos matrices, la de pesos y la de recorridos, se espera que el método devuelva las dos matrices anteriores, pero con los cálculos realizados.

```
public void testFloyd() {
    int vertice = 4;
    int infinito = 999999999;
    int
matFloyd[][]={{0,5,infinito,infinito},{50,0,15,5},{30,infinito,0,15},{15,infinito,5,0}};
    int matRec [][] = {{-1,2,3,4},{1,-1,3,4},{1,2,-1,4},{1,2,3,-1}};

    assertTrue("True", floyd.Floyd(matFloyd, vertice,matRec));
}
```

Figura 9. Prueba de Unidad al Método de Floyd.

Es importante señalar que SAG es capaz de soportar grafos con más de 10 vértices, no obstante, al trabajar con esa cantidad de vértices podría llevar al usuario a ingresar cada dato que se pide por arista, pues tomando como ejemplo un grafo de 10 vértices, el sistema solicitaría un total de 100 pesos.

De igual manera, el comportamiento con una gran cantidad de vértices podría causar que los resultados en la parte de la graficación no sean apreciables de forma esperada. Por lo que la recomendación que se le ofrece al usuario es trabajar con un máximo de 5 vértices para apreciarlo mejor en el visor de SAG.

3.2. Pruebas de funcionalidad

Se implementaron pruebas de funcionalidad a ejercicios encontrados en [27]. Para el grafo de la Figura 10 se determinaron las rutas de costo mínimo utilizando el algoritmo de Floyd, confirmando el resultado (Figura 11) por parte de SAG e indicando la ruta de costo mínimo en color rojo bajo la ruta de los nodos 1- 2- 5. Se

comprobó que la matriz de distancias del ejercicio (Figura 10) coincidía con la matriz de distancias mostrada en SAG (Figura 11).

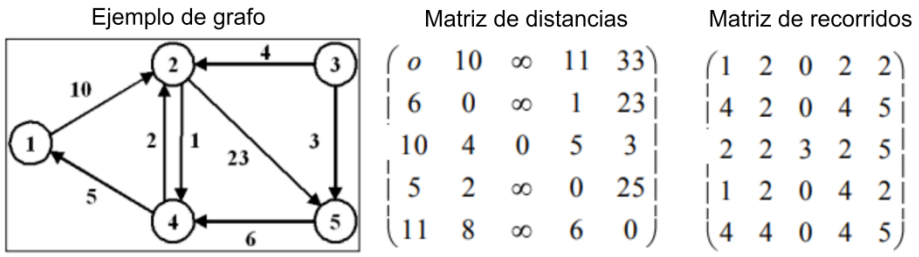

Figura 10. Grafo y matriz de distancias para obtener ruta de costo mínimo. Fuente: Tomado de [27].

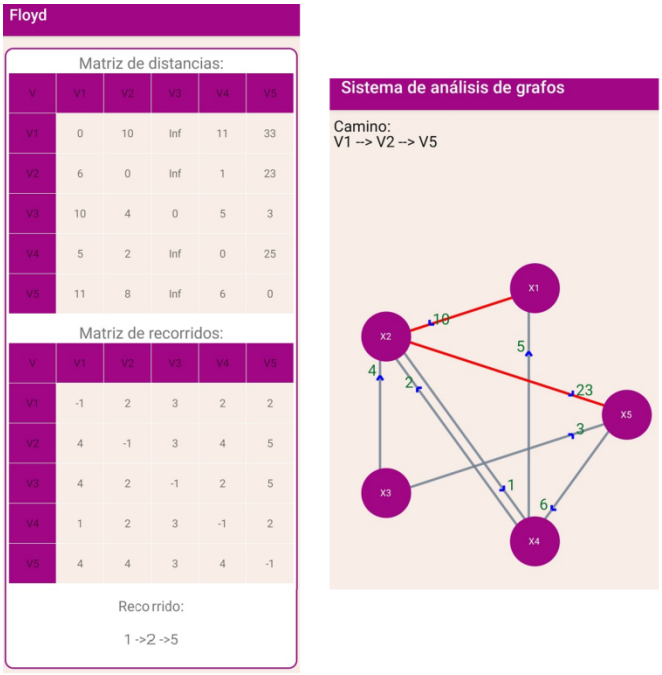

Figura 11. Grafo con ruta de coste mínimo y matriz de distancias obtenidas por SAG

A su vez, se intentó acceder a las aplicaciones descritas en la Tabla 1 y bajo las cuáles se compara a SAG para demostrar las mejoras de esta última, encontrándose lo siguiente:

- MatDis [17] vs SAG: MatDis solo maneja su interfaz por línea de comandos por lo que la interacción del usuario está restringido a la selección por opción de menú y al ingreso de los datos vía arreglos, además de que, como fue mencionado en la sección de Introducción, no incluye entre sus funciones la ejecución de algoritmos de optimización lo que impidió replicar el ejercicio de la Figura 11.
- TuMist app vs SAG [18]: no se encuentra disponible en la ruta proporcionada por sus autores: http://www.lite.etsii.urjc.es/greedex/, al ser un asistente interactivo solo de uso para el docente no se contempla funcionalidades para el alumno lo que acota su utilidad entre los usuarios, por lo que el ejercicio de replicar la actividad de la Figura 11 tampoco fue posible.
- Matemáticas Discretas App [19] vs SAG: pese estar desarrollado en Android y ser una aplicación móvil, esta solo contiene fundamentos teóricos sobre tópicos del área de matemáticas discretas, sí se encuentra disponible para su descarga sin costo en Google Play, sin embargo, su funcionalidad no brinda la resolución de problemas del tipo de la Figura 11.
- Graphynx, grafos y algoritmos [20] vs SAG: En cuanto a características según su documentación tiene funcionalidades como: ejecución de algoritmos paso a paso, exportar grafos en archivos vector, dar etiquetas a los nodos, registro de ejecución detallado para todos los algoritmos disponibles, etc. No

obstante, es una aplicación de paga por lo que no fue posible adquirirlo para realizar una comparativa respecto a su funcionalidad y, además, no cuenta con tutoriales que expliquen su uso ni bases teóricas que complementen a los algoritmos integrados en él.

- Matemática discreta a través del uso del Paquete Vilcretas [21] vs SAG: Tiene que ser instalado en la computadora. No se encuentra disponibles para su descarga en el enlace indicado por el autor http://www.escinf.una.ac.cr/discretas/index.php/archivos/category/7-packages. Es una herramienta muy completa, sin embargo, el principal obstáculo es que solo está disponible para ejecución en computadora de escritorio y depende de Wolfram para funcionar, además se presenta el inconveniente de aprender todos los comandos necesarios para su uso sin contar con una introducción guiada.

En contra parte a los inconvenientes anteriormente descritos, se puede disponer de la aplicación SAG aquí propuesta tanto para el acceso a su código fuente como para su instalación en dispositivos Android a través del siguiente enlace: https://github.com/MarioBasilio08/SAG/tree/master. Cabe mencionar que debido a su carácter de ser una aplicación de código libre también se busca permitir a otros usuarios ampliar sus funcionalidades o mejorar las ya existentes.

4. Conclusiones

Gracias a SAG, aplicar tópicos de matemáticas discretas será algo práctico de hacer, ayudará a los estudiantes de diferentes sistemas educativos de nivel superior a verificar que sus resultados sean correctos, de este modo tendrán una retroalimentación inmediata sobre su razonamiento y la ejecución de las técnicas de resolución en clase; de igual manera, la aplicación representará un recurso de utilidad a los profesores para agilizar la comprobación de los ejercicios de sus estudiantes, dando oportunidad al profesor para atender dudas durante la clase, todo esto de manera fácil y gratuita.

Tal como se presentó en la Tabla el SAG tiene características y funcionalidades comunes a algunas aplicaciones identificadas en repositorios de descarga y publicaciones científicas, por ejemplo, se corresponde, a excepción de la reproducción de los videos de ayuda, con [17, 18, 19, 20] en no requerir recursos online para su ejecución, con [17, 18, 19] en ser gratuito, y con [19, 20] en emplear como tecnología de desarrollo a Android, Java y XML.

No obstante, SAG se diferencía de todos ellos por integrar una mayor cantidad de algoritmos de optimización empleados en la solución de problemáticas en teoría de grafos pese que, a su vez, no cubre tópicos atendidos en [17] y [19], teniendo como principal contribución el seguimiento vía incrementos de la metodología orientada a objetos a través del desarrollo de artefactos que justifican y permiten replicar su implementación en tecnología móvil, o incluso ampliar otras funcionalidades más, dado que su código fuente está disponible en línea para su descarga.

El desarrollar esta aplicación bajo el enfoque de la Ingeniería de Software mediante el ciclo de vida incremental proporcionó el conocimiento para elaborar un producto de calidad siguiendo normas establecidas y llevando un control en el tiempo de implementación de cada etapa al desarrollar una aplicación móvil, así mismo se logró demostrar la secuencialidad y vinculación entre artefactos/modelos y las diferentes etapas de desarrollo, generándose la documentación pertinente para posteriores modificaciones en el software.

Como trabajos a futuro se espera incorporar una función de acercamiento (zoom) al grafo resultante, al igual que tener movilidad en los nodos de los grafos creados, aumentar el número de algoritmos y herramientas en el software, un buzón de quejas y comentarios, así como un chat-bot para que el usuario pueda aclarar dudas sobre el software y su uso. A su vez, se contempla aplicar pruebas de usabilidad a SAG con usuarios finales, tal como se ha empleado en enfoques basados en la Experiencia del Usuario como en [34], esto permitirá realizar estudios de intervención educativa tal como [35] en los que se analice la efectividad del software dentro de la asignatura para verificar si con la aplicación se obtiene mejoría en el proceso de enseñanza-aprendizaje.

5. Referencias

[1] Vital Carrillo, M. (2015). Ensayo de los principales usos de la tecnología educativa. *Vida Científica Boletín Científico de La Escuela Preparatoria No. 4*, *3* (5). https://repository.uaeh.edu.mx/revistas/index.php/prepa4/article/view/1946

[2] Del-Prado, N. (2016). *La matemática: ¿Enseñar para que la aprueben o para que la aprendan y apliquen?* http://www.cubadebate.cu/opinion/2016/08/26/la-matematica-ensenar-para-que-la-aprueben-o-para-que-la-aprendan-y-apliquen/
[3] Estopiñán Lantigua, M., Telot González, J. A. (2017). Contribución de la matemática discreta a la formación del ingeniero informático. *Atenas*, *3* (39), 18-30. http://atenas.umcc.cu/index.php/atenas/article/view/183
[4] Universidad a Distancia de Madrid. (2022). *Matemática discreta*. Universidad a Distancia de Madrid. https://www.udima.es/es/matematica-discreta.html
[5] Instituo Politécnico Nacional. (2022). *Ingeniería en Computación. Plan de Estudios*. IPN. https://sacadem.esimecu.ipn.mx/ic/plan-de-estudios
[6] Universidad Nacional Autónoma de México. (2019). *Facultad de Ingeniería / Ingeniería en Computación Mapa Curricular 2016*. UNAM. https://www.ingenieria.unam.mx/programas_academicos/licenciatura/computacion_plan2016.php
[7] Universidad de Guadalajara. (2020). *Ingeniería en Computación – UdeG Guia de Carreras*. UDG. http://guiadecarreras.udg.mx/ingenieria-en-computacion/
[8] Ballesteros-Ballesteros, V. A., Rodríguez-Cardoso, Ó. I., Lozano-Forero, S., Nisperuza-Toledo, J. L. (2020). El Aprendizaje Móvil en Educación Superior: Una Experiencia desde la Formación de Ingenieros. *Revista Científica*, *38* (2), 243–257. https://doi.org/10.14483/23448350.15214
[9] Toktarova, V. I., Blagova, A. D., Filatova, A. D., Kuzmin, N. V. (2015). Design and Implementation of Mobile Learning Tools and Resources in the Modern Educational Environment of University. *Review of European Studies*, *7* (8), 318-324. https://doi.org/10.5539/res.v7n8p318
[10] Tseng, H., Tang, Y., Morris, B. (2016). Evaluation of iTunes University courses through instructional design strategies and m-learning framework. *Educational Technology & Society*, *19* (1), 199–210. https://www.jstor.org/stable/jeductechsoci.19.1.199
[11] Šimonová, I. (2015). Mobile-assisted ESP learning in technical education. *Journal of Language and Cultural Education*, *3* (3), 1–15. https://doi.org/10.1515/jolace-2015-0016
[12] Zidoun, Y., El arroum, F. Z., Talea, M., Dehbi, R. (2016). Students' Perception About Mobile Learning in Morocco: Survey Analysis. *International Journal of Interactive Mobile Technologies (IJIM)*, *10* (4), 80-84. https://doi.org/10.3991/ijim.v10i4.5947
[13] Mejía Dávila, M. R. (2020). M-Learning: características, ventajas y desventajas, uso. *Revista Docentes 2.0*, *8* (1), 50-52. https://doi.org/10.37843/rted.v8i1.80
[14] Freire Carrera, D. E. (2017). *Estrategia metodológica apoyada por dispositivos móviles y el aprendizaje de derecho tributario en los estudiantes de la Facultad de Jurisprudencia De Uniandes* [Tesis Maestría]. Universidad Regional Autónoma de los Andes. https://dspace.uniandes.edu.ec/handle/123456789/6554
[15] INEGI. (2021). *Encuesta Nacional sobre Disponibilidad y Uso de Tecnologías de la Información en los Hogares (ENDUTIH) 2021*. https://www.inegi.org.mx/programas/dutih/2021/
[16] Lavín Zatarain, S., Zaldívar-Colado, A., Rodelo Moreno, J. A., Zaldívar Martínez, J. J. (2019). Utilización del smartphone por estudiantes del nivel superior. *Revista de Investigación En Tecnologías de La Información*, *7* (14), 89–97. https://doi.org/10.36825/RITI.07.14.008
[17] Blas, M. J., Gaspoz, C., Herrera, M. (2009, June 11). *MatDis: Aplicación de apoyo para Matemática Discreta*. 3º Congreso Nacional de Estudiantes de Ingeniería En Sistemas de Información (CNEISI). https://doi.org/10.13140/2.1.5133.3921
[18] Debdi, O., Granada, J. D., Velázquez Iturbide, J. Á. (2010). *Ayudante interactivo para los algoritmos de Prim y Kruskal*. XVI Jornadas de Enseñanza Universitaria de la Informática, Santiago de Compostela. http://hdl.handle.net/2099/11848
[19] Sabogal, S. (2018). *Matemáticas Discretas App*. https://play.google.com/store/apps/details?id=santiagosabogalco.matematicasdiscretasapp
[20] VILARIS. (2016). *Graphynx, grafos y algoritmos*. APKGT. https://apkgk.com/es/com.vilaris.graphynx
[21] Vilchez Quesada, E. (2018). Matemática discreta a través del uso del Paquete Vilcretas. *Revista Digital: Matemática, Educación e Internet*, *18* (2). https://doi.org/10.18845/rdmei.v18i2.3561
[22] Weitzenfeld, A. (2009). *Ingeniería de Software Orientada a Objetos con UML, Java e Internet*. Thomson.
[23] Pressman, R. S. (2005). *Software engineering: a practitioner's approach* (6ta Ed.). Palgrave Macmillan.
[24] Barrus, M. D., Molnar, G. (2015). Graphs with the strong Havel-Hakimi property. *Graphs and Combinatorics*, *32* (1), 1689–1697. https://doi.org/10.1007/s00373-015-1674-7

[25]Cormen, T. H., Leiserson, C. E., Rivest, R. L., Clifford, S. (2009). *Introduction to Algorithms* (3rd. Ed.). MIT Press.
[26]Rosen, K. H. (2005). *Matemática discreta y sus aplicaciones* (5th Ed.). Mcgraw-hill/Interamericana.
[27]Peñaranda Ortega, M., López Serrano, R., Quiñones Vidal, E., López García, J. J. (2006). Los «Small Worlds» y el algoritmo de Floyd: una manera de estudiar la colaboración científica. *Psicothema*, *18* (1), 78–83. https://www.redalyc.org/articulo.oa?id=72718112
[28]Khan Academy. (2022). *Algoritmos de divide y vencerás*. https://es.khanacademy.org/computing/computer-science/algorithms/merge-sort/a/divide-and-conquer-algorithms
[29]AENOR. (1999). *UNE 71044 :tecnología de la información : procesos del ciclo de vida del software. https://www.en.aenor.com/_layouts/15/r.aspx?c=N0022235*
[30]Piattini, M. G. (2003). *Análisis y diseño de aplicaciones informáticas de gestión Una perspectiva de Ingeniería del software*. Ra-ma.
[31]Sommerville, I. (2005). *Ingeniería del software* (7ma. Ed.). Pearson Educación.
[32]Rumbaugh, J., Jacobson, I., Booch, G. (2000). *El Lenguaje de Modelado Unificado. Manual de Referencia*. Addison Wesley.
[33]Balsamiq. (2022). *Balsamiq Wireframes*. https://balsamiq.com/wireframes/
[34]Tlapa García, L. A., Escalante Vega, J. E., Alonso Ramírez, L. (2021). Aplicación interactiva para el aprendizaje con evaluaciones en el área de las matemáticas. *Revista de Investigación En Tecnologías de La Información*, *9* (19), 16–31. https://doi.org/10.36825/RITI.09.19.002
[35]Martínez Noris, L., Fernández Batista, G., Sánchez Álvarez, L. Á. S. (2019). Aplicación móvil para el aprendizaje del idioma inglés en el cuarto grado. *Revista de Investigación En Tecnologías de La Información*, *7* (13), 70–76. https://www.riti.es/ojs2018/inicio/index.php/riti/article/view/156/html