



UNIVERSIDAD LAICA “ELOY ALFARO” DE MANABÍ



**FACULTAD DE CIENCIAS INFORMÁTICAS
CARRERA DE INGENIERÍA EN SISTEMAS**

**DESARROLLO DE UNA HERRAMIENTA PARA EL
DISEÑO Y EJECUCIÓN DE DIAGRAMAS DE FLUJO
ENFOCADOS A ALGORITMOS COMPUTACIONALES EN
PLATAFORMAS MÓVILES**

**TRABAJO DE TITULACIÓN MODALIDAD PROYECTO INTEGRADOR,
PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS**

AUTORES:

GUEVARA VERA CARLOS GABRIEL


PÁRRAGA CEDEÑO JORDAN JAVIER

DIRECTOR DE TEMA:

ING. JOSÉ CRISTÓBAL ARTEAGA VERA. MG.

MANTA – ECUADOR

Agosto, 2017

	UNIVERSIDAD LAICA "ELOY ALFARO" DE MANABI
	<small>Creada el 13 de noviembre de 1985 mediante Decreto Ley No.10, publicado en el Registro Oficial No. 313</small>
	FACULTAD DE CIENCIAS INFORMÁTICAS
	<small>Creada, Resolución H. Consejo Universitario del 11 de Julio del 2001</small>

Certificación

En calidad de Docente de la Facultad de Ciencias Informáticas de la Universidad Laica Eloy Alfaro de Manabí, certifico:

Haber dirigido y revisado el Trabajo de Titulación Modalidad Proyecto Integrador: “DESARROLLO DE UNA HERRAMIENTA PARA EL DISEÑO Y EJECUCIÓN DE DIAGRAMAS DE FLUJO ENFOCADOS A ALGORITMOS COMPUTACIONALES EN PLATAFORMAS MÓVILES”, proyecto que cumple con los requisitos que exige la Guía Metodológica de Titulación de la Institución y el instructivo normativo para trabajos de titulación de la carrera Ingeniería en Sistemas de la Facultad de Ciencias Informáticas y, reúne los méritos suficientes para ser sometido a la evaluación del jurado examinador que designen las autoridades.

La autoría del tema desarrollado, corresponde a los señores GUEVARA VERA CARLOS GABRIEL y PÁRRAGA CEDEÑO JORDAN JAVIER, estudiantes con estudios concluidos en la carrera Ingeniería en Sistemas, período académico 2016-2017, quienes se encuentran aptos para la defensa.

Particular que certifico para los fines consiguientes, salvo disposición de Ley en contrario.

Lo certifico:

Ing. José Cristóbal Arteaga Vera, Mg.
Docente Facultad de Ciencias Informáticas
Universidad Laica “Eloy Alfaro” de Manabí

Manta, 24 de julio de 2017.

Formando científica, tecnológica y culturalmente a los futuros profesionales en las ciencias informáticas

	UNIVERSIDAD LAICA "ELOY ALFARO" DE MANABI
	Creada el 13 de noviembre de 1985 mediante Decreto Ley No.10, publicado en el Registro Oficial No. 313
	FACULTAD DE CIENCIAS INFORMÁTICAS
	Creada, Resolución H. Consejo Universitario del 11 de Julio del 2001

TRABAJO DE TITULACIÓN MODALIDAD PROYECTO INTEGRADOR, PREVIO
A LA OBTENCIÓN DEL TÍTULO DE: INGENIERO EN SISTEMAS

“DESARROLLO DE UNA HERRAMIENTA PARA EL DISEÑO Y EJECUCIÓN DE
DIAGRAMAS DE FLUJO ENFOCADOS A ALGORITMOS COMPUTACIONALES
EN PLATAFORMAS MÓVILES”

Tribunal examinador que declara APROBADO el Grado de INGENIERO EN
SISTEMAS, de los señores:

GUEVARA VERA CARLOS GABRIEL y PÁRRAGA CEDEÑO JORDAN JAVIER

Ing. Fabricio Rivadeneira Zambrano, Mg.

Ing. Jorge Pincay Ponce, Mg

Ing. Luzmila López Reyes, Mg

Manta, 21 de agosto de 2017

Formando científica, tecnológica y culturalmente a los futuros profesionales en las ciencias informáticas

Declaración Expresa De Autoría

Nosotros, Carlos Gabriel Guevara Vera y Jordan Javier Párraga Cedeño, en calidad de autores del trabajo de titulación: “DESARROLLO DE UNA HERRAMIENTA PARA EL DISEÑO Y EJECUCIÓN DE DIAGRAMAS DE FLUJO ENFOCADOS A ALGORITMOS COMPUTACIONALES EN PLATAFORMAS MÓVILES”, autorizamos a la Universidad Laica “Eloy Alfaro” de Manabí, hacer uso completo o parcial del contenido de este trabajo de titulación del cual somos responsables, con fines estrictamente académicos o de investigación.

Los derechos que como autores nos corresponden, con excepción de la presente autorización, seguirán vigentes a nuestro favor, de conformidad con lo establecido en los artículos 5, 6, 8, 19 y demás artículos pertinentes de la Ley de Propiedad Intelectual y su Reglamento.

Asimismo, autorizamos a la Universidad Laica “Eloy Alfaro” de Manabí para que realice la digitalización y publicación de nuestro trabajo de titulación en el repositorio virtual, en conformidad a lo establecido en el Art. 144 de la Ley Orgánica de Educación Superior.

Manta, 24 de julio de 2017

Carlos Gabriel Guevara Vera
CI: 131102940-7
Telf.: 0987234013
Email: e1311029407@live.ulead.edu.ec

Jordan Javier Párraga Cedeño
CI: 131282888-0
Telf.: 0997644371
Email: e1312828880@live.ulead.edu.ec

Dedicatoria.

“El futuro mostrará los resultados y juzgará a cada uno de acuerdo a sus logros”

Nikola Tesla

A mi madre por el esfuerzo y entrega constante realizado día a día para brindarme una educación de calidad y que no me falte nada en la vida, por enseñarme a ser mejor cada día y estar siempre a mi lado. A mi padre por ser un ejemplo de liderazgo y por los consejos que me dio guiándome para ser un profesional y una persona con buenos valores y principios. A mi familia en general por su apoyo y unión.

A mis amigos por estar presente en las buenas y malas circunstancias, por los momentos inolvidables creados a lo largo de la vida, por el apoyo mutuo y ser parte de mi familia.

A mis profesores por haberme guiado en el camino del aprendizaje y ser consejeros a la hora de tomar decisiones.

Gabriel Guevara Vera

Dedicatoria.

*“Solo triunfa en el mundo quien
se levanta y busca a las circunstancias,
creándolas si no las encuentra”.*

George Bernard Shaw

A mis padres: Marcos y Janeth, hermanos: Karely, Angel y Mathew, quienes son el apoyo que necesito para seguir adelante; quizás no he estado con ellos toda mi vida, pero el solo saber que están para mí, y que esperan lo mejor de mí, me da la fuerza para librar cada batalla con la finalidad de lograr las metas propuestas.

A mis amigos, con quienes comparto momentos gratos, olvidando los problemas personales para vivir momentos felices, llenos de bromas y anécdotas que se escriben en la historia de mi vida.

Y a todas aquellas personas que siempre velaron por mi progreso, quienes me hicieron sentirme parte de su vida, de su familia, quienes compartieron momentos difíciles, brindando su apoyo desinteresado.

Todos son parte fundamental en mi vida para hoy ser quien soy.

Jordan Párraga Cedeño

Agradecimiento

*“La gratitud da sentido a
nuestro pasado, trae paz
al presente y crea
una visión para el mañana”*

Anónimo

Agradezco mi padres Carlos y Magdalena por su apoyo incondicional a lo largo de mis estudios dándome una excelente educación, por los valores inculcados, por guiarme en el transcurso de mi vida y ser un ejemplo a seguir.

A mi hermano Xavier por ser un ejemplo en lo académico y profesional motivándome a superarme constantemente.

A mis tíos, primos y abuelos por ser siempre unidos y enseñarme el valor de la familia, recibirme siempre con los brazos abiertos y apoyarme en mi crecimiento como persona y profesional.

A mis tías Piedad y Gladys por su infinita bondad y dar siempre todo de ellas para ver alegre a cada una de las personas que las rodean.

Le agradezco al Ing. José Arteaga por haber confiado en Jordan y en mí durante el desarrollo de este proyecto de titulación en calidad de director, por habernos guiado y preparado en esta etapa final de la carrera y sobre todo brindarnos nuevas oportunidades en nuestra carrera como profesionales.

A mi compañero del presente proyecto de titulación y emprendimientos profesionales Jordan Párraga por el esfuerzo y dedicación realizado en cada trabajo realizado, con la finalidad de alcanzar los objetivos y metas planteadas.

Le agradezco a cada uno mis profesores por haber sido parte de mi formación profesional en especial a: Ing. John Cevallos, Ing. Freddy Alarcon, Ing. Winther Molina y al Ing William Zamora por su apoyo, tiempo, amistad y sobre todo por compartir sus conocimientos y consejos dentro y fuera del aula de clases.

A mis amigos en cada etapa de mi vida por los momentos compartidos dentro y fuera de las aulas, por la confianza que me otorgaron y su apoyo incondicional.

A mi perro Jason por ser un compañero inseparable desde mi infancia, por cada momento compartido, los paseos, los juegos y recibirme siempre con alegría.

Gabriel Guevara Vera

Agradecimiento

*“Si un hombre no está agradecido
por lo que tiene, es probable que
no sea agradecido por lo que tendrá.”*

Frank A. Clark

A mi madre, Janeth por ser el sustento de mi familia, quien se sacrifica a diario con el fin poder apoyarme en la consecución de mis metas; enseñándome el significado del sacrificio y humildad.

A mi padre Marcos y abuela Lourdes quienes me guiaron por el buen camino, formando en mí una persona de carácter firme; quienes me inculcaron buenas enseñanzas forjando la persona que soy hoy en día.

A mi hermana Karely, quien se sacrifica a diario por su hijo, en un país diferente al que la vio nacer, siendo mi motivación para poseer las fuerzas necesarias para vencer los obstáculos que te da la vida.

A mis hermanos Angel y Mathew, quienes están iniciando en esta vida, y me motivan a ser una persona de bien, para ser un ejemplo a seguir.

Al Ing. José Arteaga, quien en calidad de tutor depositó en nosotros su confianza y fue un motivador constante para la culminación del presente proyecto de titulación.

A mi compañero, Gabriel generando en conjunto grupos académicos y profesionales; compartiendo la tare ardua del desarrollo del presente trabajo de titulación

A los docentes y personal administrativo de la FACCI, quienes fueron parte mi ciclo académico, impartiendo su conocimiento generaron en mí, actitudes y valores necesarios para mi vida profesional.

A mis amigos con quienes salgo de la rutina, despejando la mente pasando momentos amenos; en quienes puedo confiar, acudiendo a mi llamo en momentos difíciles de la vida.

A todas las personas que han sido parte de mi vida, mencionadas o no anteriormente, les doy las gracias por creer en mí, por brindarme su apoyo y dedicar su tiempo escuchando o siendo parte de mis innumerables locuras.

Mi eterna gratitud.

Jordan Párraga Cedeño

Índice de Contenido

ÍNDICE TABLAS	XIV
ÍNDICE GRÁFICO E ILUSTRACIONES.....	XVI
RESUMEN	XVIII
ABSTRACT	XIX
CONTEXTUALIZACIÓN DE LA INVESTIGACIÓN	1
INTRODUCCIÓN	1
PROBLEMATIZACIÓN	3
PLANTEAMIENTO DE PROBLEMA	3
1. <i>Ubicación y contextualización de la investigación</i>	3
2. <i>Génesis del Problema</i>	3
3. <i>Estado Actual del Problema</i>	5
4. <i>Diagrama Causa-Efecto del Problema</i>	7
OBJETIVOS	8
OBJETIVO GENERAL	8
OBJETIVOS ESPECÍFICOS	8
JUSTIFICACIÓN	9
CAPÍTULO I	11
MARCO TEÓRICO DE LA INVESTIGACIÓN	11
1.1 INTRODUCCIÓN	11
1.2 ANTECEDENTES.....	12
1.2 DEFINICIONES CONCEPTUALES.....	14
1.2.1 <i>Resolución de problemas</i>	14
1.2.2 <i>Algoritmos Computacionales</i>	15
1.2.3 <i>Software para desarrollo de algoritmos</i>	20
1.2.4 <i>Smartphone</i>	21
1.2.5 <i>Metodología Scrum</i>	25
1.3 CONCLUSIONES	26
CAPÍTULO II	29
DIAGNÓSTICO O ESTUDIO DE CAMPO	29
2.1. INTRODUCCIÓN.....	29
2.2. TIPO(S) DE INVESTIGACIÓN	29
2.2.1. <i>Investigación Exploratoria</i>	29
2.2.2. <i>Investigación de Campo</i>	30
2.3. MÉTODOS DE INVESTIGACIÓN	31
2.3.1. <i>Lógico Inductivo</i>	31
2.3.2. <i>Método Bibliográfico</i>	31
2.4. HERRAMIENTAS DE RECOLECCIÓN DE DATOS.....	31
2.4.1. <i>Observación</i>	32
2.4.2. <i>Encuesta</i>	32
2.4.3. <i>Entrevista</i>	33
2.5. FUENTES DE INFORMACIÓN DE DATOS.....	34
2.5.1. <i>Fuentes primarias</i>	34

2.5.2.	<i>Fuentes secundarias</i>	35
2.6.	INSTRUMENTAL OPERACIONAL	35
2.6.1.	<i>Estructura y características de lo(s) instrumento(s) de recolección de datos</i>	35
2.7.	ESTRATEGIA OPERACIONAL PARA LA RECOLECCIÓN Y TABULACIÓN DE DATOS.....	37
2.7.1.	<i>Plan de recolección de los datos</i>	37
2.7.2.	<i>Plan de tabulación de los datos</i>	38
2.7.3.	<i>Plan de análisis e interpretación de los datos</i>	40
2.8.	PLAN DE MUESTREO.....	41
2.8.1.	<i>Segmentación</i>	41
2.8.2.	<i>Técnica de Muestreo</i>	41
2.8.3.	<i>Tamaño de la muestra</i>	42
2.9.	PRESENTACIÓN Y ANÁLISIS DE LOS RESULTADOS	44
2.9.1.	<i>Análisis de los resultados</i>	44
CAPÍTULO III	48
DISEÑO DE LA PROPUESTA		48
3.1.	INTRODUCCIÓN	48
3.2.	DESCRIPCIÓN DE LA PROPUESTA	48
3.2.1.	<i>Especificaciones Técnica</i>	51
3.2.2.	<i>Objetivos</i>	51
3.2.3.	<i>Definición de Recurso</i>	52
3.3.	ETAPAS DE LA PROPUESTA	55
3.3.1.	<i>Metodología</i>	55
3.3.2.	<i>Personas y Roles</i>	59
3.3.3.	<i>Fases</i>	59
3.3.4.	<i>Requisitos</i>	61
3.3.5.	<i>Artefactos</i>	80
3.3.6.	<i>Diseño e implementación</i>	97
3.3.7.	<i>Entregables</i>	99
3.3.8.	<i>Codificación y Testing</i>	114
3.3.9.	<i>Producto Implementado</i>	135
CAPÍTULO IV		137
EVALUACIÓN DE RESULTADOS		137
4.1	INTRODUCCIÓN	137
4.2	SEGUIMIENTO Y MONITOREO DE RESULTADOS	137
CONCLUSIONES		145
RECOMENDACIONES		147
BIBLIOGRAFÍA		148
ANEXOS		153

Índice Tablas

<i>Tabla 1 Plan de recolección de Datos</i>	<i>38</i>
<i>Tabla 2 Plan de Análisis e Interpretación de los datos</i>	<i>41</i>
<i>Tabla 3 Muestra de los estudiantes de la FACCI.....</i>	<i>42</i>
<i>Tabla 4 Muestra de los profesores de la FACCI.....</i>	<i>43</i>
<i>Tabla 5 Recursos Humanos.....</i>	<i>52</i>
<i>Tabla 6 Recursos Tecnológicos.....</i>	<i>53</i>
<i>Tabla 7 Recursos Económicos</i>	<i>54</i>
<i>Tabla 8 Personas y Roles.....</i>	<i>59</i>
<i>Tabla 9 Requisitos Funcionales</i>	<i>65</i>
<i>Tabla 10 Requisitos no funcionales.....</i>	<i>66</i>
<i>Tabla 11 Historias de Usuario Diagramación</i>	<i>72</i>
<i>Tabla 12 Historias de Usuario Pseudocódigo.....</i>	<i>74</i>
<i>Tabla 13 Historias de Usuario Conversión Diagramas – Pseudocódigo</i>	<i>75</i>
<i>Tabla 14 Historias de Usuario Conversión Pseudocódigo – Diagrama.....</i>	<i>76</i>
<i>Tabla 15 Historias de Usuario Ejecución de Algoritmo.....</i>	<i>79</i>
<i>Tabla 16 Pilas del Producto.....</i>	<i>80</i>
<i>Tabla 17 Pilas del Sprint.....</i>	<i>84</i>
<i>Tabla 18 Sprint 1</i>	<i>88</i>
<i>Tabla 19 Sprint 2</i>	<i>90</i>
<i>Tabla 20 Sprint 3</i>	<i>92</i>
<i>Tabla 21 Sprint 4</i>	<i>94</i>
<i>Tabla 22 Sprint 5</i>	<i>96</i>
<i>Tabla 23 Clases Utilizadas en el Desarrollo de los Aplicativos y su Respectiva Funcionalidad.....</i>	<i>120</i>

<i>Tabla 24 Lista de Pruebas Realizadas a los Aplicativos Móviles.....</i>	<i>134</i>
<i>Tabla 25 Características iPad utilizado para las pruebas</i>	<i>138</i>
<i>Tabla 26 Características Moto Z Play utilizado para las pruebas.....</i>	<i>140</i>
<i>Tabla 27 Tabulación de Pregunta 1 - Encuesta estudiantes FACCI</i>	<i>157</i>
<i>Tabla 28 Tabulación de Pregunta 2 - Encuesta realizada a estudiantes FACCI</i>	<i>158</i>
<i>Tabla 29 Tabulación de Pregunta 3 - Encuesta realizada a estudiantes FACCI</i>	<i>159</i>
<i>Tabla 30 Tabulación de Pregunta 4 - Encuesta realizada a estudiantes FACCI</i>	<i>160</i>
<i>Tabla 31 Tabulación de Pregunta 5 - Encuesta realizada a estudiantes FACCI</i>	<i>161</i>
<i>Tabla 32 Tabulación de Pregunta 6 - Encuesta realizada a estudiantes FACCI</i>	<i>161</i>
<i>Tabla 33 Tabulación de Pregunta 7 - Encuesta realizada a estudiantes FACCI</i>	<i>162</i>
<i>Tabla 34 Tabulación de Pregunta 8 - Encuesta realizada a estudiantes FACCI</i>	<i>163</i>
<i>Tabla 35 Tabulación de Pregunta 9 - Encuesta realizada a estudiantes FACCI</i>	<i>164</i>
<i>Tabla 36 Tabulación de Pregunta 10 - Encuesta realizada a estudiantes FACCI</i>	<i>165</i>
<i>Tabla 37 Tabulación de Pregunta 11 - Encuesta realizada a estudiantes FACCI</i>	<i>167</i>

Índice Gráfico e Ilustraciones

<i>Ilustración 1: Diagrama de flujo de datos realizado en la herramienta MTPCASE</i>	12
<i>Ilustración 2: Interfaz de la etapa Probar correspondiente al software desarrollado</i>	13
<i>Ilustración 3 Interfaz del programa PSeInt</i>	14
<i>Ilustración 4 Sprint del proyecto (Iconos Graphic Resources, S.L)</i>	56
<i>Ilustración 5 Sprints del Proyecto</i>	58
<i>Ilustración 6 Fases del Scrum</i>	60
<i>Ilustración 7 Diagrama Lógico de PseudoDF</i>	97
<i>Ilustración 8 Diagrama Físico de PseudoDF</i>	98
<i>Ilustración 9 Diagrama de Caso de Usos Modelo General PseudoDF</i>	99
<i>Ilustración 10 Diagrama de Caso de Usos Diseño de Algoritmos</i>	100
<i>Ilustración 11 Diagrama de Caso de Usos Conversión de Algoritmos</i>	101
<i>Ilustración 12 Diagrama de Caso de Usos Conversión de Algoritmos Ejecución de Algoritmos</i>	102
<i>Ilustración 13 Diagramación de Algoritmos-Espacio de Trabajo iOS</i>	103
<i>Ilustración 14 Diagramación de Algoritmos-Espacio de Trabajo Android</i>	104
<i>Ilustración 15 Diagramación de Algoritmos-Paleta de Componentes iOS</i>	105
<i>Ilustración 16 Diagramación de Algoritmos-Paleta de Componentes Android</i>	106
<i>Ilustración 17 Diagramación de Algoritmos-Edición de Sentencias iOS</i>	108
<i>Ilustración 18 Editor de Pseudocódigo-Espacio de Trabajo iOS</i>	108
<i>Ilustración 19 Ejecución de Algoritmos-Ejecución en Vista de Diagramación iOS</i>	109
<i>Ilustración 20 Ejecución de Algoritmos-Ejecución en Vista de Diagramación Android</i>	110
<i>Ilustración 21 Ejecución de Algoritmos-Ejecución en Vista de Pseudocódigo iOS</i>	111
<i>Ilustración 22 Ejecución de Algoritmos-Alerta de Errores iOS</i>	112
<i>Ilustración 23 Ejecución de Algoritmos-Marcador de Error iOS</i>	113

<i>Ilustración 24 Características iPad utilizado para las pruebas</i>	<i>139</i>
<i>Ilustración 25 Características Moto Z Play utilizado para las pruebas</i>	<i>141</i>
<i>Ilustración 26 Gráfica de Pregunta 1 - Encuesta realizada de estudiantes de la FACCI</i>	<i>157</i>
<i>Ilustración 27 Gráfica de Pregunta 2 - Encuesta realizada a estudiantes de la FACCI</i>	<i>158</i>
<i>Ilustración 28 Gráfica de Pregunta 3 - Encuesta realizada a estudiantes de la FACCI</i>	<i>159</i>
<i>Ilustración 29 Gráfica de Pregunta 4 - Encuesta realizada a estudiantes de la FACCI</i>	<i>160</i>
<i>Ilustración 30 Gráfica de Pregunta 5 - Encuesta realizada a estudiantes de la FACCI</i>	<i>161</i>
<i>Ilustración 31 Gráfica de Pregunta 6 - Encuesta realizada a estudiantes de la FACCI</i>	<i>162</i>
<i>Ilustración 32 Gráfica de Pregunta 7 - Encuesta realizada a estudiantes de la FACCI</i>	<i>163</i>
<i>Ilustración 33 Gráfica de Pregunta 8 - Encuesta realizada a estudiantes de la FACCI</i>	<i>164</i>
<i>Ilustración 34 Gráfica de Pregunta 9 - Encuesta realizada a estudiantes de la FACCI</i>	<i>165</i>
<i>Ilustración 35 Gráfica de Pregunta 10 - Encuesta realizada a estudiantes de la FACCI</i>	<i>166</i>
<i>Ilustración 36 Gráfica de Pregunta 11 - Encuesta realizada a estudiantes de la FACCI</i>	<i>167</i>



Resumen

Los algoritmos computacionales, son una secuencia de pasos que describen un proceso lógico; existen diversas formas de representarlos, por medio lenguaje humano separando y diferenciando las diversas actividades, a través de una descripción informal conocida como pseudocódigo (o lenguaje falso), representación gráfica como son los diagramas de flujo y mediante lenguajes de programación. Este proyecto está enfocado a dos representaciones, los diagramas de flujo y los pseudocódigos que son diseñados en la herramienta permitiendo la correspondiente transformación entre ambas técnicas, además de permitir la ejecución de la lógica en cualquiera de las dos formas.

Conociendo la importancia para la lógica de programación los algoritmos computacionales, es necesario el uso de herramientas informáticas que ayuden al diseño y ejecución de los algoritmos, sin embargo, con normalidad vemos que aquellos instrumentos de apoyo son desarrollados en plataformas de escritorio, teniendo así, una escasez de dichas herramientas en un ambiente más flexible, como lo son las plataformas móviles.

Es por esta razón que se planteó el desarrollo del presente proyecto que permita diseñar y ejecutar algoritmos computacionales en las plataformas móviles más conocidas, como lo son iOS y Android; generando así, una herramienta flexible para las personas que requieran desarrollar la lógica de programación. Además, con la ayuda de esta aplicación, se reduce el trabajo que conlleva la comprobación de la lógica plasmada en los algoritmos desarrollados, permitiendo su ejecución y respectiva verificación.

PALABRAS CLAVE: Algoritmos Computacionales / Diagrama de Flujo / Pseudocódigo / Plataforma Móvil / Lógica de Programación



Abstract

Computational algorithms are a sequence of steps that describe a logical process; There are different ways of representing them, through human language, separating and differentiating the various activities, through an informal description such as pseudocode (or false language), graphical representation such as flow diagrams and programming languages. This project is focused on two representations, the flow diagrams and the pseudocodes that are designed in the tool allow the adequate transformation in both techniques, besides allowing the execution of the logic in either of the two forms.

Knowing the importance of the programming logic of computational algorithms, it is necessary to use computer tools that help the design and execution of the algorithms, however, we normally see that the supporting instruments are developed on desktop platforms, thus having a shortage of such tools in a more flexible environment, such as mobile platforms.

It is for this reason that the development of the present project was conceived that allows to design and to execute computational algorithms in the well-known mobile platforms, as they are iOS and Android; Generating thus, a flexible tool for the people who need to develop the logic of programming. In addition, with the help of this application, reduce the work that involves the verification of the logic embodied in the developed algorithms, allowing its execution and respective verification.

KEYWORDS: Algoritmos Computacionales / Diagrama de Flujo / Pseudocódigo / Plataforma Móvil / Lógica de Programación



Contextualización de la Investigación

Introducción

Un algoritmo puede ser definido como la secuencia ordenada de pasos, sin ambigüedades, que conducen a la resolución de un problema dado y expresado en lenguaje natural, por ejemplo, el castellano (Universidad Nacional Experimental del Táchira, s.f.).

El presente trabajo de titulación tiene como finalidad desarrollar una herramienta orientada a plataformas móviles que sirva como apoyo para los estudiantes en el diseño y verificación de los algoritmos computacionales (DEACON); estructurado por IV capítulos. En cada capítulo se demuestra de manera lógica el desarrollo del presente proyecto.

En el Marco Teórico de la Investigación se encuentra toda la fundamentación teórica y bibliográfica de investigaciones similares al tema tratado, con la finalidad de facilitar la comprensión al lector referentes a algoritmos computacionales, el desarrollo en dispositivos móviles y la importancia de los dispositivos móviles en la educación.

El Diagnóstico o Estudio de Campo define los método de investigación, tipos de investigación y herramientas de recolección de datos que se aplicaron en el presente trabajo de titulación, especificando las fuentes donde se obtuvieron los datos. Por otro lado, se especifica la estrategia de recolección de datos, con su respectiva tabulación y análisis de la información obtenida.

El Diseño de la Propuesta especifica los recursos humanos, tecnológicos y económicos utilizados en el desarrollo de la presente propuesta. Además, detalla la aplicación de la metodología SCRUM, definiendo la pila del producto, número de iteraciones, reuniones, diseños, codificación y testeo.



La Evaluación de Resultados detalla el seguimiento y monitoreo del funcionamiento del aplicativo desarrollando, permitiendo verificar el entregable resultante de cada iteración en función de los requerimientos.

Adicional, se incorporan conclusiones, recomendaciones, bibliográficas y anexos; siendo de apoyo para la comprensión del presente trabajo de titulación.



Problematización

Planteamiento de problema

1. Ubicación y contextualización de la investigación

Los algoritmos computacionales son parte fundamental en el desarrollo de la lógica para la construcción de software. En la actualidad se evidencia dificultad en la resolución de problemas propuestos, para el diseño de algoritmos computacionales por parte de los estudiantes que cursan carreras de ingeniería.

Los estudiantes encuentran un inconveniente a la hora de realizar el DEACON, teniendo dificultad en encontrar herramientas móviles que sirvan de apoyo a la práctica de los algoritmos computacionales. Esto ocasiona dificultad por parte de los estudiantes al realizar trabajos autónomos para ayudar a la comprensión de estos algoritmos.

2. Génesis del Problema

¿Cómo aparece el problema que se pretende solucionar?

El problema aparece a raíz de la dificultad en la elaboración de algoritmos computacionales por los estudiantes de la carrera de Ingeniería en Sistemas.

Siendo base fundamental para el desarrollo de la lógica de programación, los algoritmos computacionales tienen suma importancia en el aprendizaje de los estudiantes, los cuales, deben contar con los conocimientos para materializar la solución de un problema en un algoritmo computacional; es entonces cuando se encuentran con la dificultad de la elaboración del mismo.



¿Qué elementos o circunstancias lo originan?

Para diseñar un algoritmo computacional se debe comenzar por identificar las tareas más importantes para resolver el problema y disponerlas en el orden en el que han de ser ejecutadas (Universidad Nacional Experimental del Táchira, s.f.).

Los estudiantes presentan una debilidad al momento de ordenar estas tareas, ya que deben contar con un alto nivel de comprensión y lógica para poder obtener una solución óptima.

Para verificar la validez de un algoritmo computacional, este debe ser sometido a una prueba de escritorio, generando un proceso tedioso y al ser realizada de forma manual, tiene un alto porcentaje de ser incierta, es por ello, que se recurren a software desarrollados para la DEACON, por lo general estas herramientas son orientadas a tecnologías pocos flexibles en cuanto a la movilidad y a lo económico. Los estudiantes por esto, tienen dificultades o pierden el interés en la elaboración de los algoritmos computacionales, generando un bajo rendimiento académico, o en algunos casos la deserción de la carrera.

¿Quién o Que lo origina? ¿Cuándo y dónde se origina?

Dentro de la carrera de Ingeniería en Sistema es común asumir que los estudiantes que ingresan tienen un vasto conocimiento acerca de la algorítmica, lo que conlleva a una falta de realización de un refuerzo acerca del tema y la aplicación de nuevas metodologías y herramientas.

¿Cuáles son las causa y efectos que produce el problema?

El diseño de algoritmos computaciones, es una tarea compleja de entender, requieren de un buen análisis a un problema dado. Por lo general, en la comunidad académica utilizan un esquema de enseñanza-aprendizaje basado en imponer al estudiante la resolución de un sinnúmero de ejercicios planteados con el fin de generar un buen análisis. En este tipo de esquema se deja a un lado la importancia



de una buena lógica y el comprender la utilización de los diferentes componentes que poseen las diferentes metodologías o técnicas existente para DEACON.

Existen software como herramientas que facilitan el DEACON, mejorando su comprensión y agilizando su proceso de elaboración; sin embargo, estos software por lo general se encuentran desarrolladas en un ambiente de escritorio, dejando a un lado tecnologías modernas, como son los Smartphones, debido a su alta portabilidad y valor (más asequible económicamente) son en la actualidad el 54,25% de los dispositivos más utilizados en el mundo a comparación de las computadoras con un 45.75% ¹.

¿A qué o quienes afecta el problema, en que magnitud y con qué frecuencia?

Los estudiantes de la carrera de Ingeniería en Sistemas y afines a la misma, son los principales afectados de un mal DEACON lo que desencadena un bajo rendimiento en lo académico, obteniendo bajas calificaciones, incluso perdidas de materias.

3. Estado Actual del Problema

En la actualidad existen aplicaciones para dispositivos móviles que permiten diseñar algoritmos computacionales, sin embargo, aún siguen siendo muy escasas este tipo de aplicaciones. Además, el software existente suele ser de pago y tener varias limitaciones, como el hecho de no ser posible ejecutar la lógica del algoritmo diseñado para comprobar su correcto funcionamiento. Por otro lado, las personas que requieren este tipo de herramientas no cuentan con la suficiente practica o aún tienen que desarrollar la lógica requerida; sería conveniente que este tipo de software permitiera introducir un lenguaje más informal que facilite la comprensión

¹ Dato extraído del servicio de análisis web StatCounter GlobalStats: <http://gs.statcounter.com/platform-market-share/desktop-mobile/worldwide/#monthly-201605-201705>

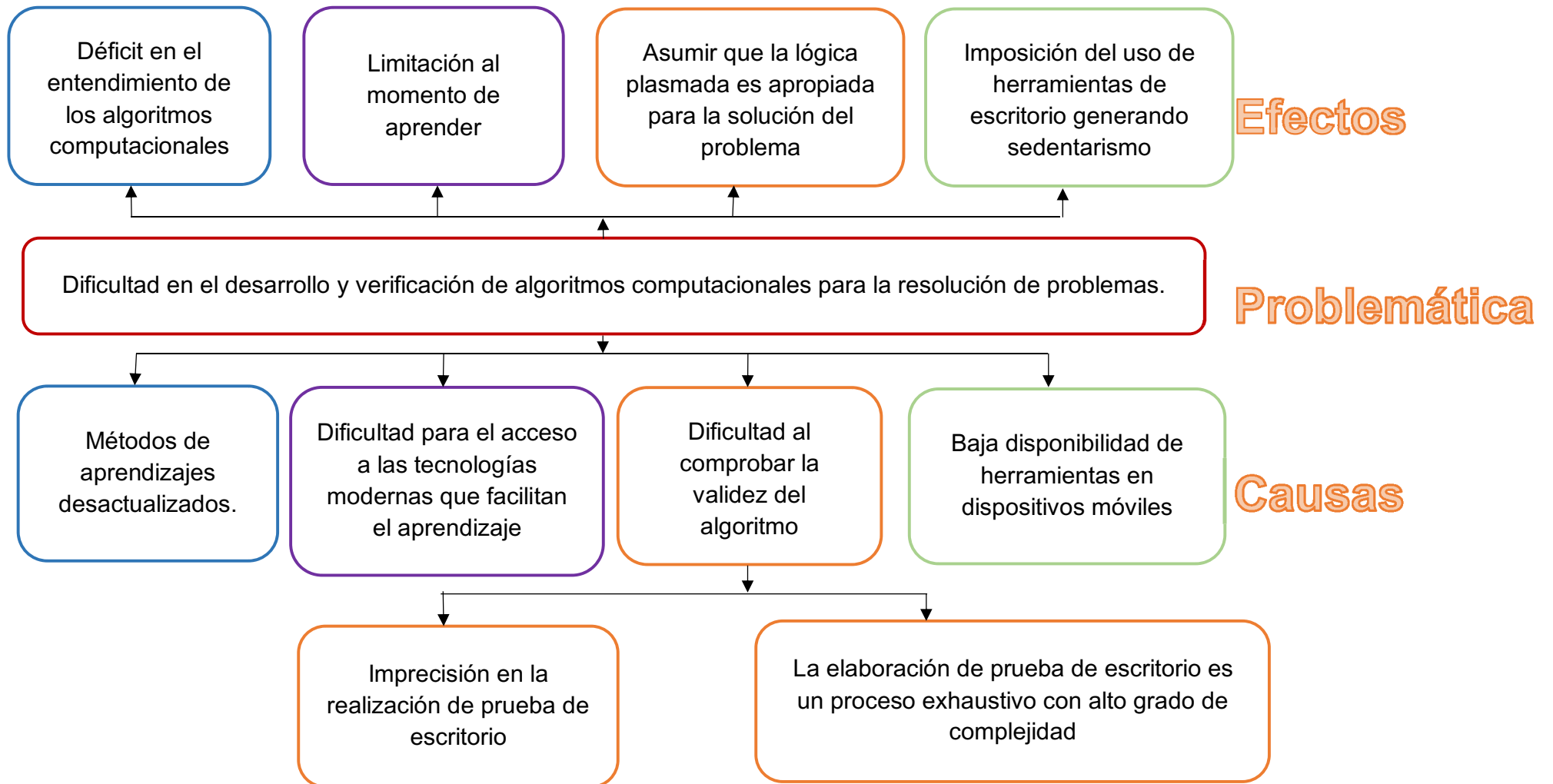


humana (como es el pseudocódigo) para la representación del algoritmo y posteriormente la conversión a un diagrama de flujo.

Puntualizando los siguientes inconvenientes:

- Escases de herramientas de apoyo para el desarrollo de la lógica de programación.
- Las aplicaciones móviles que facilitan el diseño de algoritmos computacionales, no cuentan con la funcionalidad de verificar la validez del algoritmo, realizando su respectiva ejecución.
- Como guía al estudiante, son escasas las herramientas que permiten realizar la conversión de un Diagrama de Flujo a Pseudocódigo o viceversa, permitiendo reconocer los componentes y su funcionalidad de cada una de las técnicas de representación.
- Las herramientas existentes, son desarrolladas para computadores, prescindiendo de tecnologías móviles como son los smartphome, para el apoyo académico en materias relacionadas a los algoritmos computacionales.

4. Diagrama Causa-Efecto del Problema





Objetivos

Objetivo General

Desarrollar una herramienta mediante plataformas móviles Android y iOS para facilitar el desarrollo y verificación de diagramas computacionales de complejidad media.

Objetivos Específicos

- Investigar los procesos, simbologías y representación para la construcción de diagramas de flujo y pseudocódigos.
- Investigar la metodología de desarrollo Scrum y los componentes necesarios en las correspondientes plataformas Android y iOS.
- Elaborar una herramienta de diseño y verificación de algoritmos computacionales definidos en diagramas de flujo y pseudocódigo, permitiendo a su vez la transformación entre ambas técnicas de representación.
- Evaluar el funcionamiento de diagrama de flujo o pseudocódigo de complejidad media, mediante la ejecución del algoritmo en la herramienta desarrollada.



Justificación

Un algoritmo computacional es uno de los mecanismos más utilizados para ayudar al manejo y dominio de la lógica de programación, esencial para el diseño y construcción de aplicaciones informáticas.

Para los estudiantes de la carrera de Ingeniería en Sistemas es de gran importancia el estudio y desarrollo de algoritmos computacionales.

Existen herramientas de escritorio que ayudan a comprobar el funcionamiento y operación de estos algoritmos, sin embargo, en la actualidad existen medios más flexibles, como las plataformas móviles, que nos permiten utilizar herramientas sin la necesidad de depender de un computador de escritorio.

Se justifica el presente proyecto en los siguientes términos:

- Aportar una herramienta de fácil acceso para que el estudiante pueda desarrollar la lógica de programación.
- Comprobar el funcionamiento de la lógica implementada en los diagramas de flujo.
- Guiar al estudiante para aprender a representar un algoritmo escrito en pseudocódigo mediante un diagrama de flujo, realizando una conversión de forma automatizada.
- Mejorar el rendimiento académico de los estudiantes en aquellas materias que requieren aplicación de la lógica computacional.
- Prescindir de un computador de escritorio para poder diseñar algoritmos computacionales, permitiendo plasmar ideas repentinas en un algoritmo.

CAPÍTULO I

MARCO TEÓRICO

DE LA

INVESTIGACIÓN



CAPÍTULO I

MARCO TEÓRICO DE LA INVESTIGACIÓN

1.1 Introducción

El aprendizaje y la preparación de estudiantes de las diversas carreras de ingeniería, en especial aquellas inmiscuidas en el ámbito informático, requieren la adquisición de conocimientos y habilidades para la resolución de problemas a través de computadoras por medio de algoritmos, que conlleva a la aplicación de metodologías y herramientas tecnológicas que refuercen el proceso de aprendizaje.

La implementación de una aplicación móvil como apoyo en el proceso de aprendizaje para el desarrollo de algoritmos brinda una herramienta portable de fácil acceso que facilitará el diseño de algoritmos computacionales y permite ejecutarlos en el dispositivo, otorgando la oportunidad al usuario de verificar la lógica utilizada en el algoritmo.

Las bases de esta investigación se detallan en el presente capítulo, a través de literatura técnica revisada e investigaciones realizadas en universidades extranjeras, debido a que durante el desarrollo de este trabajo no se pudo constatar investigaciones dentro del mismo campo en el Ecuador.

Además, se detallan y definen los diversos temas y términos requeridos para la comprensión y realización de la propuesta desarrollada, dividiéndose en las siguientes 6 categorías conceptuales: resolución de problemas, algoritmos computacionales, técnicas para el desarrollo de algoritmos, software para el desarrollo de algoritmos, Smartphone y aplicaciones móviles.



2.2. Antecedentes

Las investigaciones que aportan al desarrollo del proyecto están articuladas a estudios realizados por universidades en relación al desarrollo e implementación de herramientas de algoritmos para favorecer el aprendizaje de la lógica vinculada a los algoritmos, entre las que se destacan:

La investigación presentada por Ferri, Insfran y Fernández (2007) en el VIII Simposio Nacional de Tecnologías de la Información y las Comunicaciones en la Educación (SINTICE) propone la utilización de una herramienta case llamada MTPCASE para mejorar la enseñanza en las materias de ingeniería de software, debido a baja de adopción de metodologías de desarrollo de software a la fecha de desarrollo del artículo, lo cual incurría en altos costos por errores en el software.

Se realiza una revisión de la herramienta, indicando sus características, alcance y conceptos de ingeniería de software y desarrollo que maneja, destaca la posibilidad de realizar diagramas de flujo y estructura, pero sobre todo la capacidad de generación de código fuente para el lenguaje de programación C a partir de los diagramas descritos anteriormente.

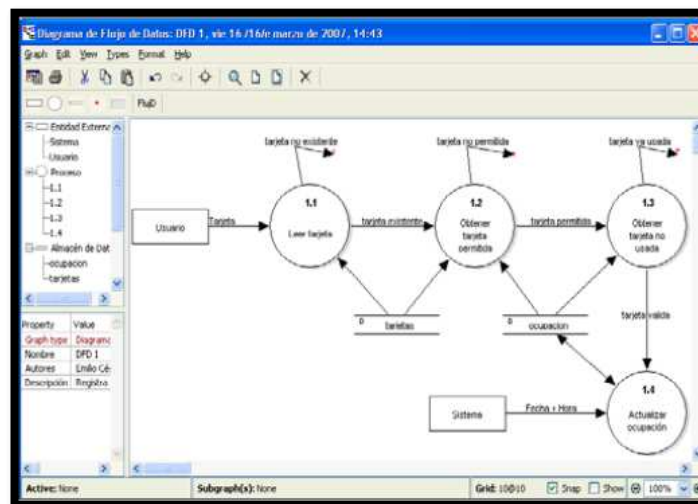


Ilustración 1: Diagrama de flujo de datos realizado en la herramienta MTPCASE

URL: <http://users.dsic.upv.es/~afernandez/files/publications/pdf/Ferri-et-al-TICAI.pdf>



Por otro lado el trabajo realizado por Arellano, Nieva, Solar, Arista (2012) en la Universidad Nacional de la Plata (SEDICI) y la Universidad del Istmo realiza una evaluación y comparación entre tres software cuyo objetivo es el desarrollo de algoritmos a través de una herramienta computacional. Se analizan las características de cada uno y sus respectivas ventajas y desventajas basadas en dos puntos de comparación los cuales son:

Haberse construido con el propósito de ser un recurso didáctico en la enseñanza-aprendizaje de algoritmos.

Emplear diagramas de flujo para la representación de las soluciones algorítmicas.

Tras el análisis de las herramientas Free DFD, Raptor y PSeInt, se realiza una propuesta para la creación e implementación de un nuevo software que cubra los inconvenientes encontrados en las aplicaciones evaluadas anteriormente en función de los puntos de comparación antes descritos, finalizando con tablas comparativas entre las tres herramientas estudiadas y la herramienta desarrollada.

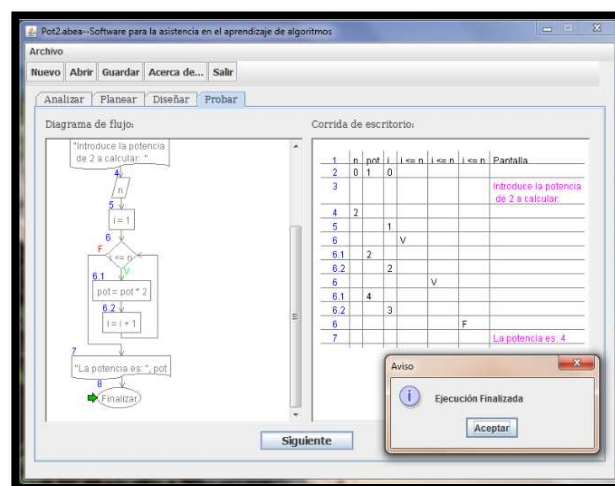


Ilustración 2: Interfaz de la etapa Probar correspondiente al software desarrollado

URL:http://sedici.unlp.edu.ar/bitstream/handle/10915/25531/Documento_completo.pdf?sequence=1

Además la investigación realizada por Carrizo, Corso y Olmedo (2015) en relación a la aplicación de una herramienta para la enseñanza de algoritmos y lógica



de programación en la Universidad Tecnológica Nacional de Argentina, establece el uso de una estrategia heurística para la resolución de problemas en conjunto con software libre para el desarrollo de algoritmos en forma de pseudolenguaje, explicando la manera en que se debe aplicar la estrategia de basada en las etapas de la heurística de Polya y cuáles son las características principales de la herramienta PSeInt.

Durante el desarrollo del artículo se explican las bases del desarrollo de un algoritmo y las principales fallas por parte de los estudiantes de materias relacionadas al campo informático y solución de problemas por computadora, dentro de lo cual se detalla la simbología utilizadas en los diagramas de flujo y se incluye el desarrollo de un caso de estudio, planteando un problema el cual es analizado y plasmado dentro de la aplicación PSeInt.

```
1 Proceso ejemplo1
2 bandera<-0;
3 Leer leg;
4 Mientras leg>0 Hacer
5     Leer nom,cant;
6     Si bandera=0 Entonces
7         mayor<-cant;
8         mayorNom<-nom;
9         bandera<-1;
10    Sino
11        Si cant>mayor Entonces
12            mayor<-cant;
13            mayorNom<-nom;
14        FinSi
15    FinMientras
16 Leer leg;
17 FinMientras
18 Escribir "El operario que mas produjo en la jornada fue", mayorNom;
19 FinProceso
```

Comandos

- /Hoja / Escribir
- /Dato / Leer
- [A=B:] Asignar
- [Si] Si-Entonces
- [Segun] Segun
- [Mientras] Mientras

Ilustración 3 Interfaz del programa PSeInt

URL: http://www.edutecne.utn.edu.ar/coini_2015/trabajos/F030_COINI2015.pdf

1.2 Definiciones Conceptuales

1.2.1 Resolución de problemas

Las etapas y técnicas para la resolución de problemas es parte esencial en el desarrollo de algoritmos computacionales que permitan encontrar solución a dichas situaciones.



Según lo establecido por Polya (1957), cuando se resuelven problemas, intervienen cuatro operaciones mentales:

1. Entender el problema
2. Trazar un plan
3. Ejecutar el plan (resolver)
4. Revisar

En relación al ámbito informático Joyanes (1996) expresa que “la primera fase de la resolución de un problema con computadora es el análisis del problema. Esta fase requiere una clara definición, donde se contemple exactamente lo que debe hacer el programa y el resultado o solución deseada” (p. 47).

Además, Hernández (2010) define que “es importante que se conozca lo que se desea que realice la computadora; mientras esto no se conozca no tiene mucho caso continuar con la siguiente etapa” (p. 8).

Para Pérez & Monzalvo (2012) “la resolución de todo problema exige tres grandes elementos: datos del problema, resultados solicitados y algoritmo de resolución” (p. 1).

En relación a la literatura revisada de los autores consultados es preciso tomar en cuenta que para plantear una solución por computadora es necesario seguir las bases para la resolución de problemas, permitiendo desarrollar algoritmos computacionales sólidos.

1.2.2 Algoritmos Computacionales

De los autores citados se considera que algoritmos computacionales representan una secuencia de pasos a ejecutarse en un orden determinado con la



finalidad de encontrar la solución a un problema determinado, los cuales pueden ser representados de diversas formas.

Para Joyanes (1996) los algoritmos representan “la secuencia ordenada de pasos, sin ambigüedades, que conducen a la solución de un problema dado y expresado en lenguaje natural, el cual debe ser: preciso, definido y finito” (p. 3).

Además, es necesario destacar lo manifestado por Rivera & Arrieta (2007) quienes indican que “un algoritmo no solo es un código fuente para obtener un programa para la computadora, sino, puede ser, tanto una herramienta para realizar su trabajo y sus planeaciones cotidianas, como una forma de argumentar” (p. 459).

De los autores de esta investigación se comparte que el desarrollo de algoritmos es un proceso que requiere un correcto análisis a una situación problemática que culmina en un diseño algorítmico aquel que establece una secuencia lógica de pasos que convergen en una solución expresada en un lenguaje similar al humano; estos pueden ser tomados como un punto de partida para una codificación en lenguaje de programación.

1.2.2.1 Diseño de Algoritmos

Se parte de lo manifestado anteriormente al considerar que la aplicación de metodologías junto con las bases relacionadas con los algoritmos computacionales, se establecen técnicas que permitan desarrollar y diseñar estos algoritmos.

Conforme la definición de Joyanes (1996) el algoritmo es “el proceso que convierte los resultados del análisis del problema en un diseño modular con refinamientos sucesivos que permitan una posterior traducción a un lenguaje” (p. 48).

Desde otra perspectiva Pérez & Monzalvo (2012), definen que “el diseño y construcción de algoritmos para computadora se basan en la definición clásica de



un sistema, el denominado diagrama de la “caja negra”: entrada- proceso-salida” (p. 2).

En función de obtener una mejor solución Quetglás, Toledo y Cerverón (2002) manifiestan que “durante el diseño es posible y aconsejable, realizar comparaciones entre algoritmos que resuelven el mismo problema” (p. 85).

Las definiciones enunciadas por los autores han llevado a la conclusión que el diseño de un algoritmo requiere un análisis de un problema para llegar a una solución, la cual deberá ser refinada y para ello se requiere realizar una retroalimentación de la situación problema y el resultado obtenido, además de ser el caso es recomendable tomar en cuenta soluciones similares existentes.

1.2.2.1.1 Técnicas para desarrollo de algoritmos

Diversos autores citados, entre ellos Pérez (2011), Monzalvo (2011) y Joyanes (1996) hacen referencia a las técnicas más utilizadas para el desarrollo y representación de algoritmos, de las cuales destacan los diagramas de flujo y el pseudocódigo.

Dentro de las técnicas para desarrollo de algoritmos Joyanes (1996) establece que “las dos herramientas más utilizadas comúnmente para diseñar algoritmos son: diagramas de flujo y pseudocódigos” (p. 48).

Para Pérez, Monzalvo (2011) “las técnicas de diagramación como los diagramas de flujo, el pseudocódigo, entre otras, son esquemas representativos además de un lenguaje para expresar algoritmos” (p. 2).

Por otra parte, Pérez & Monzalvo (2012) indican que “actualmente, los tres tipos fundamentales de herramientas de diagramación más utilizadas, dentro de los lenguajes algorítmicos son: los Diagramas de Flujo, el Pseudocódigo y los Diagramas N-S” (p. 2).

Además, Pérez, Monzalvo (2012) agregan que:



“los estudiantes consideran con mayores ventajas de uso y didácticas al diagrama de flujo, seguido del pseudocódigo y finalmente al diagrama N-S; y en cuanto al mayor nivel de preferencia de uso, el pseudocódigo es el primero, seguido del diagrama de flujo y al último el diagrama N-S” (p. 4).

En función de lo aquí manifestado, se muestra que existen varias formas de representar algoritmos, de las cuales las más destacadas son los diagramas de flujo y los pseudocódigos, ambos son ampliamente utilizados para la enseñanza en el ámbito educativo.

1.2.2.1.1 Pseudocódigo

Es una técnica para la representación de algoritmos más utilizadas, sobre todo en el inicio del proceso de aprendizaje del desarrollo de algoritmos, debido a que es una forma de expresión similar al lenguaje natural de las personas, siendo relativamente sencillo implementarlo en los distintos idiomas existentes.

El pseudocódigo es definido por Joyanes (1996) como “una herramienta de programación en la que las instrucciones se escriben en palabras similares al inglés o español, que facilitan tanto la escritura como la lectura de programas. En esencia, el pseudocódigo se puede definir como un lenguaje de especificaciones de algoritmos” (p. 49).

Conforme lo establecido por Quetglás, Toledo & Cerverón (2002) es “un lenguaje de especificación de algoritmos (no de programación) basado en un sistema notacional, con estructuras sintácticas y semánticas, similares a los lenguajes procedurales, aunque menos formales que las de éstos, por lo que no puede ser ejecutado directamente por un computador” (p. 89).

Para Pérez & Monzalvo (2012) consideran que es “una mezcla de lenguaje natural y símbolos, términos y otras características comúnmente utilizadas en uno o más lenguajes de alto nivel” (p. 2).



En relación a lo manifestado por los autores se define que el pseudocódigo es una forma de representación de algoritmos similar al lenguaje natural de las personas, utilizando una serie de palabras claves para definir el tipo de sentencia, proceso o estructura de control a utilizar.

1.2.2.1.1.2 Diagramas de Flujo

Es considerada la segunda técnica de preferencia por estudiantes para realizar la representación de algoritmos debido a que tiene un nivel de complejidad superior al pseudocódigo. Es representado por medio de figuras o símbolos estandarizados.

Además, el autor Joyanes (1990) agrega que es “la representación gráfica de los procedimientos y de la secuencia u orden en que deben ejecutarse éstos; en resumen, es la representación gráfica de la solución de un problema o de un procedimiento”.

Por otro lado, Quetglás, Toledo y Cerverón (2002) establecen que “son herramientas graficas utilizadas tanto para representar algoritmos, como en la ayuda en el diseño de programas. Están compuestos por una serie de símbolos, unidos con flechas, donde cada símbolo representa una acción distinta y las flechas el orden de realización de las acciones” (p. 90.)

Para Pinales & Velázquez (2014) “es la representación gráfica mediante símbolos especiales, de los pasos o procedimientos de manera secuencial y lógica que se deben realizar para solucionar un problema dado” (p. 17).

Las investigaciones de los diversos autores consultados identifican a los diagramas de flujo como una representación gráfica de algoritmos, por medio de símbolos estandarizados que se interconectan por medio de flechas para indicar las secuencias de los pasos a ejecutar.



1.2.3 Software para desarrollo de algoritmos

En la actualidad existe una variedad de software que sirve de apoyo para desarrollar y diseñar algoritmos utilizando las dos técnicas principales de diagramación antes descritas, siendo al presente parte del aprendizaje de los estudiantes.

En su análisis Carrizo, Cors, & Olmedo (2015) han comprobado que con la ayuda del software PSeInt:

“la diagramación se realiza de manera automática. En la medida que el alumno hace una especificación del algoritmo mediante el pseudolenguaje propuesto por la herramienta, el Diagrama de Flujo se genera de manera simultánea. La incorporación de esta herramienta en esta etapa agiliza y dinamiza de manera significativa la construcción del algoritmo por parte de los alumnos” (p. 6).

Por otro lado, Ferri, Insfran, & Fernández (2007) indican que “la implantación de una herramienta específica de estas características permite dar soporte al proceso de enseñanza-aprendizaje permitiendo a los alumnos dedicar mayor esfuerzo a las tareas creativas de diseño y razonamiento sobre el diseño y su posterior representación automática en elementos de programación” (p. 2).

Además, Arellano, Nieva, Solar, & Arista (2012) expresan que “el software para la enseñanza-aprendizaje de algoritmos estructurados es un recurso didáctico muy valioso que debe ser considerado por profesores y alumnos relacionados con cursos introductorios de algoritmos y programación a nivel superior” (p. 31).

Las investigaciones referenciadas concluyen que el uso de software para diseño de algoritmo representa un apoyo para el aprendizaje siendo un recurso didáctico importante ya que permite dinamizar y agilizar el proceso de construcción de un algoritmo automatizando tareas y otorgando flexibilidad en el proceso de corrección y verificación de errores.



1.2.4 Smartphone

El avance tecnológico ha permitido el auge de nuevos dispositivos electrónicos como los smartphones, los cuales cuentan con capacidades de procesamiento y sistemas operativos similares a las computadoras, por lo cual se han vuelto parte esencial del día a día y empiezas a jugar un papel importante tanto en el trabajo como en el aprendizaje.

En definición por Gómez & Ramírez (2011) “es un teléfono inteligente que tiene más funcionalidades que un teléfono celular común. Posee características similares a la de un sistema operativo, por lo cual son vulnerables a los virus o a los ataques al mismo sistema operativo tal como sucede con un computador normal (p. 21).

Un factor importante manifestado por Robledo (2012) indica que “los dispositivos móviles se están volviendo tan esenciales para la vida cotidiana de un estudiante como, podríamos decir, el desayuno” (p. 1).

Además, manifiestan Várelo, Redondo, & Palacín (2012) que “la generalización del uso de las tecnologías móviles en la sociedad favorece que la información llegue a mayor número de personas y en más ocasiones; por lo tanto, se presenta como una opción que incrementa la formación virtual” (p. 19).

Los concuerdan en la gran importancia que poseen los smartphones en nuestra vida cotidiana actualmente y debido a que sus capacidades han llegado a ser similares a las de un computador, se han convertido en una herramienta de utilidad para el trabajo y aprendizaje, por lo cual la creación de soluciones específicas para estas plataformas permite brindar soluciones portables para distintas áreas.



1.2.4.1 Android

En relación al nivel de aceptación y uso de los sistemas operativos móviles Gómez & Ramírez (2011) expresan que:

“es el sistema operativo más vendido a nivel mundial con un 52.5% de ventas mundiales por encima de Symbian y IOS de Apple lo cual está revolucionando actualmente el mercado mundial del Smartphone, haciendo de este sistema operativo móvil un gigante de la industria de las telecomunicaciones” (p. 58).

En el ámbito de su arquitectura Rivera, Cardona, & Franco (2012) expresan que:

“Está construido sobre el kernel de Linux. Además, se utiliza una máquina personalizada virtual que fue diseñada para optimizarlos recursos de memoria y de hardware en un entorno móvil. Android es de código abierto, y además puede ser libremente ampliado para incorporar nuevas tecnologías de vanguardia que van surgiendo” (p. 37).

En función de su estructura Ávila (2012) define que “este sistema operativo se compone de aplicaciones que se ejecutan en un framework Java de aplicaciones orientadas a objetos, todas ellas soportadas por la máquina virtual Dalvik con compilación en tiempo de ejecución” (p. 44).

El sistema operativo Android ha tenido un gran crecimiento desde su aparición, siendo actualmente el sistema operativo móvil más utilizado a nivel mundial y presente en una diversidad de plataformas móviles de diversos fabricantes.

1.2.4.1.1 *Java*

Los autores Gómez & Ramírez (2011) establecen que “Java es un lenguaje de programación y fue la primera plataforma informática creada por Sun



Microsystems en 1995. Es la tecnología subyacente que permite el uso de programas punteros como herramientas, juegos y aplicaciones de negocios” (p. 25).

La definición expresada por Gosling, Joy, Steele, Bracha, & Buckley (2015) establece que:

“[...] el lenguaje de programación Java es un lenguaje de propósito general, concurrente, basado en una clase y orientado a objetos. Está diseñado para ser lo suficientemente simple como para que muchos programadores logren fluidez en el lenguaje. El lenguaje de programación Java está relacionado con C y C ++, pero está organizado de manera diferente, con varios aspectos de C y C ++ omitidos y algunas ideas de otros lenguajes incluidos. Se pretende que sea un lenguaje de producción, no un lenguaje de investigación, por lo que, como sugirió C. A. R. Hoare en su obra clásica sobre el diseño del lenguaje, el diseño ha evitado incluir rasgos nuevos y no comprobados” (p. 1).

1.2.4.2 iOS

En relación a la estructura del sistema operativo Gómez & Ramírez (2011) lo definen que “sistema operativo IOS tiene cuatro capas de abstracción: la capa del núcleo del sistema operativo, la capa de servicios principales, la capa de medios de comunicación y la capa de Cocoa Touch” (p. 85).

Para Correa (2013) “es un sistema operativo orientado a equipos de escritorio que ha sido reinventado para dispositivos móviles. Ya que está basado en el sistema operativo informático más avanzado del mundo, OS X, iOS presenta un rápido desempeño y una estabilidad muy sólida” (p. 43).

Además, la Universidad de Alicante (2015) establece que “iOS es un derivado de OS X que, como a su vez está basado en Darwin BSD, es un sistema operativo UNIX (lo cual, por su estructura, implica robustez y fiabilidad)” (p. 17).



1.2.4.2.1 Swift

La empresa desarrolladora del este lenguaje de programación Apple Inc. (2017) define que:

“Swift es un nuevo lenguaje de programación para aplicaciones iOS, macOS, watchOS y tvOS que se basa en lo mejor de C y Objective-C, sin las limitaciones de la compatibilidad C. Swift adopta patrones de programación seguros y añade características modernas para hacer la programación más fácil, más flexible y más divertida” (párr. 1).

“Swift se siente familiarizado con los desarrolladores de Objective-C. Adopta la legibilidad de los parámetros nombrados de Objective-C y la potencia del modelo de objetos dinámicos de Objective-C. Proporciona un acceso sin fisuras a los frameworks existentes de Cocoa y la interoperabilidad mix-and-match con el código Objective-C. Partiendo de este terreno común, Swift introduce muchas nuevas características y unifica las partes procedimentales y orientadas a objetos del lenguaje” (párr. 2).

1.2.4.3 Aplicación Móvil

Las aplicaciones móviles son parte esencial de los sistemas operativos móviles, a través de ellas las personas pueden realizar tareas específicas para las cuales han sido desarrolladas, existiendo tantas aplicaciones que van desde el ocio hasta herramientas empresariales y educativas.

Por otro lado, Herrera & Fennema (2011) expresan que “los dispositivos móviles constituyen una de las tecnologías más usadas y presentan ventajas en cuanto a su portabilidad y a su sensibilidad al contexto. Este tipo de e-learning mediado por tecnologías móviles se conoce como mobile-learning (m-learning)” (p. 620).

En relación al aprendizaje Espinoza (2010) expresa que:



“el haber seleccionado un dispositivo y herramientas conocidas y utilizadas previamente por ellos, ha facilitado el proceso de aprendizaje y, por tanto, ha mejorado su interpretación de los resultados. Se considera por tanto importante los conocimientos adquiridos previamente, con respecto a la tecnología que va a utilizarse” (p. 170).

Así mismo The New Media Consortium & Universitat Oberta de Catalunya (2012) expresan que “en el ámbito educativo, las aplicaciones móviles han ganado popularidad; una prueba de ello es la consideración por parte del Consejo Asesor de que el horizonte de adopción de esta tecnología es inmediato” (p. 5).

Para Fermín, Raúl, & Javier (2013) “el uso de la tecnología móvil en el aula tiene unos beneficios relativamente importantes en el aprendizaje de los estudiantes” (p. 15).

Los autores Campión, Filv, & Ochoa (2014) agregan que “la seleccin de apps educativas y su utilizacin con los dispositivos mviles constituye un contexto, amplio, flexible y verstil y posibilita el aprendizaje y acceso a los contenidos dentro y fuera del aula, incluso fuera del horario escolar” (p. 8).

Los estudios de los autores establecen que en la actualidad las aplicaciones mviles ofrecen una solucin para diversos problemas existentes en la sociedad, esto es comparable con el software que se desarrolla para las computadoras, ofreciendo soluciones empresariales o a situaciones de la vida diaria; dentro del mbito educativo las aplicaciones mviles ofrecen portabilidad al estudiante teniendo la posibilidad de utilizarlas dentro y fuera del aula, prcticamente en cualquier lugar y hora.

1.2.5 Metodologa Scrum

Scrum es una metodologa de desarrollo que forma parte del manifiesto gil, la metodologa expresa una serie de roles para los proyectos y un modelo de desarrollo basado en reuniones e iteraciones.



Schwaber & Sutherland (2016) establecen que el marco de trabajo Scrum consiste en los Equipos Scrum y sus roles, eventos, artefactos y reglas asociadas. Cada componente dentro del marco de trabajo sirve a un propósito específico y es esencial para el éxito de Scrum y para su uso.

Según Alcázar (2016) Scrum es un proceso iterativo e incremental utilizado para la construcción de productos. Esto significa que el proceso se compone de diferentes interacciones a las que llamaremos Sprints. Estas interacciones o sprints son fijos en el tiempo y se recomienda que tengan una duración de 1 a 4 semanas máximo. El objetivo de estos sprints es el de construir un incremento del producto que potencialmente se pudiera utilizar por parte de los clientes. Por tanto, no nos serviría entregar algo que no pudiéramos utilizar al final del proceso.

Al utilizar scrum como metodología de desarrollo se crean porciones entregables y ejecutables del producto final, lo cual permite mostrar un progreso incremental en cada iteración, de esta manera se pueden hacer las correcciones apropiadas por cada entregable probado por el cliente de forma temprana y no cuando está terminado software en su totalidad.

1.3 Conclusiones

En virtud de lo consultado tanto en los temas de investigación como las definiciones conceptuales y el respectivo análisis se concluye lo siguiente:

El estudio e interpretación de los artículos científicos relacionados establecen un punto de partida para la realización del análisis del problema del presente trabajo, enfocándose en el refuerzo de los métodos de aprendizaje de algoritmos computacionales por medio de un aplicativo móvil.

Las dos técnicas principales por preferencia y facilidad para el desarrollo de algoritmos son los diagramas de flujo y el pseudocódigo, por lo cual al desarrollar un aplicativo móvil como apoyo en la elaboración de algoritmos este es basado en estas dos técnicas.



El auge de los Smartphones y las avanzadas capacidades de procesamiento actuales de los mismos, ha logrado que sean una tecnología de gran importancia en varias actividades diarias, siendo de esta manera un aporte de portabilidad y accesibilidad las aplicaciones que permiten resolver problemas puntuales.

Los sistemas operativos móviles iOS y Android son las principales plataformas en la actualidad, es decir aquellas cuentan con el mayor número de usuarios, por lo cual el aplicativo es desarrollado para ambas plataformas.

CAPÍTULO II

**DIAGNÓSTICO O
ESTUDIO DE
CAMPO**



CAPÍTULO II

DIAGNÓSTICO O ESTUDIO DE CAMPO

2.1. Introducción.

En el presente capítulo se describe la estrategia metodológica empleada para la formulación y resolución del proceso investigativo, así como las técnicas y herramientas para la recolección de datos y el análisis de los resultados obtenido en el presente trabajo; mencionando el conjunto de procedimientos lógicos, técnicos y operacionales comprendidos en todo el proceso de investigación; con la finalidad de permitir manifestar y estudiar los supuesto del presente trabajo y de reconstruir los datos, a partir de los conceptos teóricos. Además, este tiene como finalidad:

- Comprender el tipo de investigación realizada en el presente trabajo de titulación.
- Determinar la población y universo del proyecto, en base a los objetivos planteados.
- Puntualizar los métodos y técnicas para la recolección de los datos.
- Establecer el plan para la recolección y análisis de los datos.

2.2. Tipo(s) de Investigación

2.2.1. Investigación Exploratoria

La realización de la investigación exploratoria, como lo expresa Palella & Pestana (2012) “permite focalizar el tópico de interés, formular el problema y/o delimitar futuros temas de investigación.” (p. 92).



Tomando como referencia lo manifestado, este tipo de investigación fue aplicada para conocer las razones que generaban dificultades en la elaboración de algoritmos computacionales en los estudiantes, los mismo que son causantes de un bajo rendimiento académico o incluso la deserción de la carrera.

2.2.2. Investigación de Campo

Para el presente investigación también se realizó una investigación de campo, Palella & Pestana (2012) afirman que la investigación de campo consiste en la recolección de datos directamente. (p. 88).

Por otro lado Arias (2012) define “Es la realidad donde ocurren los hechos, sin manipular o controlar las variables. Estudia los fenómenos sociales en su ambiente natural. El investigador no manipula variables debido a que esto hace perder el ambiente de naturalidad en el cual se manifiesta y desenvuelven el hecho.” (p.31).

En el presente trabajo de titulación, se aplicó la investigación de campo, enfocándose en el proceso de enseñanza-aprendizaje de la carrera de Ingeniería en Sistemas con respecto a los algoritmos computacionales, siendo estos parte fundamental en el área de desarrollo de software, encontrando falencias en los estudiantes a la hora del DEACON, donde se realizó un trabajo en conjunto con los discentes, docentes y autoridades de la FACCI, obteniendo datos relevantes que permitieron hacer un análisis de la situación actual en el contexto estudiado.

Al emplear la investigación de campo en este estudio, permitió obtener nuevos conocimientos en el campo de la realidad estudiada, y analizar la situación actual de los estudiantes que cursan la materia de Fundamento de Programación.



2.3. Métodos de Investigación

2.3.1. Lógico Inductivo.

Este método permitió identificar que los estudiantes tienen problemas al momento de diseñar algoritmos computacionales, concluyendo después de someterse a un proceso investigativo que es una de las mayores causas de deserción en los primeros niveles de la carrera de Ingeniería en Sistemas.

2.3.2. Método Bibliográfico.

El método de investigación bibliográfica es el sistema que se sigue para obtener información contenida en documentos. En sentido más específico, el método de investigación bibliográfica es el conjunto de técnicas y estrategias que se emplean para localizar, identificar y acceder a aquellos documentos que contienen la información pertinente para la investigación (López de Prado, s.f.).

El presente proyecto de titulación, basa sus fundamentos teóricos en la investigación de documentos relevantes al tema del DEACON. Estas fuentes de información, ya sean impresas o digitales, manifiestan la importancia de los Algoritmos Computacionales en el ámbito académico.

2.4. Herramientas de Recolección de Datos.

La recolección de datos es fundamental para generar una investigación altamente estructurada, ésta a su vez, tienen como finalidad buscar información que sea útil a la presente investigación.



Las herramientas de recolección de datos es en principio cualquier recurso de que pueda valerse el investigador para acercarse a los fenómenos y extraer de ellos información (Rivero, s.f.). En el presente trabajo de titulación se utilizó, la observación, la entrevista y la encuesta como herramienta de recolección de datos.

2.4.1. Observación.

La observación es el registro de forma visual de lo que acontece, verificando, clasificando y consignando acontecimientos pertinentes de acuerdo con algún esquema previsto y según el problema que se estudia.

Al aplicar este método se identificaron falencias en la etapa del análisis del problema, siendo una etapa de suma importancia, debido a que por ella podemos entender y comprender el problema a resolver, para dar una óptima solución. Por lo general, esta situación surge debido al bajo nivel de análisis que se encuentra en los estudiantes.

Del análisis se parte a la etapa del DEACON, el cual depende de un buen planteamiento del problema propuesto para su correcta elaboración; se observó que los estudiantes de la FACCI tienen dificultad para el diseño de los algoritmos puesto que en ocasiones no conocen en su totalidad los componentes y sus funcionalidades.

2.4.2. Encuesta.

García Fernando, Ibañez, & Alvira (1994) definen que una encuesta es una investigación realizada sobre una muestra de sujetos representativa de un colectivo



más amplio, que se lleva a cabo en el contexto de la vida cotidiana, utilizando procedimientos estandarizados de interrogación, con el fin de obtener mediciones cuantitativas de una gran variedad de características objetivas y subjetivas de la población.

Para la realización del presente estudio, se necesitó información relevante a la problemática tratada y a su entorno, por lo cual se aplicó este instrumento de recolección de datos, permitiendo realizar la tabulación y en base a los resultados poder emitir conclusiones, respecto al estado actual de los estudiantes de la FACCI frente al DEACON.

2.4.3. Entrevista

En una entrevista además de obtener los resultados subjetivos del encuestado acerca de las preguntas del cuestionario, se puede observar la realidad circundante, anotando el encuestador además de las respuestas tal cual salen de la boca del entrevistado, los aspectos que considere oportunos a lo largo de la entrevista (Torres, 2014).

Se aplicó la entrevista, realizando una serie de preguntas a los docentes que imparten las materias de Fundamento de Programación y Programación Orientada a Objetos de la FACCI, con la finalidad, en base a su experiencia, conocer cuáles son las causas desde su perspectiva, por las cuales los estudiantes tienen problemas en la DEACON.



2.5. Fuentes de Información de datos.

Fuente de información es todo aquello que suministre una noticia, una información o un dato, que pueda transmitir conocimiento (López Carreño, 2008). En la presente sección se determinarán las fuentes de investigación primarias y secundarias que fueron utilizadas en el proceso del presente estudio.

2.5.1. Fuentes primarias

Las fuentes primarias contienen información original, que ha sido publicada por primera vez y que no ha sido filtrada, interpretada o evaluada por nadie más. Son producto de una investigación o de una actividad eminentemente creativa (Silvestrini Ruiz & Vargas Jorge, 2008).

Por lo manifestado en el texto anterior, en el presente estudio se consideró como fuentes primarias, la información brindada por los estudiantes de primer y segundo nivel de la FACCI, así como también, por los docentes que imparten las materias de Fundamentos de Programación y Programación Orientada a Objetos del periodo 2016-2017(1); cuya información fue obtenida por la aplicación de las técnicas de Entrevista, Encuesta y Observación:

- Dificultad en el análisis de los Problemas planteados para la correcta elaboración del DEACON en los estudiantes de la FACCI.
- Resultados obtenidos por la ejecución de las entrevistas a los docentes que imparten las materias de Fundamentos de Programación y Programación Orientada a Objetos de la FACCI.
- Encuesta dirigida a los estudiantes del primer y segundo nivel de la FACCI.



2.5.2. Fuentes secundarias.

Las fuentes secundarias contienen información primaria, sintetizada y reorganizada. Están especialmente diseñadas para facilitar y maximizar el acceso a las fuentes primarias o a sus contenidos. Componen la colección de referencia de la biblioteca y facilitan el control y el acceso a las fuentes primarias. (Silvestrini Ruiz & Vargas Jorge, 2008).

Como fuentes secundarias, se utilizó los conocimientos impartidos por otros autores, que sirven como conocimientos previos para la presente investigación.

Se utilizó bibliografías existentes relacionada con la telemática de la presente investigación, tales como:

- Libros y Artículos referente a Algoritmos Computacionales.
- Documentación sobre resolución de problemas mediante Algoritmos Computacionales.
- Libros, artículos y documentación sobre técnicas o herramientas para el desarrollo de algoritmos.
- Documentación sobre software para el desarrollo de algoritmos.
- Libros y artículos del Sistema Operativo Android.
- Libros y artículos del Sistema Operativo iOS.

2.6. Instrumental Operacional

2.6.1. Estructura y características de lo(s) instrumento(s) de recolección de datos

2.6.1.1. Encuesta

La encuesta se realizó según a los objetivos planteados y en base a la exigencia de la presente investigación. Esta fue estructurada en base a preguntas



cerradas de las cuales se recolectó información necesaria para la elaboración del presente trabajo.

La presente encuesta cuenta con un número de 13 preguntas cerradas, permitiendo seleccionar las opciones planteadas por los investigadores, agilizando el proceso de realización, para no generar un gasto de tiempo excesivo o el aburrimiento de los estudiantes encuestados. (Ver anexo 1)

Esta encuesta ayudó a procesar la información de forma ágil, permitiendo dar el punto de vista referente a la presente investigación de cada encuestado.

2.6.1.2. Entrevista

Esta técnica de recolección de datos, permitió dialogar con los profesores que imparten las materias de Fundamento de Programación y Programación Orientada a Objetos de las FACCI en el periodo 2017-2018(1), realizando preguntas referentes a la realización de un óptimo DEACON por los estudiantes de sus respectivas materias; puntualizando sus consideraciones en cuanto a las posibles causas por la cual se genera en los estudiantes dicha dificultad.

La entrevista fue estructurada con preguntas abiertas permitiendo al docente dar su opinión con total libertad; éstas preguntas fueron enfocadas a la experiencia que tienen estos docentes a la hora de enseñar las técnicas para el DEACON y la dificultad para ciertos estudiantes realizar un algoritmo computacional óptimo. (Ver anexo 2).

2.6.1.3. Observación

En el presente estudio se utilizó la técnica de observación bajo las directrices de una observación no estructurada la cual emplea el procedimiento de la “observación participante” en la que el investigador actúa como observador y se familiariza con el lugar para posteriormente volverse participante activo, desarrollar



un plan de muestreo de eventos y seleccionar las posiciones para llevar a cabo la observación, para ello se recaba información referida al ambiente, los participantes, sus actividades e interacciones, la frecuencia y duración de los eventos para ir tomando “notas de campo”, “notas de observación”, “notas teóricas”, “notas metodológicas” y “notas personales”, obteniendo así información sobre la dinámica de grupo y el fenómeno a estudiar (Aniorte Hernández, 2016).

2.7. Estrategia Operacional para la recolección y tabulación de datos

2.7.1. Plan de recolección de los datos

La recolección de datos, es parte fundamental de una investigación, con esta podemos realizar un análisis de la situación actual del problema y encaminar la investigación a una solución óptima.

Para realizar la recolección de datos de forma correctas, se debe realizar un plan; en la siguiente tabla se muestra el plan realizado para la presente investigación, en base a los indicadores, *¿Quién?*, *¿Cómo?*, *¿Cuándo?*, *¿Dónde?*

Plan de recolección de Datos	
¿Quién aplicara los instrumentos?	Los encargados de aplicar los instrumentos son los responsables de la presente investigación, orientados por su tutor encargado
¿Cómo se recolectarán los datos?	La forma de recolectar los datos suele ser una de las más importantes, debido



	a esto, en el presente trabajo de titulación, se utilizarán como instrumentos de recolección de datos, la encuesta, entrevista y observación mediante el trabajo de campo realizado. La encuesta se realizó por medio de formularios electrónicos.
¿Cuándo se recolectarán los datos?	El tiempo en que se llevara a cabo este plan puede variar debido a la disponibilidad de los alumnos y profesores. Por lo que se asume que el tiempo de recolección de los datos serian aproximadamente 1 mes, comenzando desde el 1 de junio del 2017
¿Dónde se aplicarán los instrumentos?	La presente investigación es enfocada a la FACCI de la ULEAM, por ende, será en esta institución donde se aplicará este plan de recolección de datos.

Tabla 1 Plan de recolección de Datos

2.7.2. Plan de tabulación de los datos

Para el desarrollo del estudio, se aplicó la estadística descriptiva e inferencial.

La estadística descriptiva trata del recuento, ordenación y clasificación de los datos obtenidos por las observaciones, mientras que la estadística inferencial plantea y resuelve el problema de establecer previsiones de conclusiones generales



sobre una población a partir de los resultados obtenidos de una muestra (Borrego, 2008).

Para realizar el proceso de tabulación, se agruparon los datos en tablas de frecuencia absoluta y así obtener un porcentaje, para luego realizar su respectiva representación gráfica, siendo una de las técnicas fundamentales que permite resumir la información, debido a su facilidad de comprensión incluso entre aquellas personas que no poseen conocimientos de estadística (Borrego, 2008).

Por lo manifestado anteriormente, se consideró el siguiente plan:

Nº	Acción	Detalle
1	Recolección y Organización de los datos	Se recolectan los datos para poder realizar un análisis a temas puntuales del presente trabajo, luego se realizó la organización de estos datos ya que esto facilitará la comprensión de la información obtenida.
2	Realizar tablas de frecuencia absoluta	Registrar la información recolectada en tablas de frecuencia absoluta, permitiendo clasificar para una posterior comprensión de los datos
3	Representar resultados mediante Gráficas	La representación gráfica permite una mayor interpretación de los datos, permitiendo observar con mayor facilidad, la variación de los datos.
4	Análisis de las Gráficas	Las gráficas facilitan el análisis de los datos recolectados, gracias a su fácil comprensión, estos serán sustento del análisis planteado.



5	Interpretar los resultados	La interpretación de los datos, es una tarea de suma importancia, ya que nos da una idea de la situación actual, y nos permite proponer una solución acorde a lo necesitado.
---	----------------------------	--

2.7.3. Plan de análisis e interpretación de los datos

El análisis consiste en la búsqueda de explicaciones y de comprensión. Durante esa búsqueda es posible que se postulen, consideren y elaboren teorías y conceptos (Blaxter, Hughes, & Tight, 2002).

En base a los resultados de las encuestas, se realizará un análisis donde se interpretará la problemática, y si la solución planteada tendrá acogida por parte de los estudiantes que realicen el DEACON como parte del aprendizaje.

Para realizar este proceso, se responderá las siguientes preguntas:

Plan análisis e interpretación de los datos	
¿Quién realizará el análisis e interpretación?	Los autores del presente trabajo de titulación, serán los encargados de la realización del análisis e interpretación.
¿Cómo se realizará el análisis e interpretación?	Comparando los resultados obtenidos a través de las herramientas de recolección de datos, con la problemática planteada, aclarando las inquietudes generadas.
¿Cuándo se realizará el análisis e interpretación?	El análisis e interpretación de los resultados se realizó después de la



	recolección de los datos, en el mes de junio del 2017, por ende, este se llevó a cabo en el mes de julio del 2017
¿Dónde se realizará el análisis e interpretación?	En la FACCI de la ULEAM

Tabla 2 Plan de Análisis e Interpretación de los datos

2.8. Plan de Muestreo

2.8.1. Segmentación.

La técnica de segmentación permite dividir la población beneficiaria, en grupos de acuerdo a sus necesidades, con el fin de enfocar la tabulación y análisis de los datos según al grupo que pertenecen. Es por ello que, en el presente trabajo de titulación, enfocado en la comunidad académica de la FACCI, se dividió la población en dos grupos:

- Los estudiantes de primero y segundo nivel de la FACCI del periodo 2017-2018(1).
- Los docentes que imparten las materias Fundamento de Programación y Programación Orientada a Objetos de la FACCI del periodo 2017-2018(1).

2.8.2. Técnica de Muestreo

La Técnica de Muestreo casual o incidental fue la que se implementó en el desarrollo de la presente propuesta, ya que se trata de un proceso en el que el investigador selecciona directa e intencionadamente los individuos de la población.

El caso más frecuente de este procedimiento es utilizar como muestra los individuos a los que se tiene fácil acceso (Univ. Nacional de Entre Ríos, s.f.), en este



caso los estudiantes de primero y segundo nivel de la FACCI, puesto que en los primeros niveles es donde se origina la problemática planteada. Por otro lado, los docentes que imparten las materias de Fundamentos de Programación y Programación Orientada a Objetos, a quienes se entrevistará debido a que se encuentran involucrados directamente con el problema.

2.8.3. Tamaño de la muestra

Como se lo estipuló en la segmentación, el presente trabajo se enfocó en dos muestras. La primera los estudiantes de primero y segundo nivel de la FACCI en el periodo 2016-2017(1) y la segunda en los profesores que imparten las materias de Fundamento de Programación y Programación Orientada a Objetos en la FACCI en el periodo 2016-2017(1).

En el siguiente cuadro se presenta la información antes mencionada de forma más detallada.

Muestra de estudiantes:

Nivel	Paralelo	Cantidad
Primer Nivel	A	40
	B	26
	C	38
	Subtotal	104
Segundo Nivel	A	40
	B	40
	C	41
	D	30
	Subtotal	151
Total	255	

Tabla 3 Muestra de los estudiantes de la FACCI



Muestra de profesores:

Materia	Cantidad
Fundamento de Programación	1
Programación Orientada a Objetos	1
Total	2

Tabla 4 Muestra de los profesores de la FACCI

Para realizar el muestreo de la población se calcula aplicando la siguiente formula utilizando el nivel de confianza del 95%:

$$n = \frac{Z^2 P Q N}{Z^2 P Q + e^2}$$

Donde:

n = Tamaño de la muestra

N = Universo

e = Margen de error admisible. (En este caso se trabajará con el 5 %)

P = Probabilidad de ocurrencia

Q= Probabilidad de no ocurrencia

Z = Confiabilidad 95%

$$n = \frac{(1,96)^2(0,5)(0,5)(255)}{(1,96)^2(0,5)(0,5) + (0,05)^2} = \frac{244,902}{1,5979} = 153,26491 = 153 \approx$$

Se calcula la constante de muestreo

$$x = \frac{100 \times n}{N}$$

$$x = \frac{100 \times 153}{255} = 60,10$$



$$x = 60,10\%$$

2.9. Presentación y Análisis de los resultados

2.9.1. Análisis de los resultados.

2.9.2.1. Análisis general de la encuesta dirigida estudiantes.

Antes de emitir un informe general en relación a los objetivos definidos en el presente proyecto de titulación, se analiza de manera independiente la encuesta dirigida a los estudiantes (Ver anexo 3):

- Según la encuesta realizada, el 98% de los estudiantes consideran a los algoritmos computacionales como la base en el desarrollo del software; siendo el Diagramas de Flujo y el Pseudocódigo las técnicas más conocidas, prefiriendo de entre ellas el Diagrama de Flujo por un 54% encima del Pseudocódigo.
- Según la encuesta realizada, el 78% de los estudiantes conocen o han utilizado herramientas desarrolladas para computadores, con la finalidad de facilitar el diseño de algoritmos. Por lo consiguiente, con el 59%, se afirma que algunas de estas herramientas permiten realizar la respectiva verificación de los algoritmos, concluyendo que, en la actualidad, existen herramientas que facilitan el DEACON a través de un computador
- Según la encuesta realizada, el 82% de los estudiantes desconocen herramientas desarrolladas para dispositivos móviles, con la finalidad de facilitar el diseño de algoritmos. Por lo consiguiente, con el 88%, se niega que algunas de estas herramientas permitan realizar la respectiva verificación de los algoritmos, concluyendo que, en la actualidad, son escasas o desconocidas las herramientas que facilitan el DEACON a través de un dispositivo móvil



- Según la encuesta realizada, el 93% de los estudiantes considera que las herramientas desarrolladas para computadores o dispositivos móviles que facilitan el DEACON, favorecen el proceso de aprendizaje. Adicional a esto, el 83% considera relevante, que estas herramientas permitan realizar una conversión de Diagramas de Flujo a Pseudocódigo y viceversa.
- Según la encuesta realizada, los Sistemas Operativos con mayor frecuencia de uso por los estudiantes son Android y iOS teniendo una 79% y 12% respectivamente.
- Según la encuesta realizada, 75% de los estudiantes, concuerda que la dificultad para un correcto DEACON, genera un bajo rendimiento académico generando dificultad avanzar los diferente nivele de la carrera y en algunos casos desertando de sus estudios.

2.9.2.2. Análisis general de la entrevista dirigida a docentes.

Una vez aplicada la entrevista a los docentes que imparten las materias de Fundamentos de Programación y Programación Orientada a Objetos, se pudo concluir que ambos coinciden en lo manifestado, indicando que a su consideración es fundamental el dominio de los algoritmos computacionales por parte de los estudiantes de la FACCI, debido a que uno de los perfiles con los que egresa un estudiante es el de programador, el cual necesita de una buena lógica para poder codificar. También consideran que los algoritmos computacionales son la base para el desarrollo de software, y agregan que en otros países los algoritmos son impartidos en todas las carreras universitarias, dando a entender que no solo son enfocados al desarrollo de software.

En cuando a la dificultad que se genera al no comprender el DEACON causando inconvenientes en el avance académico en la carrera, los profesores consideraron que estos inconvenientes se generan en materias relacionadas a los algoritmos y manifiestan que se debe a la deficiencia en la lógica matemáticas



Por otro lado, consideran que, de las técnicas existentes para representar los algoritmos computacionales, el Diagrama de Flujo tiene mayor facilidad, en comparación con Pseudocódigo debido a su entorno visual. Adicionalmente alegaron que la utilización de herramientas como Pselnt, incentivan al estudiante a practicar el DEACON.

En cuanto a la verificación de los algoritmos, consideran que una herramienta que permita esto, facilitaría a los estudiantes la comprensión de los algoritmos; así como también permitir realizar una conversión de Diagrama de Flujo a Pseudocódigo y viceversa, dando la oportunidad de realizar una comparación entre ambas técnicas permitiendo comprender la técnica de mayor dificultad como lo consideran al Pseudocódigo.

Por último, manifestaron que unas de las principales causas que conlleva a los estudian a bajar su rendimiento académico, y en algunos casos a la deserción de la carrera, se debe al bajo nivel de lógica matemática con la que ingresan los estudiantes, siendo esta fundamental para el DEACON.

CAPÍTULO III

DISEÑO DE LA PROPUESTA



CAPÍTULO III

DISEÑO DE LA PROPUESTA

3.1. Introducción

El presente capítulo detalla la planificación del proyecto desarrollado, partiendo de los resultados obtenidos de la investigación y la aplicación de las técnicas de recolección de datos utilizadas, realizando la extracción de los requerimientos funcionales y no funcionales.

La aplicación de la metodología Scrum conllevó a la agrupación de requerimientos en cinco Sprints, de tal forma que al final de cada uno de ellos se obtuvo un entregable funcional, además se presentan los artefactos utilizados como son: las historias de usuario, casos de uso, pila de producto, pilas de Sprints, empleados por el equipo de trabajo en coordinación con el Scrum Master y el Product Owner para desarrollar el producto destinado a los Stakeholders.

Además se detalla el presupuesto y los recursos tanto humanos como tecnológicos requeridos para llevar a cabo el desarrollo en conjunto con las fases de la metodología aplicada.

3.2. Descripción De La Propuesta

La propuesta busca llegar al mayor número de usuarios móviles relacionados con el desarrollo de algoritmos computacionales a través de las plataformas de mayor distinción, para permitir elaborar soluciones a problemas puntuales por medio de las dos técnicas principales para el desarrollo de algoritmos antes mencionadas como son los diagramas de flujo y el pseudocódigo, brindando la posibilidad de realizar una conversión entre ellas, es decir de un pseudocódigo ingresado generar



el correspondiente diagrama o viceversa, adicionando la posibilidad de ejecutar y probar la solución elaborada, lo cual favorece a la detección y corrección de errores.

La aplicación cuenta con las siguientes características:

- Herramienta de diagramación de flujo de datos
- Herramienta de edición de pseudocódigo
- Conversión de diagrama de flujo a pseudocódigo
- Conversión de pseudocódigo a diagrama de flujo
- Ejecución de algoritmos

Herramienta de diagramación de flujo de datos

La herramienta de diagramación de flujo de datos, provee un espacio de dibujo similar a un lienzo, el cual se encarga de contener las figuras o componentes del diagrama que representan el algoritmo y conforman el diagrama de flujo como tal, permitiendo añadir, eliminar, conectar y mover figuras dentro del espacio de trabajo, el cual se expande o contrae según la posición de los componentes para proveer el espacio requerido por el diagrama desarrollado. Además, dispone de una paleta de componentes, la cual incluye los símbolos requeridos para la representación del algoritmo.

Herramienta de edición de pseudocódigo

La herramienta de edición de pseudocódigo permite al usuario representar el algoritmo con un lenguaje más informal, para lo cual el beneficiario utiliza palabras claves estandarizadas para definir la solución a un problema, la herramienta resalta las palabras reservadas utilizadas por el usuario en una tonalidad diferente al resto del contenido, por otro lado cada paso de la solución planteada deberá ser definida en una línea del editor, lo cual indica a la aplicación el orden correcto de la secuencia de pasos a seguir.



Conversión de diagrama de flujo a pseudocódigo

Partiendo de un diagrama de flujo ya planteado, la herramienta de conversión se encarga de iterar a través de los componentes del diagrama transformando dicha representación en su equivalente palabra reservada dentro del editor de pseudocódigo incluyendo su asociación con el texto establecido dentro del componente y representando la separación de la secuencia de pasos entre componentes por saltos de líneas y la inclusión de tabulaciones correspondientes a los flujos condicionales y estructuras de repetición.

Conversión de pseudocódigo a diagrama de flujo

A partir de un pseudocódigo previamente establecido la herramienta itera entre cada línea del algoritmo definida en el editor, interpretando las palabras claves utilizadas por el usuario y el contenido asociado a ellas, posteriormente convierte cada palabra clave encontrada a su correspondiente representación o componente de diagrama de flujo de datos añadiendo el texto asociado dentro del símbolo resultante y uniendo los símbolos nuevos con aquellos que le preceden en función del orden de líneas escritas en el editor, definiendo de esta forma el flujo o secuencia correspondiente a la solución.

Ejecución de algoritmos

La ejecución de algoritmos se encarga de recorrer el diagrama de flujo definido por el usuario, identificando el tipo de componente del paso actual para ejecutar el comportamiento adecuado sobre el texto contenido del símbolo que se está procesando, permitiendo mostrar resultados en pantalla por medio del componente display y el ingreso de valores por medio del componente manual input que pertenecen a la simbología estándar de diagramas de flujo; además al momento de ocurrir errores al ejecutar la solución, esta es detenida y marca el paso donde ocurrió el error equivalente a una excepción de un lenguaje de programación. Cabe mencionar que, si el usuario ha realizado el ingreso del algoritmo por medio del



editor de pseudocódigo, este es transformado al diagrama de flujo correspondiente previo a la ejecución.

3.2.1. Especificaciones Técnica.

Desarrollo:

- Plataformas de Desarrollo: Android Y IOS.
- Lenguaje de Programación: Java y Swift.
- Librerías:
 - Android: Canvas
 - iOS: SwitSVG
- IDE: Android Studio, Xcode

3.2.2. Objetivos

- Emplear metodologías ágiles para la gestión y desarrollo del presente proyecto.
- Definir los recursos humanos, tecnológicos y económico para el desarrollo del software.
- Desarrollar una herramienta móvil para el DEACON en las principales plataformas móviles, iOS y Android.
- Albergar la herramienta móvil en las respectivas tiendas de aplicaciones de iOS y Android



3.2.3. Definición de Recurso

3.2.3.1. Humano

Los recursos humanos se definen en base a las personas involucradas en la resolución del presente proyecto de titulación, las cuales, formaron parte del proceso y dieron su aporte a la realización del proyecto.

Recursos Humanos	Función
Docentes y Estudiantes de la FACCI de la ULEAM	Dotaron de información esencial para la realización del presente proyecto de titulación por medio de entrevistas y encuestas
Arteaga Vera José Cristóbal	Tutor y guía del presente proyecto de titulación
Carlos Gabriel Guevara Vera	Autor y Desarrollador del presente trabajo
Jordan Javier Párraga Cedeño	Autor y Desarrollador del presente trabajo

Tabla 5 Recursos Humanos

3.2.3.2. Tecnológicos

Un recurso tecnológico, es un medio que se vale de la tecnología para cumplir con su propósito. Los recursos tecnológicos pueden ser tangibles (como una computadora, una impresora u otra máquina) o intangibles (un sistema, una aplicación virtual).²

En el presente trabajo de titulación se emplearon los siguientes recursos tecnológicos:

Recursos Tecnológicos	Función
Computador Personal	Utilizada para desarrollar la aplicación en las plataformas iOS y Android

² Dato extraído de la página web: <http://definicion.de/recursos-tecnologicos/>



Impresora	Utilizadas para la impresión de la documentación correspondiente al presente trabajo.
Tablet	Utilizada para probar y testear el funcionamiento de la aplicación en un entorno iOS real durante el proceso de desarrollo y lanzamiento.
Smartphone	Utilizado para probar y testear el funcionamiento de la aplicación en un entorno Android real durante el proceso de desarrollo y lanzamiento.
Play Store	Su función es alojar en la tienda de aplicaciones de Google la aplicación para estar a la disposición de ser descarga e instalada por los usuarios del sistema operativo Android.
App Store	Su función es alojar en la tienda de aplicaciones de Apple la aplicación para estar a la disposición de ser descarga e instalada por los usuarios del sistema operativo iOS.
Android Studio	Es el IDE oficial para desarrollo en Android patrocinado por Google Inc. y desarrollado por JetBrains.
Xcode	Es el IDE oficial para desarrollo en iOS desarrollado por Apple Inc.
Bitbucket	Repositorio de código y proyectos que emplea Git para el control de versiones, siendo utilizado para alojar las fuentes de la aplicación desarrollada para iOS y Android.
Office 365	Es utilizada para el desarrollo de las encuestas y su tabulación, compartición de archivos del proyecto y desarrollo de la documentación.

Tabla 6 Recursos Tecnológicos

3.2.3.3. Económicos (presupuesto)

El presupuesto económico invertido en el presente trabajo de titulación, fue asumido por parte de los autores del mismo.

Obteniendo el siguiente presupuesto:



Cantidad	Medida	Descripción	Costo	Valor Total
Recursos Humanos				
224	h	Desarrollo del software	\$ 8,00	\$ 1792,00
120	h	Realización del trabajo de investigación	\$ 8,00	\$ 960,00
	h	Tutorías	\$ 0,00	\$ 0,00
Recursos Tecnológicos				
Equipos informáticos				
375	hoja	Impresiones	\$ 0,025	\$ 9,38
2	unidad	Computadora Personal	\$ 0,00	\$ 0,00
1	unidad	Tablet	\$ 0,00	\$ 0,00
2	unidad	Smartphone	\$ 0,00	\$ 0,00
2	unidad	Android Studio	\$ 0,00	\$ 0,00
1	unidad	Xcode	\$ 0,00	\$ 0,00
Servicios				
2	unidad	Cuentas en Bitbucket	\$ 0,00	\$ 0,00
2	unidad	Cuentas en Office 365	\$ 0,00	\$ 0,00
1	unidad	Cuenta en Play Store	\$ 25,00	\$ 25,00
1	unidad	Cuenta en App Store	\$ 99,00	\$ 99,00
Recursos Material				
1	unidad	Resmas de Papel	\$ 4,00	\$ 4,00
2	unidad	Carpeta	\$ 0,60	\$ 1,20
Otros Recursos				
17	movilización	Transporte	\$ 35,00	\$ 35,00
		Subtotal		\$ 2925,58
		IVA 14%		\$ 409,58
		Descuento 2%		\$ 58,51
		Total		\$ 3276,65

Tabla 7 Recursos Económicos



3.3. Etapas De La Propuesta

3.3.1. Metodología

Scrum es una metodología de desarrollo de software adaptable a cualquier tipo de proyecto, por lo que es considerado también como un método de gestión de proyectos, con la ejecución de un conjunto de buenas prácticas para trabajar colaborativamente en equipo.

En Scrum se realizan entregas parciales y regulares del producto final, conocidas como iteraciones o normalmente llamadas sprints. El objetivo de un sprint es construir el incremento de un producto que potencialmente se pudiera utilizar por parte de los clientes. Estos sprints son fijos en el tiempo y se recomienda que tengan una duración de 1 a 4 semanas.³

Por lo expuesto anteriormente, en el presente trabajo se implementó Scrum como metodología de desarrollo, obteniendo el siguiente proceso de trabajo:

³ Información obtenida del curso de Gestión de proyectos con metodologías Ágiles y enfoques Lean en Miríada X: <https://miriadax.net/cursos>

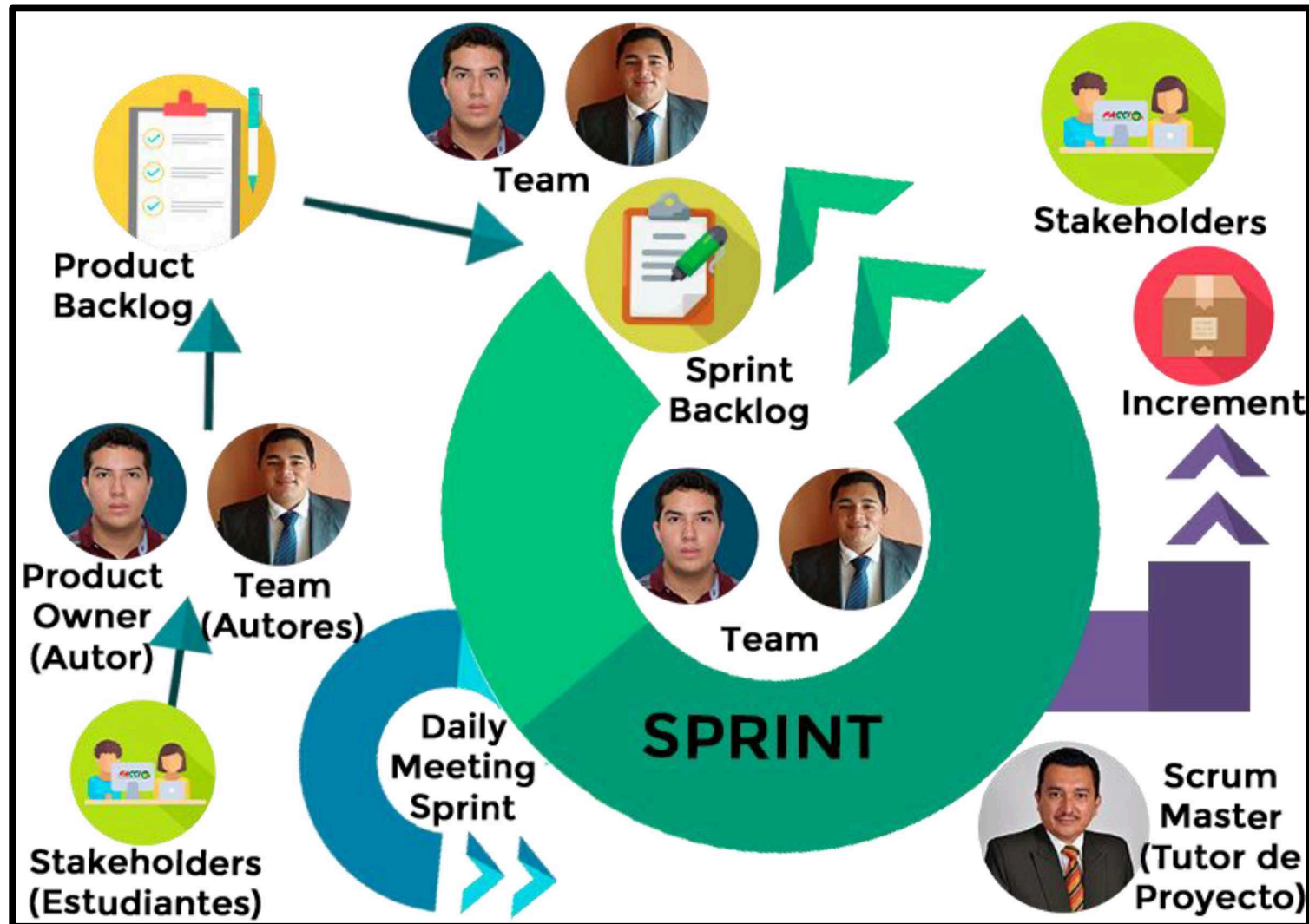


Ilustración 4 Sprint del proyecto (Iconos Graphic Resources, S.L)



3.3.1.1. Desarrollo Incremental

El desarrollo de las aplicaciones móviles para iOS y Android se llevó a cabo bajo la metodología ágil Scrum, el proyecto se dividió en 5 Sprints, resultando al final de cada uno de ellos un ejecutable del producto total.

El desarrollo incremental, consistió en la realización de iteraciones entre el desarrollo y entregas de productos, al final de cada Sprint se utilizaron sujetos de prueba para comprobar el correcto funcionamiento de cada entregable resultante del Sprint en relación a los requerimientos resultantes de la investigación.

Debido a que el producto no está dirigido a un usuario o entidad en específico, por el contrario, es de uso general para personas afines a las carreras de informática, estas pruebas con sujetos involucrados en el campo fue considerada como la entrega del producto en cada Sprint.

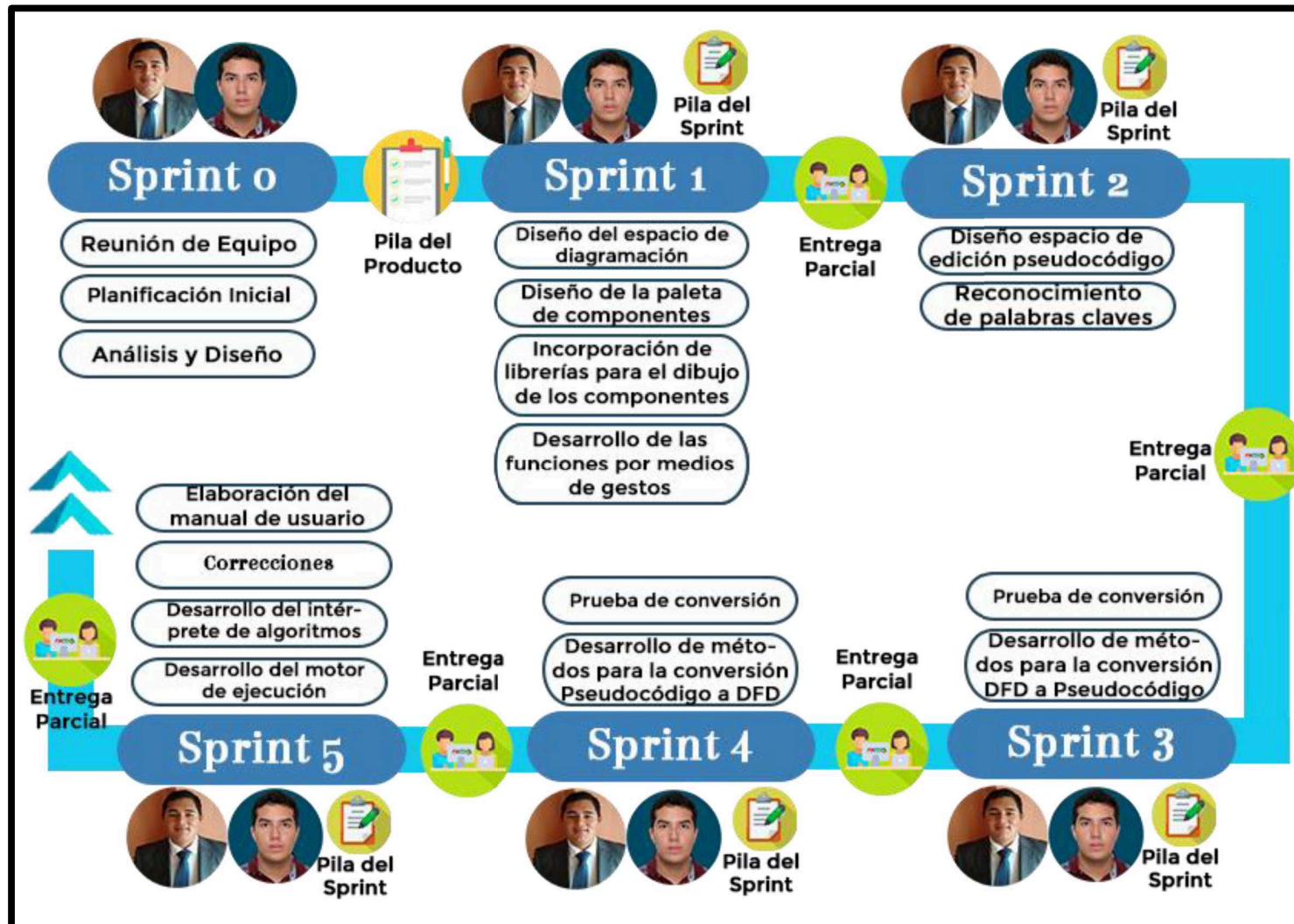


Ilustración 5 Sprints del Proyecto



3.3.2. Personas y Roles

Persona	Contacto		
Arteaga Vera José Cristóbal	130465085-4	0983750408 / jose.arteaga@live.uleam.edu.ec	Scrum Master
Guevara Vera Carlos Gabriel	131102940-7	0987234013 / e1311029407@live.uleam.edu.ec	Scrum Team / Product Owner
Párraga Cedeño Jordan Javier	131282888-0	0997644371 / e1312828880@live.uleam.edu.ec	Scrum Team

Tabla 8 Personas y Roles

3.3.3. Fases

Al aplicar la metodología Scrum, el desarrollo del proyecto se realizó en base a iteraciones, de las cuales se obtuvo como resultado un entregable funcional, que forma parte del producto final, es decir podemos considerar que cada iteración es un proyecto en si mismo, que forma parte del proyecto principal. Cada una de las iteraciones consistió en cinco fases: Reunión de Planificación del Sprint, Scrum Diario, Trabajo de Desarrollo durante el Sprint, Revisión del Sprint y Retrospectiva del Sprint.



Ilustración 6 Fases del Scrum

3.3.3.1. Reunión de Planificación

En base al grupo de requerimientos extraídos en la investigación realizada, se definieron cinco Sprints, para lo cual previo al desarrollo de cada uno de ellos se realizaron reuniones definiendo el resultado a entregar al final del Sprint y como se iba a desarrollar.



3.3.3.2. Scrum Diario

Se realizaron reuniones previo al inicio de cada jornada de desarrollo para sincronizar las actividades y coordinar el trabajo diario en función del último sprint y las tareas pendientes.

3.3.3.3. Trabajo de Desarrollo

Partiendo de los requisitos y las actividades de la pila del sprint en curso, se procedió al desarrollo de las aplicaciones móviles, verificando el correcto funcionamiento de las unidades y módulos en relación al requerimiento que representan.

3.3.3.4. Revisión

Durante el término de cada Sprint y al final del desarrollo, se realizaron pruebas de verificación, comprobando el correcto funcionamiento del producto en relación a los requerimientos y evitando fallos en la aplicación que causen comportamientos no esperados, además se socializaron, analizaron y resolvieron los problemas ocurridos durante el desarrollo.

3.3.3.5. Retrospectiva

Se realizó un análisis al final de cada Sprint con el objetivo de estudiar las relaciones entre los involucrados, los procesos y las herramientas, buscando mejorar cada una de las iteraciones futuras.

3.3.4. Requisitos

La aplicación tiene la finalidad de ser un apoyo en el proceso de aprendizaje en el diseño y desarrollo de algoritmos computacionales, pretendiendo brindar facilidades y automatización al momento de realizar el DEACON. Para ello se ha



tomado como referencia las principales dificultades observadas en los estudiantes de la Facultad de Ciencias Informáticas de la ULEAM.

3.3.4.1. Requerimientos Funcionales

Los requerimientos han sido levantados a partir de las actividades y necesidades de los estudiantes de Fundamentos de Programación en relación a las técnicas de representación de algoritmos utilizadas por ellos y las principales dificultades presentes en las técnicas utilizadas.

REQUERIMIENTOS DE LA APLICACIÓN	
Paleta de componentes	Mostar un panel o paleta que contenga los símbolos disponibles para realizar la representación de diagramas de flujo.
Espacio de diseño adaptable	El área o lienzo de diagramación deberá ajustar su tamaño en función de los componentes que contiene y la ubicación de los mismos. Además, permitir hacer zoom in y zoom out.
Creación de componentes	Los componentes deben ser creados a partir de la paleta de componentes, tocando o arrastrando desde la misma el símbolo que se desea; y ser ubicado en una ubicación por defecto o donde termina el evento de arrastre del usuario en el lienzo de diagramación.



Eliminación de componentes	Debe permitirse eliminar componentes específicos que haya seleccionado el usuario.
Selección de componente	Cuando el evento de toque en pantalla se realiza sobre un componente del lienzo, dicho componente debe mostrar un marco de selección con opciones de conexión con otros componentes y de eliminación.
Actualización del posicionamiento de componentes	Debe permitirse mover los componentes del lienzo a otra ubicación cuando están seleccionados o en su defecto cuando existe un toque continuo sobre la figura, debe seleccionarse y proceder con el movimiento de dicho componente.
Unión de componentes	Los componentes del lienzo deben poder ser conectados entre si, por medio de las opciones de conexión que muestran al ser seleccionados.
Edición del contenido de los componentes	El texto de los componentes debe poder ser modificado por medio de un campo de texto que aparece sobre el teclado tras dar doble click sobre un componente reflejando los cambios realizados sobre el mismo.



Editor de texto para pseudocódigo	Se debe incluir un editor de texto para ingresar un algoritmo por medio de pseudocódigo.
Resaltar palabras claves utilizadas	Se deben resaltar las palabras reservadas que forman parte de la descripción de un paso del algoritmo dentro del pseudocódigo con otro color.
Tabulación automática	Al crear una nueva línea esta debe tabularse automáticamente en función de las condiciones y bucles repetitivos abiertos.
Selección de editor de algoritmos	Debe permitirse cambiar del editor de diagramas de flujo al editor de pseudocódigo y viceversa.
Conversión de diagrama de flujo a pseudocódigo	Se debe realizar la conversión de un diagrama de flujo realizado a su equivalente en pseudocódigo.
Conversión de pseudocódigo a diagrama de flujo	Se debe realizar la conversión de un pseudocódigo realizado a su equivalente en diagrama de flujo.
Ejecución de algoritmos	Los algoritmos realizados en la aplicación deben poder ser ejecutados de forma similar a un programa, pidiendo las entradas y mostrando las



	salidas correspondientes que se hayan descrito.
Marcado de errores en ejecución	Cuando se produce un error al ejecutar un algoritmo, debe detenerse la ejecución y marcar con un marco rojo el paso donde ocurrió el error.
Guardado de trabajos	La aplicación debe permitir guardar un archivo que describa el algoritmo realizado.
Carga de trabajos	La aplicación debe permitir cargar archivos previamente realizados y guardados.

Tabla 9 Requisitos Funcionales

3.3.4.2. Requerimientos no Funcionales

Estos requerimientos no forman parte de la funcionalidad final del sistema, sin embargo, constituyen una parte importante para su desarrollo.

REQUERIMIENTOS NO FUNCIONALES	
Lenguaje de Programación	Se requiere que las aplicaciones sean desarrolladas en Swift y Java para plataformas móviles.
Versión iOS	La versión del sistema operativo soportado por la aplicación debe ser mínimo iOS 8.0.



Versión Android	La versión del sistema operativo soportado por la aplicación debe ser mínimo Android 4.2.
Versionamiento	Alojar el código fuente en un repositorio de código.
Acceso	La aplicación de estar disponible para su descarga desde las correspondientes tiendas de aplicaciones (App Store y Play Store).

Tabla 10 Requisitos no funcionales

3.3.4.3. Historias de Usuario

Identificador (ID) de la Historia	Enunciado de la Historia				Criterios de Aceptación			
	Rol	Característica / Funcionalidad	Razón / Resultado	Número de Escenario	Criterio de Aceptación (Título)	Contexto	Evento	Resultado / Comportamiento Esperado
UH-MDF-DFD Diagramación de Flujo de Datos								
UH-MDF-DFD-01	Como un usuario	Necesito crear un nuevo algoritmo	Con la finalidad de materializar un nuevo algoritmo sin modificar los que ya están guardados	1	Creación de un espacio de trabajo vacío para diagramas	Solicitando un nuevo diagrama en la vista principal	Cuando crea el nuevo espacio de trabajo	Iniciar con un lienzo y editor de pseudocódigo vacíos sin alterar trabajos existentes mostrando por defecto el editor de diagramas
UH-MDF-DFD-02	Como un usuario	Necesito visualizar los componentes de diagramación	Con la finalidad de poder crear nuevos componentes en el lienzo para la representación	1	Visualización de la paleta de componentes	Agregando un nuevo componente	Al presionar la opción de un nuevo componente	Mostrar una vista que contenga los símbolos para la construcción del flujo

		disponibles	ción del algoritmo					
UH-MDF-DFD-03	Como un usuario	Necesito agregar nuevos componentes al lienzo de diagramación	Con la finalidad de construir un algoritmo	1	Agregar componente al lienzo por presión continua	Al mantener presionado continuamente un componente de la paleta este generará una figura para el lienzo	Al generar un componente desde la paleta	Deberá ser agregado donde termine de arrastrarlo el usuario
				2	Agregar componente al lienzo por toque de pantalla	Al dar un toque sobre un componente de la paleta este generará una figura para el lienzo	Al generar un componente desde la paleta	Deberá ser agregado en una ubicación por defecto en el lienzo
UH-MDF-DFD-04	Como usuario	Necesito mover los componentes	Con la finalidad de reorganizar	1	Mover componente del lienzo por	Al mantener presionado continuamente un	Presión continua y arrastre de componente	La figura debe actualizar su posición según el evento y tener

		tes del lienzo	el diagrama de flujo		presión continua	componente del lienzo este será seleccionado y se moverá conforme se mueva el gesto activo		como posición final la localización donde termina el mismo
				2	Mover componente del lienzo por toque de pantalla	Cuando un componente del lienzo este previamente seleccionado se moverá conforme se mueva el gesto activo	Cuando se arrastre de componente	La figura debe actualizar su posición según el evento y tener como posición final la localización donde termina el mismo
UH-MDF-DFD-05	Como usuario	Necesito seleccionar las figuras del lienzo	Para poder ver las opciones disponibles sobre ellas	1	Selección de componente	Al dar click sobre un componente no seleccionado	Cuando se selecciona un componente	Mostrar un cuadro con opciones alrededor del componente seleccionado
				2	Deselección de	Al dar click fuera de un componente seleccionado	Cuando se deselecciona un componente	Ocultar el cuadro de opciones del componente que

					componente			estaba seleccionado
UH-MDF-DFD-06	Como usuario	Necesito eliminar componentes	Para quitar del lienzo un paso del algoritmo que ha sido descartado	1	Eliminación de componentes	Al dar click sobre la opción de eliminar componente	Cuando se elimina el componente	Debe quitarse del lienzo de diagramación y eliminar las conexiones que están asociadas a él con otras figuras
UH-MDF-DFD-07	Como usuario	Necesito conectar los componentes del lienzo	Para representar la secuencia del flujo	1	Conexión de componentes	Al dar click sobre la opción de conexión del componente y arrastrar hacia un punto de conexión de otro	Cuando se conectan dos componentes	Debe crearse un conector representado por una o varias líneas rectas que crean una ruta entre ambas sobre el lienzo
UH-MDF-DFD-08	Como usuario	Necesito seleccionar	Para poder ver las opciones	1	Selección de conexión	Al dar click sobre una conexión no seleccionada	Cuando se selecciona una conexión	Mostrar un cuadro con opciones alrededor de la

		conexiones	disponibles sobre ellas					conexión seleccionada
				2	Deselección de conexión	Al dar click fuera de una conexión seleccionada	Cuando se deselecciona una conexión	Ocultar el cuadro de opciones de la conexión que estaba seleccionada
UH-MDF-DFD-09	Como usuario	Necesito eliminar conexiones entre componentes	Para poder corregir la secuencia del flujo	1	Desconexión de componentes	Al dar click sobre la opción de eliminar conexión	Cuando se elimina la conexión	Debe removerse del lienzo y quedar libre los componentes para otras conexiones
UH-MDF-DFD-10	Como usuario	Necesito cambiar el texto de la figura	Para ingresar la instrucción a ejecutar	1	Selección de componente para edición	Al dar doble click sobre el componente	Cuando se inicia la edición de un componente	Debe mostrarse el teclado con una caja de texto encima para poder ingresar la instrucción
				2	Modificación del contenido	Al escribir en el cuadro de texto	Cuando se modifica el texto	Debe reflejarse el cambio en el componente que se seleccionó para edición

UH-MDF-DFD-11	Como usuario	Necesito guardar el trabajo	Para poder visualizar o editarlo en un futuro	1	Guardado de diagrama	Conforme se modifica el diagrama debe guardar su estado	Cuando se modifica el diagrama	Debe guardarse el estado actual para su posterior recuperación
UH-MDF-DFD-12	Como usuario	Necesito cargar trabajos anteriores	Para poder visualizarlo o editarlo	1	Carga de diagrama	En la vista inicial quiero poder visualizar los trabajos anteriores y poder abrirlos	Cuando se abre un trabajo guardado	Debe mostrarse el lienzo con el ultimo estado de ese diagrama

Tabla 11 Historias de Usuario Diagramación

Identificador (ID) de la Historia	Enunciado de la Historia				Criterios de Aceptación			
	Rol	Característica / Funcionalidad	Razón / Resultado	Número de Escenario	Criterio de Aceptación (Título)	Contexto	Evento	Resultado / Comportamiento Esperado
UH-MEP-EPS Edición de Pseudocódigo								
UH-MEP-EPS-01	Como un usuario	Necesito crear un nuevo algoritmo	Con la finalidad de materializar un nuevo algoritmo sin modificar los que ya están guardados	1	Creación de un espacio de trabajo vacío para pseudocódigo	Solicitando un nuevo pseudocódigo en la vista principal	Cuando crea el nuevo espacio de trabajo	Iniciar con un lienzo y editor de pseudocódigo vacíos sin alterar trabajos existentes mostrando por defecto el editor de pseudocódigo
UH-MEP-EPS-02	Como usuario	Necesito escribir sobre un editor de texto	Para poder escribir la secuencia de pasos en forma	1	Edición de pseudocódigo	Escribir sobre un editor de texto de	Cuando se escribe por medio del teclado	Debe escribir sobre el editor de texto

			de pseudocódigo			múltiples líneas.		
UH-MEP-EPS-03	Como usuario	Necesito diferenciar las palabras reservadas	Para identificar el tipo de instrucciones utilizadas	1	Marcación de palabras claves	Buscar palabras reservadas en el pseudocódigo	Cuando se escribe una palabra reservada	Debe cambiar el color de las palabras reservadas
UH-MEP-EPS-04	Como usuario	Necesito mantener tabulado el pseudocódigo	Para identificar las estructuras condicionales y bucles de repetición	1	Tabulación del pseudocódigo	Durante la edición del pseudocódigo y se identifica un salto de línea	Al realizar un salto de línea	Debe realizarse una tabulación en función del número de estructuras de control abiertas

Tabla 12 Historias de Usuario Pseudocódigo

Identificador (ID) de la Historia	Enunciado de la Historia				Criterios de Aceptación			
	Rol	Característica / Funcionalidad	Razón / Resultado	Número de Escenario	Criterio de Aceptación (Título)	Contexto	Evento	Resultado / Comportamiento Esperado
UH-MCD-CDP Conversión de Diagrama a Pseudocódigo								
UH-MCD-CDP-01	Como usuario	Necesito convertir el diagrama de flujo a pseudocódigo	Para poder observar la correcta forma de representación de dicha técnica	1	Conversión de diagrama de flujo a pseudocódigo	Si existen modificaciones en el diagrama, cuando se ejecuta o cambia a vista de pseudocódigo	Conversión de técnica de representación	Pseudocódigo equivalente al diagrama de flujo

Tabla 13 Historias de Usuario Conversión Diagramas – Pseudocódigo

		Enunciado de la Historia			Criterios de Aceptación			
Identificador (ID) de la Historia	Rol	Característica / Funcionalidad	Razón / Resultado	Número de Escenario	Criterio de Aceptación (Título)	Contexto	Evento	Resultado / Comportamiento Esperado
UH-MCP-CPD Conversión de Pseudocódigo a Diagrama								
UH-MCP-CPD-01	Como usuario	Necesito convertir el pseudocódigo a diagrama de flujo	Para poder observar la correcta forma de representación de dicha técnica	1	Conversión de pseudocódigo a diagrama de flujo	Si existen modificaciones en el pseudocódigo, cuando se ejecuta o cambia a vista de diagramación	Conversión de técnica de representación	Diagrama de flujo equivalente al Pseudocódigo

Tabla 14 Historias de Usuario Conversión Pseudocódigo – Diagrama

Identificador (ID) de la Historia	Enunciado de la Historia				Criterios de Aceptación			
	Rol	Característica / Funcionalidad	Razón / Resultado	Número de Escenario	Criterio de Aceptación (Título)	Contexto	Evento	Resultado / Comportamien to Esperado
UH-MEA-EJA Ejecución de Algoritmos								
UH-MEA- EJA-01	Como un usuario	Necesito ejecutar el algoritmo	Para comprobar el funcionamiento y visualizar los resultados	1	Componentes correctamente conectados	Al tener un inicio y fin correctos y las secuencias correctamente conectadas	Ejecuci ón del algorith o	Se ejecutará a partir de la figura de inicio que solo tiene conectada su salida y se recorrerá en el orden de conexión ejecutando las instrucciones correspondient es pidiendo

								entradas y mostrando salidas
				2	Componentes sin conexiones	Al no tener conexión entre sus componentes	Ejecución del algoritmo	Mostrar un mensaje de aviso y detener la ejecución
				3	Varios componentes de inicio	Cuando existe más de un componente que expresa inicio o fin	Ejecución del algoritmo	Mostrar un mensaje de aviso y detener la ejecución
				4	Error o excepción durante ejecución	Cuando se está procesando una	Ejecución del algoritmo	Mostrar un mensaje de aviso y detener la

						instrucción y existe una excepción como similar a los lenguajes de programación		ejecución, marcando con un marco rojo el paso en el cual ocurrió la excepción
--	--	--	--	--	--	---	--	--

Tabla 15 Historias de Usuario Ejecución de Algoritmo

3.3.5. Artefactos

3.3.5.1 Pila del Producto

Id de la Pila	Id de la Historia	Historia	Grupo	Estado	Tiempo en Días	Iteración (Sprint)	Prioridad	Finalizado
A	UH-MDF-DFD	Diagramación de Flujo de Datos	Diseño de Algoritmos	Terminado	23	1	Alta	100%
B	UH-MEP-EPS	Edición de Pseudocódigo	Diseño de Algoritmos	Terminado	4	2	Alta	100%
C	UH-MCD-CDP	Conversión de Diagrama a Pseudocódigo	Conversión de Algoritmos	Terminado	6	3	Media	100%
D	UH-MCP-CPD	Conversión de Pseudocódigo a Diagrama	Conversión de Algoritmos	Terminado	6	4	Media	100%
E	UH-MEA-EJA	Ejecución de Algoritmos	Ejecución de Algoritmos	Terminado	17	5	Alta	100%

Tabla 16 Pilas del Producto

3.3.5.2 Pila del Sprint

Grupo	ID de la Pila	Historia	ID de Tarea	Tarea	Tipo	Responsable	Iteración (Sprint)	Prioridad	Estado	Aprobado
Diseño de Algoritmos	A	Diagramación de Flujo de Datos	1	Diseño del espacio de diagramación	Diseño	Jordan Párraga	1	Media Alta	100 %	✓
			2	Diseño de la paleta de componentes	Diseño	Jordan Párraga		Media Alta	100 %	✓
			3	Creación de archivos con la representación de componentes de diagramación	Diseño	Gabriel Guevara		Media Alta	100 %	✓
			4	Incorporación de librerías para dibujar sobre componentes desde archivos svg e imágenes vectoriales	Programación	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
			5	Desarrollo de las funciones de diagramación por	Programación	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓

				medio de gestos en pantalla						
			6	Pruebas de diagramación	Testing	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
			7	Retroalimentación	Team Scrum	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
Diseño de Algoritmos	B	Edición de Pseudocódigo	8	Diseño del espacio de edición de pseudocódigo	Diseño	Gabriel Guevara	2	Media Alta	100 %	✓
			9	Reconocimiento de palabras claves en el editor	Programación	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
			10	Pruebas de diseño de algoritmos en editor	Testing	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
			11	Retroalimentación	Team Scrum	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
Conversión de Algoritmos	C	Conversión de Diagrama a Pseudocódigo	12	Desarrollo de métodos para mapeo de datos del diagrama a pseudocódigo	Programación	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓

			13	Pruebas de conversión	Testing	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
			14	Retroalimentación	Team Scrum	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
Conversión de Algoritmos	D	Conversión de Pseudocódigo a Diagrama	15	Desarrollo de métodos para mapeo de datos del pseudocódigo a diagrama de flujo	Programación	Gabriel Guevara - Jordan Párraga	3	Alta	100 %	✓
			16	Pruebas de conversión	Testing	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
			17	Retroalimentación	Team Scrum	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
Ejecución de Algoritmos	E	Ejecución de Algoritmos	18	Desarrollo del motor de ejecución	Programación	Gabriel Guevara	4	Alta	100 %	✓
			19	Desarrollo del intérprete de algoritmos	Programación	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
			20	Pruebas de ejecución de algoritmos	Testing	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓

			21	Retroalimentación	Team Scrum	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓
			22	Entrega del producto final	Team Scrum	Gabriel Guevara - Jordan Párraga		Alta	100 %	✓

Tabla 17 Pilas del Sprint

3.3.5.3 Sprints

Sprint	Inicio	Días	Jornada																							
1	27/03/2017	23	4 hrs	L	M	X	J	V	L	M	X	J	V	L	M	X	J	V	L	M	X	J	V	L	M	X
				27 - mar	28 - mar	29 - mar	30 - mar	31 - mar	03 - abr	04 - abr	05 - abr	06 - abr	07 - abr	10 - abr	11 - abr	12 - abr	13 - abr	14 - abr	17 - abr	18 - abr	19 - abr	20 - abr	21 - abr	24 - abr	25 - abr	26 - abr
			Tareas Pendientes	2 2	2 2	2 2	2 1	2 1	2 0	2 0	2 0	1 9	1 9	1 9	1 9	1 8	1 8	1 8	1 7	1 7	1 7	1 7	1 6	1 6	1 6	1 5
			Horas Pendientes	2 2 4	2 2 0	2 1 6	2 1 2	2 0 8	2 0 4	2 0 0	1 9 6	1 9 2	1 8 8	1 8 4	1 8 0	1 7 6	1 7 2	1 6 8	1 6 4	1 6 0	1 5 6	1 5 2	1 4 8	1 4 4	1 4 0	1 3 6
Sprint Backlog																										
Tarea	T. Est (hrs)	Estad o	Respon sable	Esfuerzo																						

Diseño del espacio de diagramación	23	Terminado	Jordan Párraga	23	20	17	14	11	7	4	1															
Diseño de la paleta de componentes	13	Terminado	Jordan Párraga	13	13	13	13	13	13	13	13	13	9	5	1											
Creación de archivos con la representación de componentes de diagramación	9	Terminado	Gabriel Guevara	9	5	1																				

Incorporación de librerías para dibujar sobre componentes desde archivos svg e imágenes vectoriales	7	Terminado	Gabriel Guevara - Jordan Párraga	7	7	7	7	4																				
Desarrollo de las funciones de diagramación por medio de gestos en pantalla	27	Terminado	Gabriel Guevara - Jordan Párraga	27	27	27	27	27	27	23	25	23	21	19	15	11	7	3										
Pruebas de diagramación	8	Terminado	Gabriel Guevara - Jordan Párraga	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	6	4	2							

Retroalimentación	1	Terminado	Gabriel Guevara - Jordan Párraga	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
-------------------	---	-----------	----------------------------------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	--

Tabla 18 Sprint 1

Sprint	Inicio	Días	Jornada				
2	27/04/2017	4	4 hrs	J	V	L	M
				27 - abr	28 - abr	29 - abr	30 - abr
			Tareas Pendientes	15	13	12	11
			Horas Pendientes	136	122	118	117
Tarea	T. Est (hrs)	Estado	Responsable	Esfuerzo			
Diseño del espacio de edición de pseudocódigo	5	Terminado	Gabriel Guevara	1			

Reconocimiento de palabras claves en el editor	7	Terminado	Gabriel Guevara - Jordan Párraga	3			
Pruebas de diseño de algoritmos en editor	6	Terminado	Gabriel Guevara - Jordan Párraga	6	4		
Retroalimentación	1	Terminado	Gabriel Guevara - Jordan Párraga	1	1	1	

Tabla 19 Sprint 2

Sprint	Inicio	Días	Jornada						
3	01/05/2017	6	4 hrs	X	J	V	L	M	X
				01 - may	02 - may	03 - may	06 - may	08 - may	09 - may
			Tareas Pendientes	11	11	11	10	9	8
			Horas Pendientes	117	106	102	100	96	95
Tarea	T. Est (hrs)	Estado	Responsable	Esfuerzo					
Desarrollo de métodos para mapeo de datos del diagrama a pseudocódigo	17	Terminado	Gabriel Guevara - Jordan Párraga	13	8	4			

Pruebas de conversión	8	Terminado	Gabriel Guevara - Jordan Párraga	8	8	8	4		
Retroalimentación	1	Terminado	Gabriel Guevara - Jordan Párraga	1	1	1	1	1	

Tabla 20 Sprint 3

Sprint	Inicio	Días	Jornada						
4	10/05/2017	6	4 hrs	J	V	L	M	X	J
				10 - may	11 - may	14 - may	15 - may	16 - may	17 - may
			Tareas Pendientes	8	8	8	7	6	5
			Horas Pendientes	95	84	80	76	72	71
Tarea	T. Est (hrs)	Estado	Responsable	Esfuerzo					
Desarrollo de métodos para mapeo de datos del pseudocódigo a diagrama de flujo	17	Terminado	Gabriel Guevara - Jordan Párraga	13	8	4			
Pruebas de conversión	8	Terminado	Gabriel Guevara - Jordan Párraga	8	8	8	4		

Retroalimentación	1	Terminado	Gabriel Guevara - Jordan Párraga	1	1	1	1	1	
-------------------	---	-----------	-------------------------------------	---	---	---	---	---	--

Tabla 21 Sprint 4

Sprint		Inicio	Días	Jornada																		
5		18/05/2017	4	4 hrs	V	L	M	X	J	V	L	M	X	J	V	L	M	X	J	V	L	
					18 - may	21 - may	22 - may	23 - may	24 - may	25 - may	28 - may	29 - may	30 - may	31 - may	01 - jun	04 - jun	05 - jun	06 - jun	07 - jun	08 - jun	11 - jun	
					Tareas Pendientes	5	5	5	5	5	5	4	4	4	4	4	4	3	3	1	1	0
					Horas Pendientes	7 1	6 1	5 7	5 3	4 9	4 5	4 1	3 7	3 3	2 9	2 5	2 1	1 7	1 2	8	4	0
Tarea	T. Est (hrs)	Estado	Responsable	Esfuerzo																		
Desarrollo motor ejecución del de	27	Terminado	Gabriel Guevara	2 3	1 9	1 5	1 1	7	3													

Desarrollo del intérprete de algoritmos	23	Terminado	Gabriel Guevara Jordan Párraga	-	2 3	2 3	2 3	2 3	2 3	2 3	2 1	1 7	1 3	9	5	1						
Pruebas de ejecución de algoritmos	13	Terminado	Gabriel Guevara Jordan Párraga	-	1 3	1 3	1 3	1 3	1 3	1 3	1 3	1 3	1 3	1 3	1 3	1 0	6	2				
Retroalimentación	5	Terminado	Gabriel Guevara Jordan Párraga	-													3					
Entrega del producto final	3	Terminado	Gabriel Guevara Jordan Párraga	-	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2	1	

Tabla 22 Sprint 5



3.3.6. Diseño e implementación

3.3.6.1. Diagrama Lógico

El diagrama lógico representa la separación por capas de la lógica de la aplicación, dividiéndose su estructura en tres capas: presentación, lógica de negocio y datos como se muestra en el siguiente gráfico.

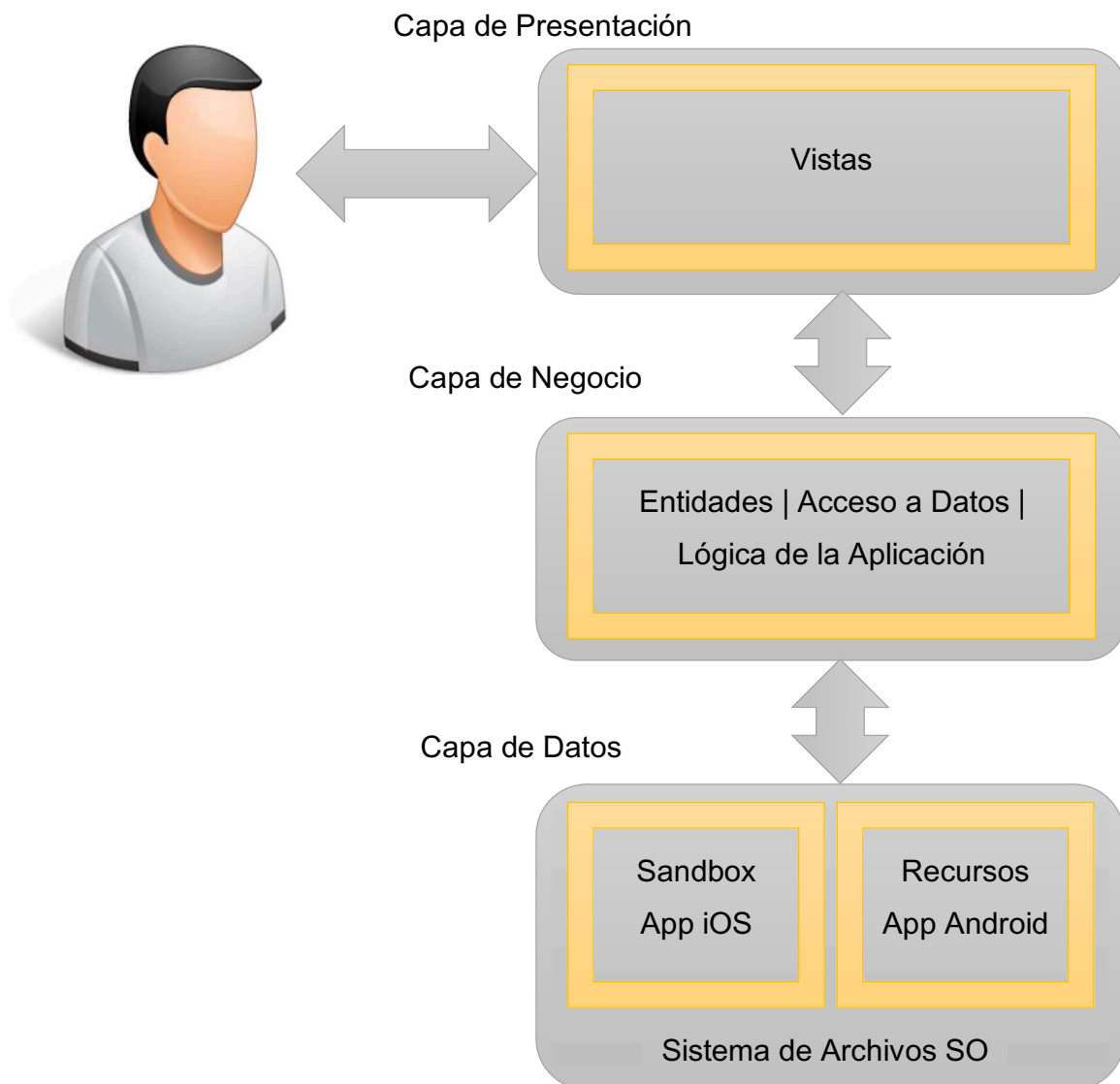


Ilustración 7 Diagrama Lógico de PseudoDF



3.3.6.2. Diagrama Físico

El diagrama físico representa un ambiente real o de producción en el cual es utilizado el aplicativo, mostrando los equipos físicos requeridos para el funcionamiento del mismo.

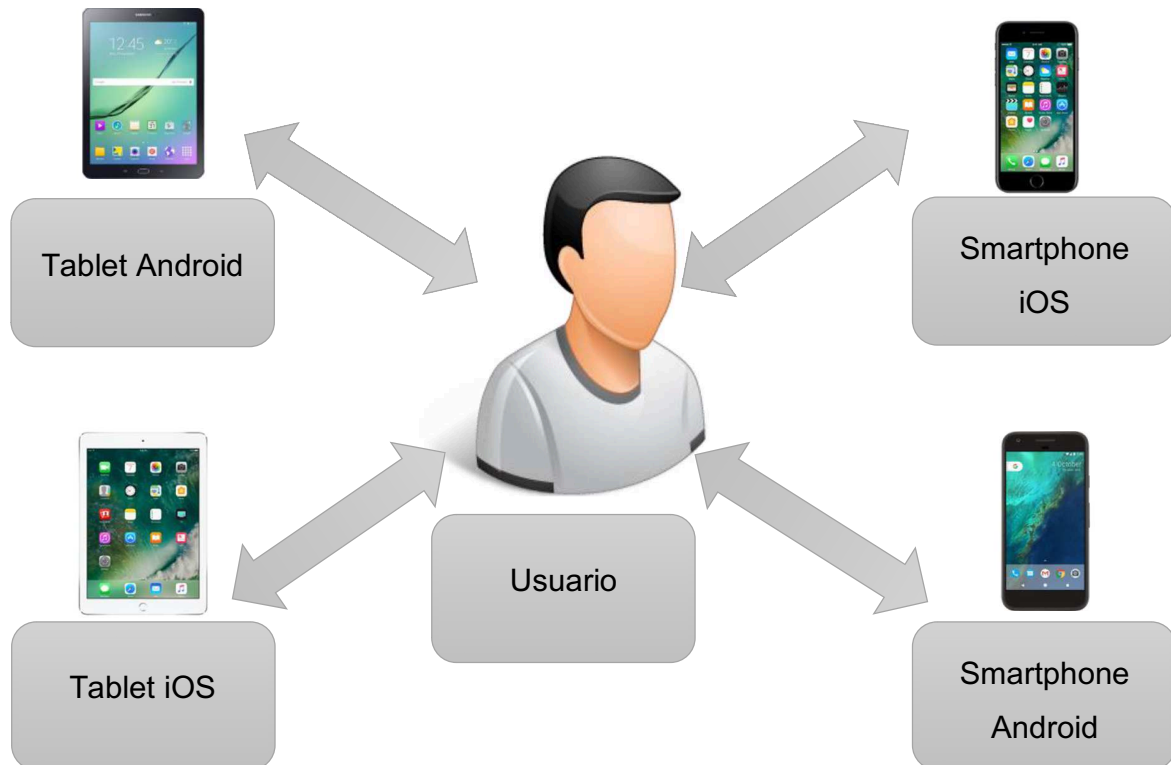


Ilustración 8 Diagrama Físico de PseudoDF



3.3.7. Entregables

3.3.7.1 Modelos de Caso de Uso

Modelo General PseudoDF

Representa una visión general del aplicativo móvil, el cual consta únicamente del usuario del Smartphone, el cual tendrá acceso a: el editor de diagramas de flujo y pseudocódigo para diseñar algoritmos, funcionalidad para conversión de diagrama a pseudocódigo y viceversa, y por último tendrá la posibilidad de ejecutar los algoritmos.

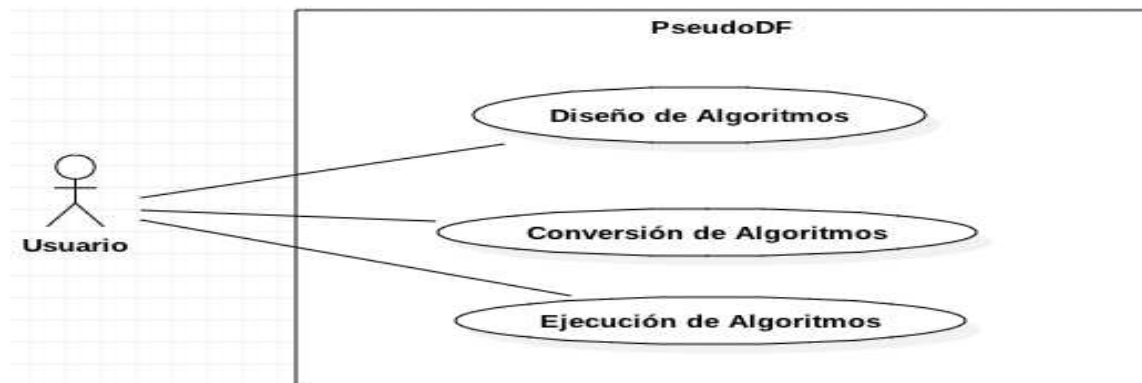


Ilustración 9 Diagrama de Caso de Usos Modelo General PseudoDF



Diseño de Algoritmos

El usuario podrá crear nuevo diagrama de flujo o nuevos pseudocódigos, además tendrá la opción de abrir algoritmos previamente creados y continuar su edición, por último, podrá eliminar algoritmos guardados si así lo desea.

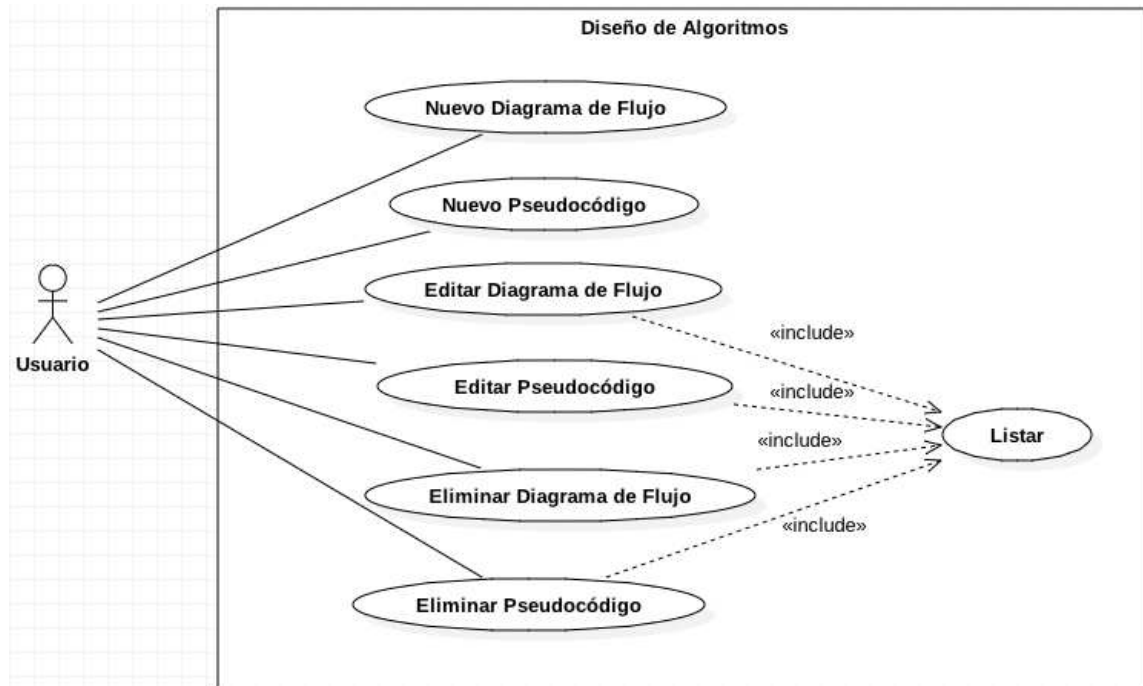


Ilustración 10 Diagrama de Caso de Usos Diseño de Algoritmos



Conversión de Algoritmos

El usuario podrá cambiar entre las vistas de edición de diagrama y edición de pseudocódigo, con lo cual se realizará la conversión del algoritmo entre la técnica de representación activa a su equivalente seleccionada.

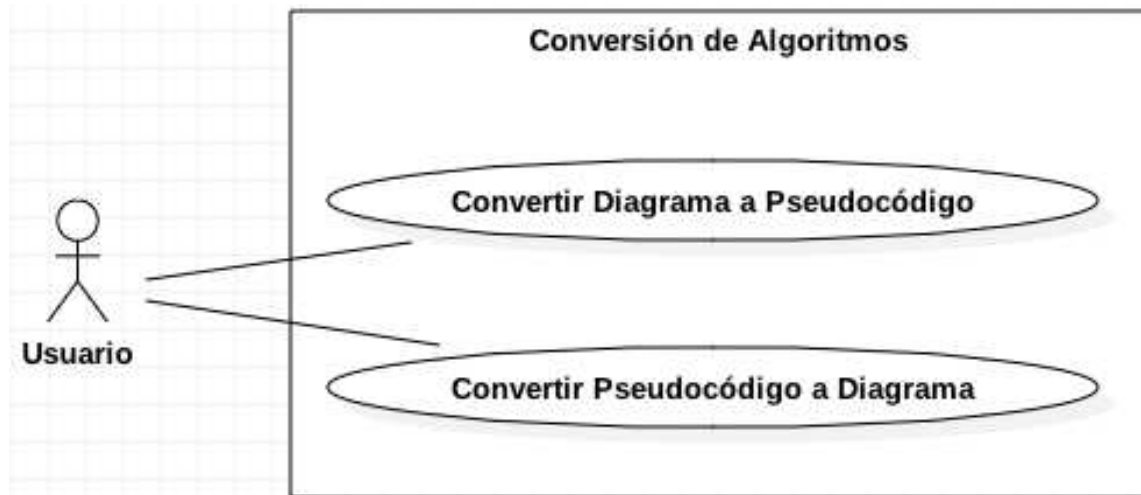


Ilustración 11 Diagrama de Caso de Usos Conversión de Algoritmos



Ejecución de Algoritmos

El usuario podrá ejecutar el algoritmo desde la vista de edición de diagrama de flujo o edición de pseudocódigo.

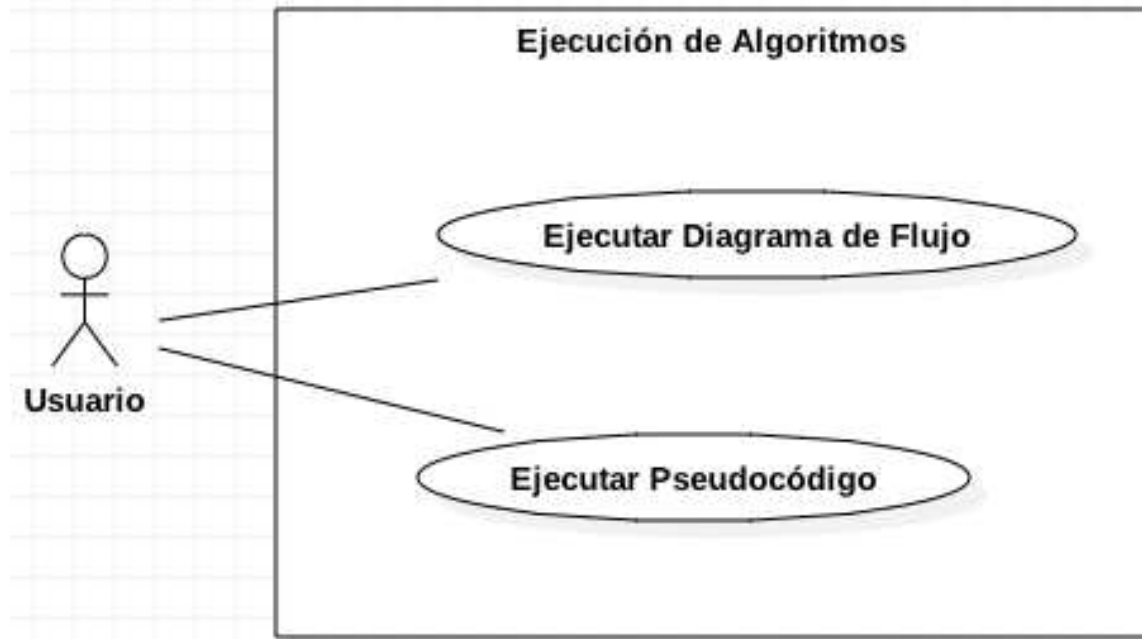


Ilustración 12 Diagrama de Caso de Usos Conversión de Algoritmos Ejecución de Algoritmos



3.3.7.2 Interfaces

A continuación, se presentan las interfaces de usuario que permiten la interacción con la aplicación PseudoDF.

Diagramación de Flujos

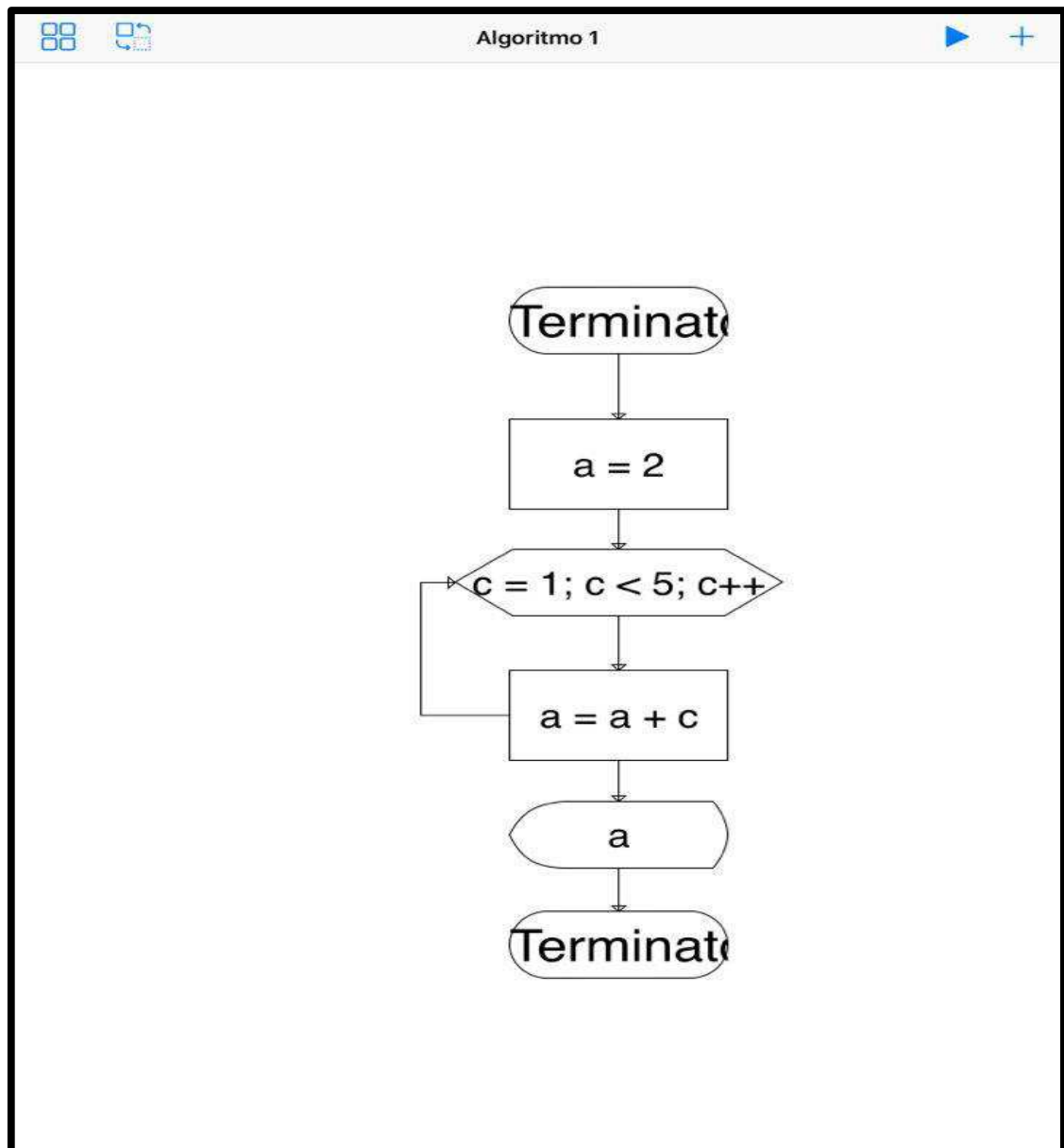


Ilustración 13 Diagramación de Algoritmos-Espacio de Trabajo iOS

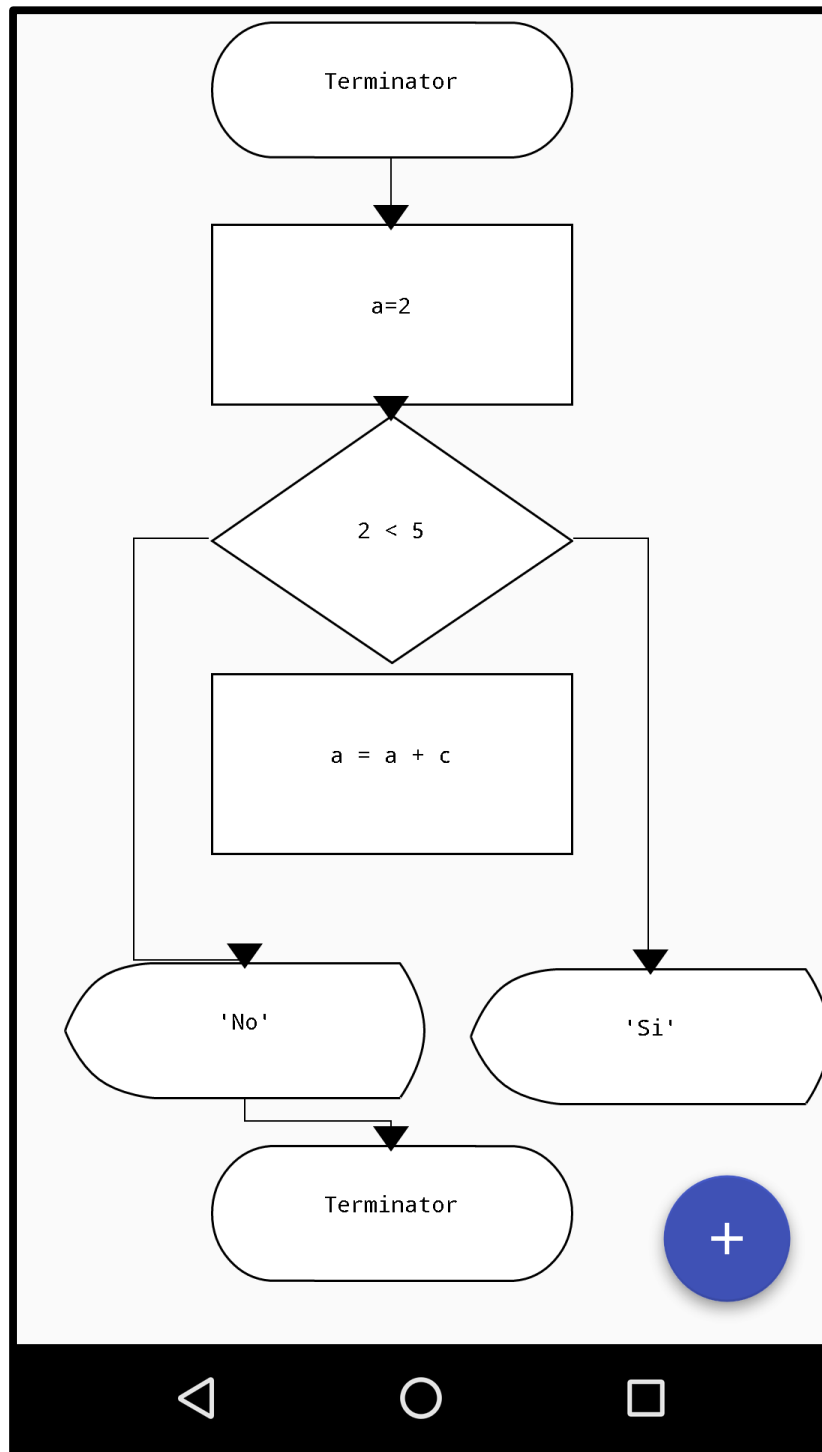


Ilustración 14 Diagramación de Algoritmos-Espacio de Trabajo Android

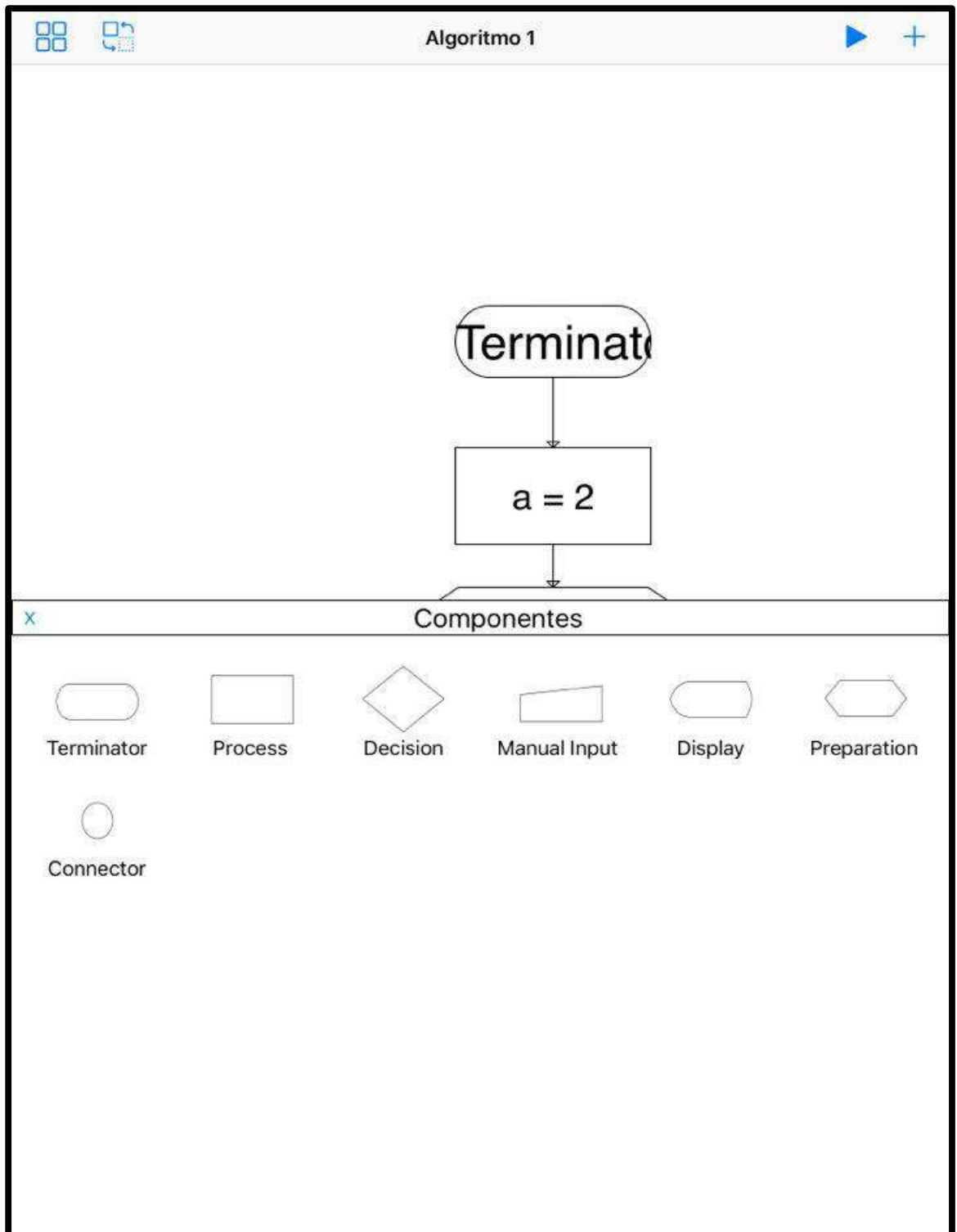


Ilustración 15 Diagramación de Algoritmos-Paleta de Componentes iOS

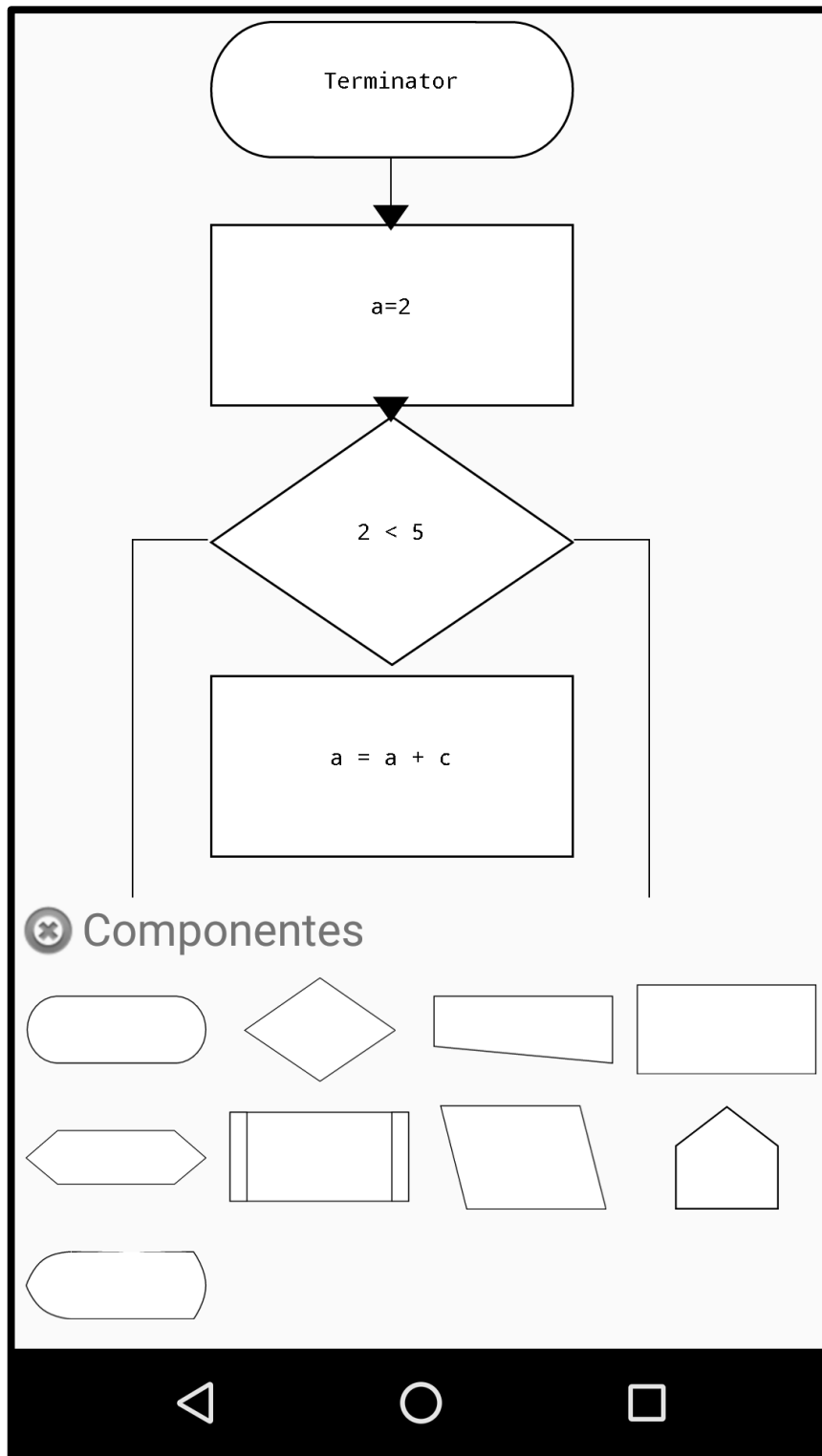


Ilustración 16 Diagramación de Algoritmos-Paleta de Componentes Android

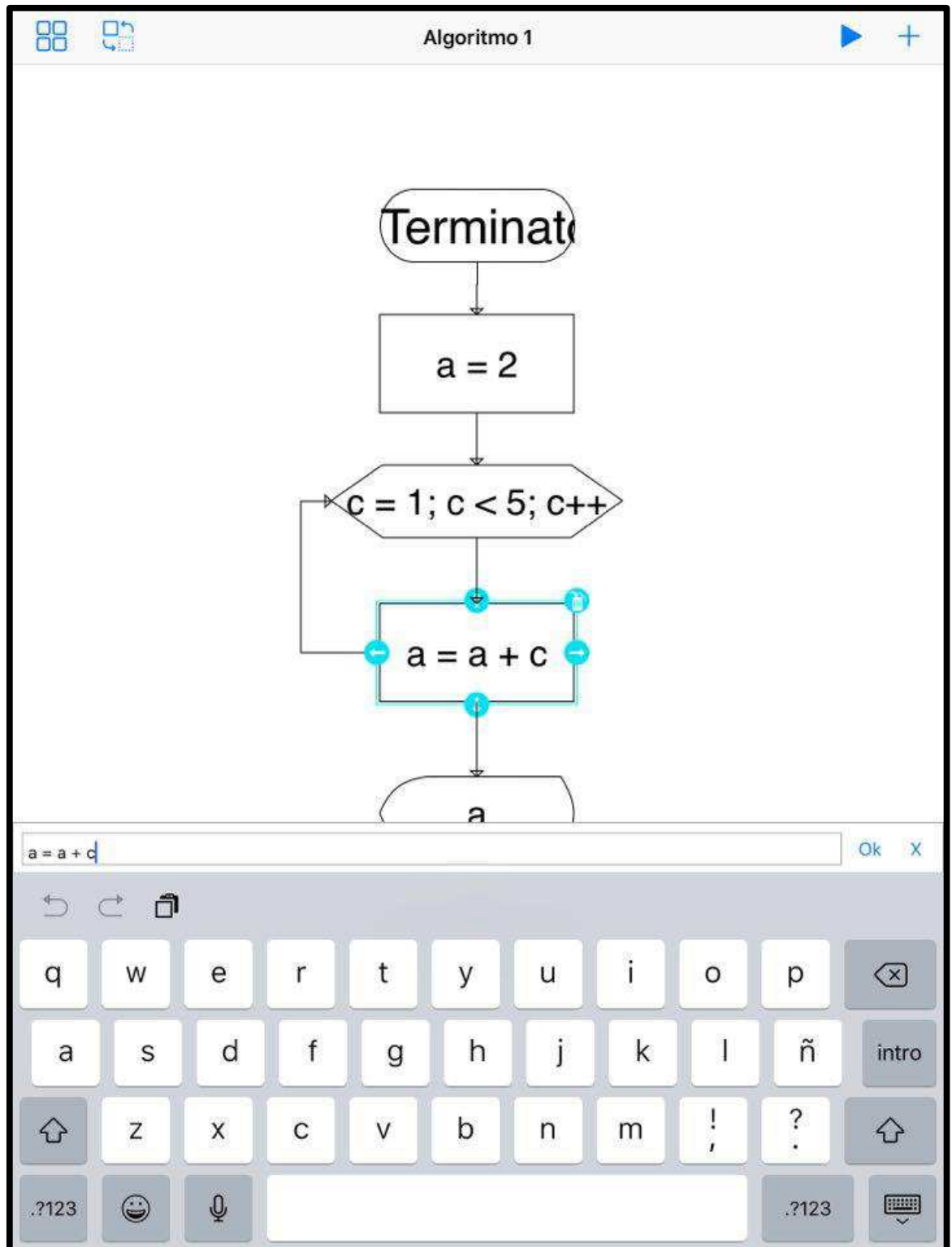




Ilustración 17 Diagramación de Algoritmos-Edición de Sentencias iOS

Editor de Pseudocódigo

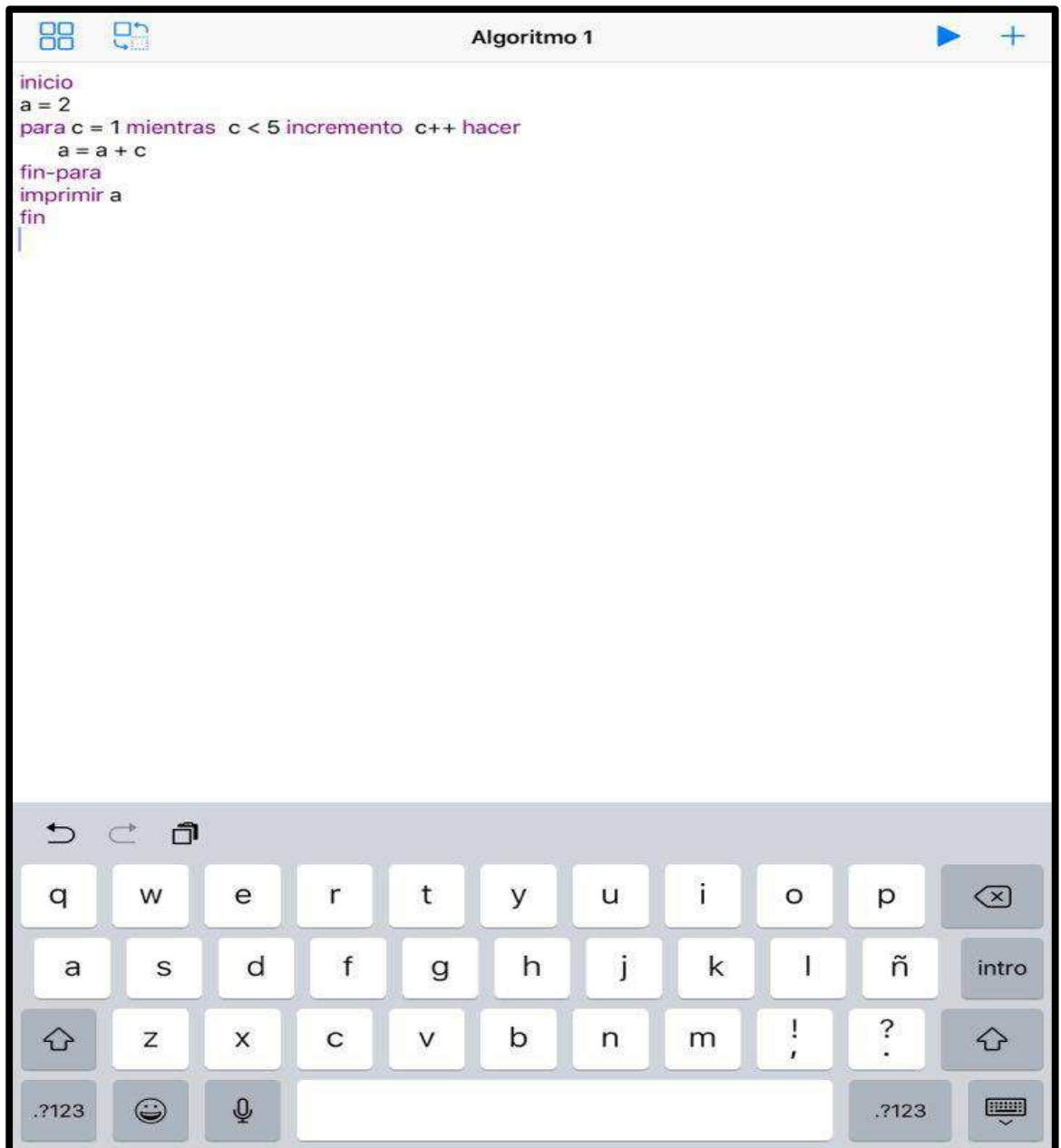


Ilustración 18 Editor de Pseudocódigo-Espacio de Trabajo iOS



Ejecución de Algoritmos

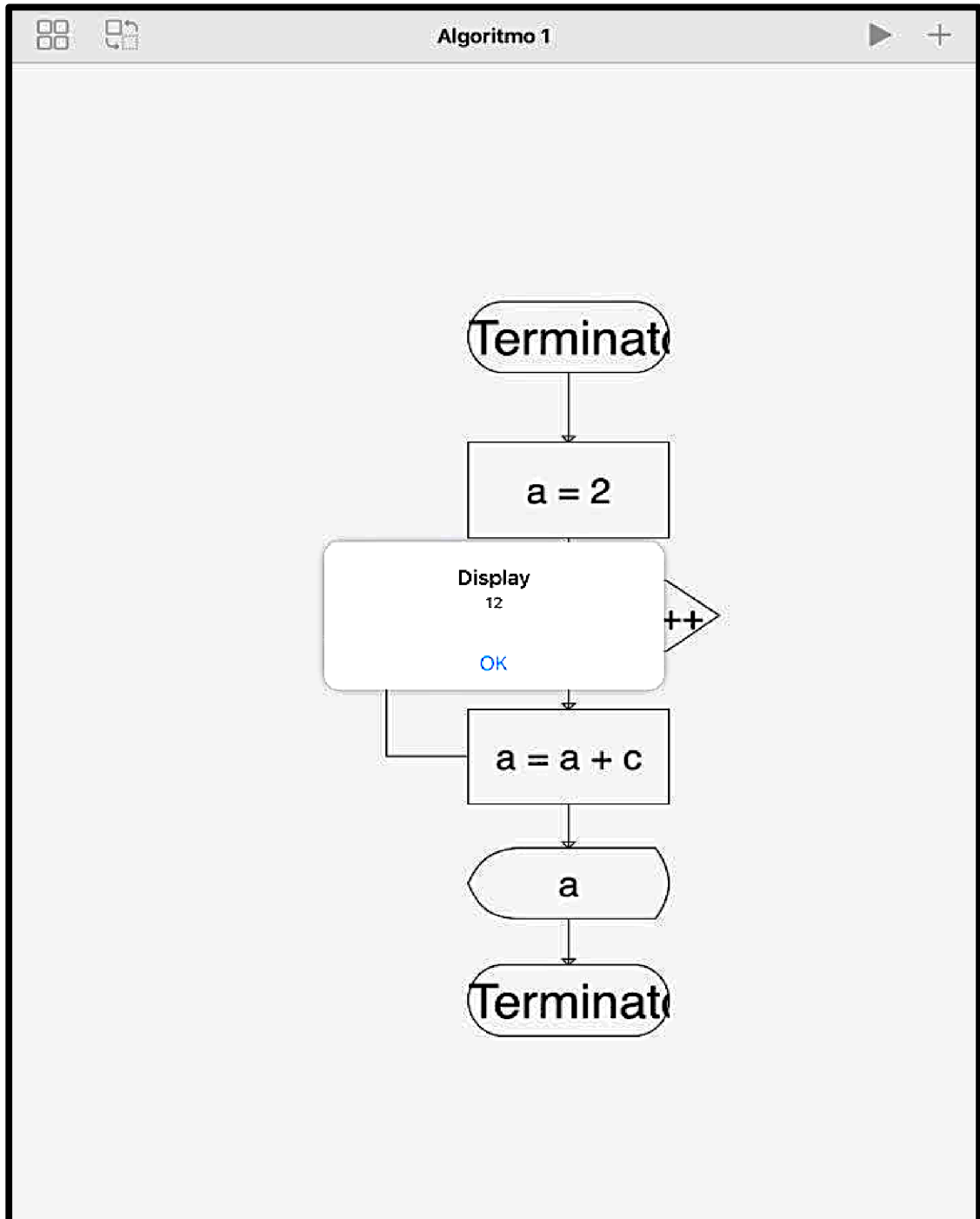


Ilustración 19 Ejecución de Algoritmos-Ejecución en Vista de Diagramación iOS

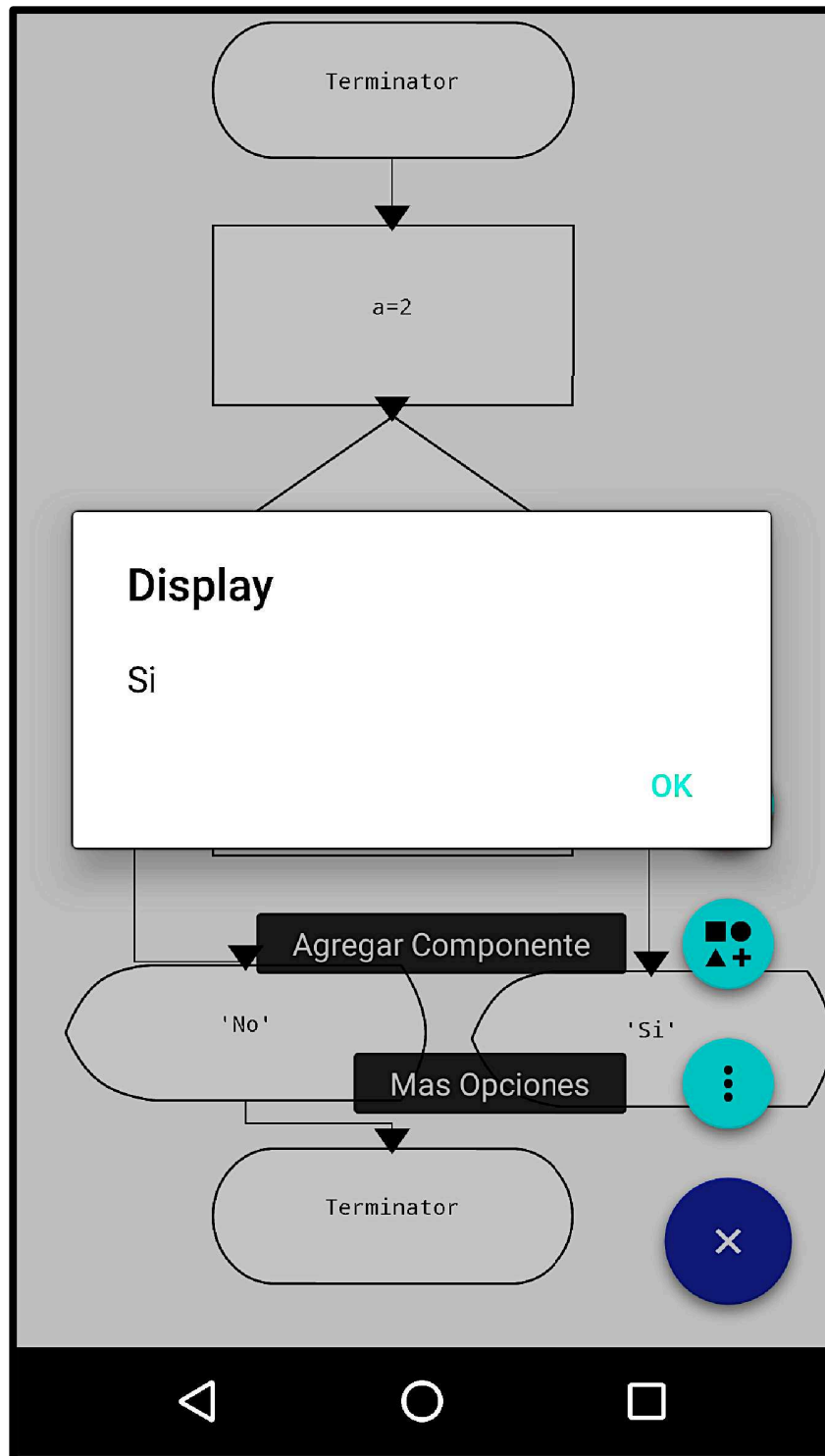


Ilustración 20 Ejecución de Algoritmos-Ejecución en Vista de Diagramación Android

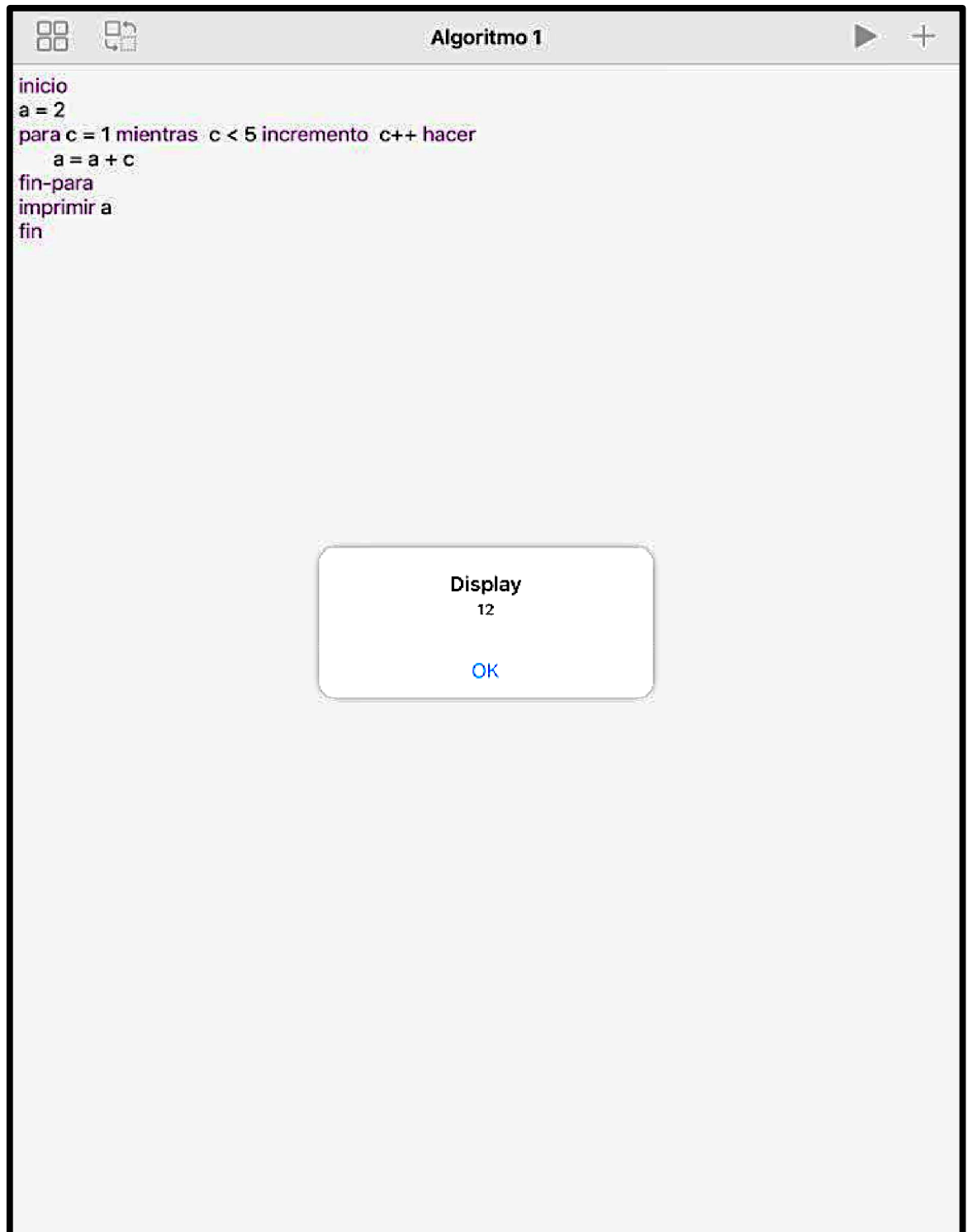


Ilustración 21 Ejecución de Algoritmos-Ejecución en Vista de Pseudocódigo iOS

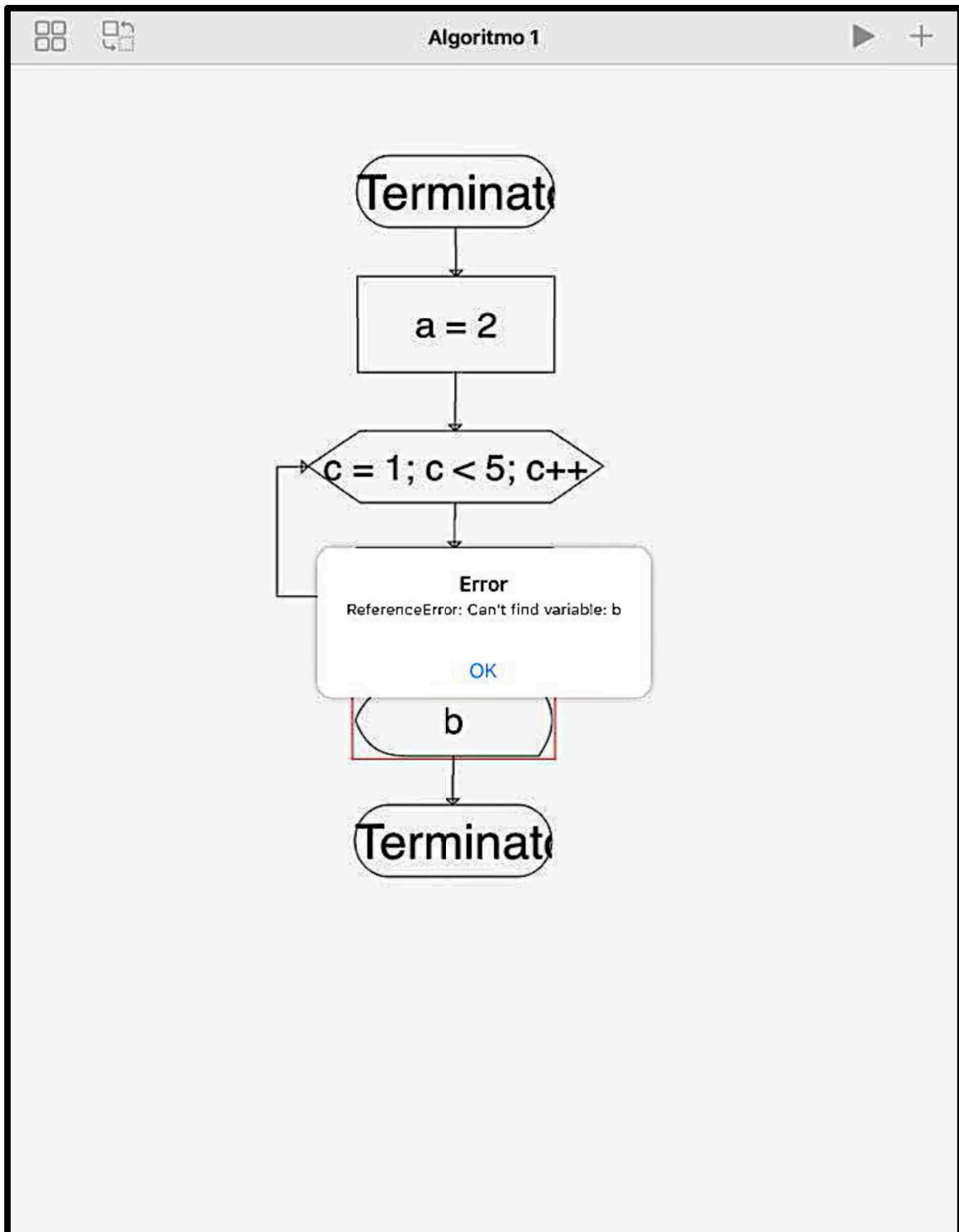


Ilustración 22 Ejecución de Algoritmos-Alerta de Errores iOS

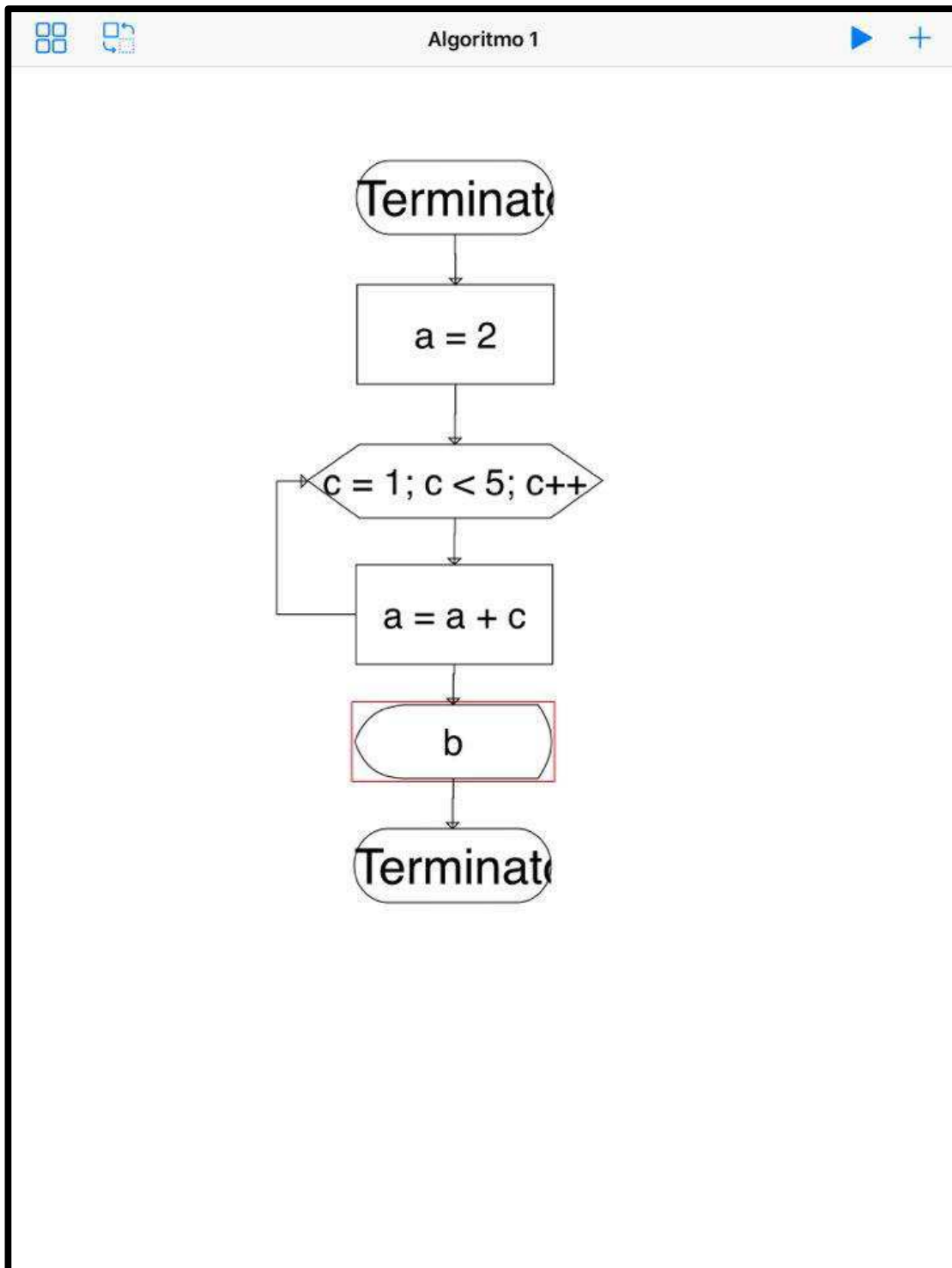


Ilustración 23 Ejecución de Algoritmos-Marcador de Error iOS

3.3.8. Codificación y Testing

3.3.8.1 Clases

Para el desarrollo de las aplicaciones se utilizó el patrón MVC comúnmente implementado en plataformas móviles, separando la definición de las vistas, las clases controladoras y las clases de acceso a datos, interactuando la segunda como intermediaria entre los eventos de la vista hace el acceso a datos. Además, se implementaron clases específicas para el manejo de los componentes gráficos del diagrama por capas de dibujo.

Carpeta	Clase Swift	Clase Java	Descripción
Raíz	ViewController.swift	ViewCanvas.java	Encargada de manejar los eventos en el espacio de diagramación, agregando, editando, eliminando y moviendo figuras, además de conectarlas entre si para lograr representar un flujo de datos.
	CustomPaletteViewController.swift		Maneja la acción de cerrado de la paleta de componentes e inyecta el canvas sobre el que se agregaran las figuras al CollectionController de la paleta.

	PseudocodeEditorViewController.swift		Se encarga de manejar los eventos de modificación del texto del editor de pseudocódigo, destacando las palabras reservadas con un color diferente.
	DiagramPseudocodeConverter.swift		Permite convertir el algoritmo representado en el editor de diagramas de flujo a su equivalente pseudocódigo.
	DesignerChooserViewController.swift		Mantiene instancias del editor de pseudocódigo y el editor de diagramas de flujo, permitiendo realizar el cambio entre ambas vistas, ejecutando la conversión entre ambas técnicas de diagramación.
	PseudocodeDiagramConverter.swift		Permite convertir el algoritmo representado en el editor de pseudocódigo a su equivalente diagrama de flujo.
	LaunchScreen.storyboard		Contiene el View a mostrarse como SplashScreen al iniciar la aplicación, el cual contiene el logo de la aplicación.

	Main.storyboard		Contiene los Views correspondientes al editor de pseudocódigo y de diagramas de flujos, junto con otras vistas como la paleta de componentes y el contenedor principal de los editores.
Diagram Compone nts	DiagramShape.swift	ComponenteDrawable.java	Capa gráfica que representa un componente de un diagrama de flujo y mantiene los metadatos, como pueden ser, coordenadas y conectores.
	OptionsShape.swift	OptionsComponenteDrawable.java	Capa gráfica que representa las opciones de un componente de diagrama de flujo y mantiene los metadatos, como pueden ser, coordenadas y opciones disponibles.
	Shape.swift		Carga archivos de imágenes svg para ser dibujados en un DiagramShape, obteniendo como resultado una representación gráfica de un componente de diagrama.
	TextShape.swift	TextDrawable.java	Capa de texto que contiene la sentencia a ejecutar dentro de un componente del diagrama de flujo.

	ConnectionLine.swift	LineDrawable.java	Capa gráfica que representa un conector en forma de línea entre dos componentes del diagrama de flujo.
	OptionsConnector.swift	OptionsLineDrawable.java	Capa gráfica que representa las opciones de un conector del diagrama de flujo y mantiene los metadatos, como pueden ser, coordenadas y opciones disponibles.
Diagram Executors	ExecutionContext.swift	ExecutionContext.java	Contiene un motor de javascript capaz de ejecutar las sentencias escritas en un componente de diagrama, iniciando el proceso a partir del elemento inicio y continuando de forma secuencial, además controla las excepciones que puedan ocurrir.
	Executor.swift	Executo.java	Clase que contiene el comportamiento base para que cada componente pueda ejecutar sus sentencias utilizando el ExecutionContext y continuar al siguiente paso de forma secuencial.
	ProcessExecutor.swift	ProcessExecutor.java	Contiene el comportamiento de ejecución de un componente de diagrama del tipo Proceso.

	ConditionExecutor.swift	ConditionExecutor.java	Contiene el comportamiento de ejecución de un componente de diagrama del tipo Decisión.
	TerminatorExecutor.swift	TerminatorExecutor.java	Contiene el comportamiento de ejecución de un componente de diagrama del tipo Inicio / Fin.
	DataExecutor.swift	DataExecutor.java	Contiene el comportamiento de ejecución de un componente de diagrama del tipo Datos.
	ManualInputExecutor.swift	ManualInputExecutor.java	Contiene el comportamiento de ejecución de un componente de diagrama del tipo Entrada Manual.
	DisplayExecutor.swift	DisplayExecutor.java	Contiene el comportamiento de ejecución de un componente de diagrama del tipo Mostrar.
	PreparationExecutor.swift	PreparationExecutor.java	Contiene el comportamiento de ejecución de un componente de diagrama del tipo Para (bucle).
	ConnectorExecutor.swift		Contiene el comportamiento de ejecución de un componente de diagrama del tipo Conector dentro de una misma página.

UICompo nents/Pal ette	ComponentsPaletteCol lectionController.swift		Maneja el comportamiento y las subvistas de la paleta de componentes.
	ComponentsPalletColl ectionCell.swift		Maneja el comportamiento y las subvistas de las celdas de la paleta de componentes.
	ComponentsPalletHe aderCollectionReusabl e.swift		Maneja el comportamiento y las subvistas de la cabecera de la paleta de componentes.
Helpers	FilesHelpers.swift		Recupera las imágenes definidas en el archivo de configuración Info.plist, dentro del diccionario PaletteComponents.
	AlertsManager.swift	AlertsManager.java	Permite mostrar cuadros de alerta para ingreso de datos, mostrar información o errores en un ViewController específico y permitiendo retomar el proceso de ejecución luego de ocultar la alerta.
	EditorProtocol.swift		Define las operaciones básicas que deben implementar los editores de algoritmos.

Factorie	ExecutorFactory.swift	ExecutorFactory.java	Crea y recupera una instancia de un Executor específico para un componente de diagrama dependiendo del tipo solicitado y para un contexto o proceso específico. Adicionalmente contiene los tipos de ejecutores y palabras reservadas como enumeradores.
-----------------	-----------------------	----------------------	--

Tabla 23 Clases Utilizadas en el Desarrollo de los Aplicativos y su Respectiva Funcionalidad.

3.3.8.2 Pruebas

Con la finalidad de verificar el correcto funcionamiento de los aplicativos móviles se ha realizado una lista de las acciones realizables en cada módulo, en conjunto con el resultado esperado para cada acción y validando lo establecido con los resultados de los test realizados en los aplicativos móviles, permitiendo aplicar las respectivas correcciones de errores.

Módulo	Casos de Prueba	Resultado esperado	Resultado	Aprobado
Diagramación de Flujo de Datos	Agregar figura desde la paleta de componentes.	<ul style="list-style-type: none">Al mostrar la paleta de componentes debe poder realizarse drag and drop de las figuras disponibles en la paleta hacia el espacio de trabajo, ocultando la paleta al seleccionar una figura y volviendo a mostrarse después de haber terminado la acción.	El tamaño de la paleta, no permite visualizar los componentes con facilidad.	×

			La paleta, permite visualizar correctamente los componentes, pero no permite arrastrar los componentes fuera de ella.	✘
			La paleta permite arrastrar componentes al espacio de trabajo sin embargo no se oculta mientras el evento drag está activado.	✘
			La paleta de componentes permite arrastrar elementos al espacio de trabajo, ocultándose cuando el evento drag está activo y mostrándose cuando termina la acción.	✓
Selección de componente.		<ul style="list-style-type: none"> Al realizar un toque en pantalla sobre un componente del diagrama, deberá seleccionarse mostrando un cuadro de opciones alrededor de él, 	El cuadro de opciones no aparece alrededor del componente cuando ha sido desplazado de su posición inicial.	✘
			El cuadro de opciones se muestra sobre el componente al dar un toque sobre él en la	✓

		mostrando las opciones de conexión y eliminación.	ubicación de pantalla que se encuentre.	
	Deseleccionar componente	<ul style="list-style-type: none"> • Al estar seleccionado un componente y realizar un toque en el espacio de trabajo donde no se encuentra parte del elemento, este deberá ocultar su cuadro de opciones. • En el caso de que el toque en pantalla coincida con otro componente deberá de ocultarse el cuadro de opciones de la selección anterior y mostrar el cuadro de la nueva selección. 	Al presionar sobre otro componente cuando hay un elemento seleccionado previamente, se muestra el cuadro de opciones de ambos componentes.	x
			Al presionar sobre un componente cuando hay otro seleccionado, se deselectiona el anterior ocultando su cuadro de dialogo y mostrando el de la nueva selección.	✓

Eliminación de componente en diagrama.	<ul style="list-style-type: none"> Al presionar la opción de eliminación mostrada en el cuadro de opciones de un componente seleccionado, este deberá ser removido del espacio de trabajo junto con las líneas de conexión que estén asociadas a él. 	El componente es eliminado al presionar la opción correspondiente, sin embargo, las líneas asociadas se mantienen en el espacio de trabajo.	✘
		El componente es eliminado al presionar la opción correspondiente, junto con las líneas asociadas a él.	✓
Conectar componentes	<ul style="list-style-type: none"> Al presionar y arrastrar desde una de las opciones de conexión del cuadro de opciones de un componente seleccionado deberá graficarse una línea de conexión desde el origen del conector, al punto de destino actual del toque en pantalla. 	Las líneas de conexión cruzan a través de los componentes involucrados en la asociación.	✘
		La línea de conexión no se redibuja al ser movido alguno de los componentes involucrados.	✘
		Los conectores de los componentes permiten conexiones a múltiples líneas.	✘
		Los conectores solo aceptan asociación con una línea, las cuales al conectar dos	✓

		<ul style="list-style-type: none"> • De terminar el evento sobre un conector de otra figura deberá asociarse la línea a la figura de origen y de partida, redibujándose de la forma correcta. • De terminar el evento sobre el mismo componente, en un espacio vacío o en un conector de destino previamente utilizado, la línea deberá desaparecer. 	componentes cruzan alrededor de ellos y se redibuja su recorrido al mover alguno de los elementos asociados.	
	Mover componente	<ul style="list-style-type: none"> • Al iniciar un toque en pantalla sobre un componente y realizar un arrastre por pantalla, el componente, deberá 	El componente arrastrado no avanza a la misma velocidad que el evento drag y termina en una posición diferente.	✘
			Las líneas no se redibujan al mover el componente, quedando en la misma posición que tenían.	✘

		<p>moverse a la posición actual del evento.</p> <ul style="list-style-type: none"> Las líneas de conexión asociadas al componente desplazado, deberán redibujar para adaptarse a la nueva posición. 	<p>El componente se desplaza en conjunto con la posición del evento de arrastre y las líneas asociadas se redibujan en función de las ubicaciones del desplazamiento y la posición final.</p>	✓
Edición de Pseudocódigo	Resaltar las palabras claves del pseudocódigo.	<ul style="list-style-type: none"> Al escribir palabras dentro del editor de pseudocódigo se deberá resaltar con un color diferente las palabras que sean reconocidas como reservadas. 	<p>Cuando se escribe una palabra reservada que contiene otra, no se resalta, por ejemplo: si (palabra reservada), fin-si (palabra reservada compuesta), esta última no resalta el contenido a partir del '-'.</p>	✗
			<p>Las palabras no reservadas que contienen palabras reservadas, resaltan el fragmento que coincide, por ejemplo: si (palabra reservada), simple (palabra no</p>	✗

			reservada), esta última resalta los dos primeros caracteres.	
			Las palabras reservadas se resaltan de un color diferente, no se resaltan cuando son encontradas como parte de otra palabra.	✓
	Tabulación automática.	<ul style="list-style-type: none"> Al presionar la tecla de retorno para una nueva línea se deberá agregar un espacio de tabulación cuando la línea de procedencia indique la apertura de una estructura de control, y deberá mantenerse hasta finalizar la estructura. 	Las tabulaciones no se acumulan dentro de las estructuras de control anidadas.	✘
			Al cerrar las estructuras de control, se mantiene el nivel de tabulación acumulado.	✘
			Se incrementa el nivel de tabulación al abrir una estructura de control y disminuye al cerrarla.	✓

Conversión de Diagrama a Pseudocódigo	Creación de palabras claves según los tipos de componentes utilizados en el diagrama.	<ul style="list-style-type: none"> Al cambiar de la vista de diagrama a pseudocódigo, se deberá crear una nueva línea con la palabra clave equivalente al tipo de componente de diagrama de cada paso del flujo. 	El pseudocódigo traducido desde el diagrama aparece por duplicado.	✘
			Se crea una palabra clave equivalente por cada componente del diagrama de flujo.	✓
	Creación de sentencia en forma secuencial.	<ul style="list-style-type: none"> El orden de creación de las sentencias en el editor de pseudocódigo, será definido por la secuencia original del flujo. 	Al utilizar estructuras de control anidadas las sentencias de pseudocódigo no siguen el orden correcto.	✘
			Las sentencias de pseudocódigo siguen la secuencia definida por las líneas de conexión del diagrama de flujo.	✓
			El nivel de tabulación no coincide al existir varias estructuras de	✘

	Tabulación de estructuras de control.	<ul style="list-style-type: none"> El contenido de las estructuras de control traducidas desde el diagrama deberá incluir el nivel de tabulación correspondiente según el número de estructuras anidadas. 	control anidadas con caminos condicionados.	
			El nivel de tabulación incrementa a medida que se ingresa a estructuras de control anidadas	✓
Conversión de Pseudocódigo a Diagrama	Creación de tipos de componentes según las palabras claves utilizadas en el pseudocódigo.	<ul style="list-style-type: none"> Al cambiar de la vista de pseudocódigo a diagrama, se deberá crear un nuevo componente con la figura equivalente a la palabra clave de cada paso del pseudocódigo. 	El contenido de los componentes creados para el diagrama incluye la palabra reservada.	✗
			Se crea un componente equivalente por cada palabra reservada, el contenido de la sentencia excluye las palabras reservadas y se traducen cuando sea necesario.	✓

	Ordenar componentes del flujo en forma secuencial.	<ul style="list-style-type: none"> El orden de los componentes de diagrama y la secuencia de las conexiones deberá coincidir y seguir la secuencia de líneas (pasos) definida en el pseudocódigo. 	Existen conectores que constan solo de un componente de origen pero no de destino al usar estructuras de control anidadas.	✘
			Las líneas del pseudocódigo, siguen el orden del diagrama de flujo representado por los conectores.	✓
	Anidar estructuras de control	<ul style="list-style-type: none"> Se deberán conectar en forma anidada los componentes de diagrama de estructuras de control coincidiendo con las anidaciones definidas en el pseudocódigo. 	La estructura de control if no contiene componentes en el camino 'NO', ya que son agregados al finalizar el 'SI'.	✘
			El paso final del bucle for, no regresa al inicio del bucle.	✘
			Las estructuras de control siguen la secuencia especificada en el pseudocódigo, creando múltiples	✓

			caminos para las condiciones y uniendo el paso final del bucle al inicio del mismo.		
Ejecución de Algoritmos	Ejecutar algoritmo siguiendo secuencia predefinida.	el la	<ul style="list-style-type: none"> Los pasos del algoritmo se deberán ejecutar, tomando como referencia el orden o secuencia en que se encuentran conectados los componentes del diagrama, tomando como punto de partida el componente inicio/fin que contenga una conexión saliente. 	El motor de ejecución no distingue el componente inicial entre los elementos del diagrama.	✘
				El camino 'No' de una estructura condicional no es ejecutado.	✘
				El bucle for reinicia el contador al regresar al primer paso, resultando en un bucle infinito.	✘
				El motor de ejecución corre el algoritmo partiendo del componente inicio / fin que tiene un conector saliente, ejecutando de manera secuencial los demás pasos por medio de las conexión	✓

			que poseen y aplicando el comportamiento de las estructuras de control.	
Ejecutar las sentencias según la palabra clave o componente de diagrama utilizado.	<ul style="list-style-type: none"> El comportamiento o forma en que se ejecutará la sentencia escrita en un componente, dependerá del tipo de componente utilizado, realizando operaciones matemáticas, repeticiones de pasos (bucles), controlando el camino del flujo con condiciones y mostrando mensajes de lectura e ingreso de datos. 	El resultado de las condiciones siempre es el mismo, conllevando a seguir el camino 'SI' en cada ejecución.	×	
		El bucle for no reconoce el operador incremento, ni decremento.	×	
		Las alertas de entrada y salida no se muestran al cambiar la vista de edición.	×	
		El motor de ejecución interpreta las sentencias dentro de un paso del diagrama o pseudocódigo y ejecuta un comportamiento específico en función del tipo de	✓	

			paso actual, incluyendo las estructuras de control y mostrando alertas en la vista actual para el ingreso y salida de datos.	
	Mostrar mensaje de error cuando ocurra una excepción.	<ul style="list-style-type: none"> Al producirse una excepción en la ejecución, deberá detenerse el proceso, y mostrar una alerta con la excepción producida. 	Tras ocurrir una excepción en un paso determinado el motor de ejecución continuaba con los demás pasos en la secuencia.	✘
			El motor de ejecución detiene el proceso tras una excepción y muestra una alerta con el mensaje que indica el error ocurrido..	✓
		<ul style="list-style-type: none"> Al producirse una excepción, deberá 	El paso donde ocurre la excepción es identificado por el motor de ejecución, sin embargo	✘

	Marcar el pasó donde ocurrió una excepción.	señalarse el paso donde ocurrió el error, con un marco rojo alrededor del componente de diagrama	no muestra el marco rojo alrededor.	
			Se muestra el marco alrededor del paso donde ocurrió la excepción, sin embargo el marco no desaparece al presionar algún punto del espacio de trabajo o al volver a ejecutar el algoritmo.	✘
			Se muestra un marco rojo alrededor del paso donde ocurrió la excepción y desaparece al presionar algún punto del espacio de trabajo o al volver a ejecutar el algoritmo.	✓

Tabla 24 Lista de Pruebas Realizadas a los Aplicativos Móviles



3.3.9. Producto Implementado

El desarrollo de las aplicaciones móviles se llevó a cabo mediante el uso de la metodología SCRUM, permitiendo seccionar el desarrollo de la aplicación en una serie de entregables al agrupar las funcionalidades extraídas de los requerimientos resultantes de la investigación.

Al concluir el levantamiento de requerimientos y su respectivo análisis, el desarrollo de los aplicativos se dividió en 5 partes según la funcionalidad: Diagramación de Flujos de Datos, Edición de Pseudocódigo, Conversión de Diagrama a Pseudocódigo, Conversión de Pseudocódigo a Diagrama y Ejecución de Algoritmos.

Las aplicaciones fueron desarrolladas en 5 sprints permitiendo verificar y testear los entregables al termino de cada iteración, resultando en la subida y actualización en las tiendas de aplicaciones respectivas para los sistemas operativos iOS y Android.

3.3.9.1 Manual del Producto

En concordancia a la última versión liberada al concluir la presente investigación, se elaboró el respectivo manual de usuario para el apoyo y orientación de los beneficiarios en la utilización de los aplicativos móviles para las plataformas iOS y Android (Ver anexo 4)

CAPÍTULO IV

EVALUACIÓN DE RESULTADOS



CAPÍTULO IV

EVALUACIÓN DE RESULTADOS

4.1 Introducción

Las aplicaciones móviles, han sido desplegadas en el App Store y Google Play de iOS y Android respectivamente, a través de este medio es posible descargar e instalar las aplicaciones en dispositivos móviles que posean la versión 8.0 o superior del sistema iOS y 4.2 o superior de Android, accediendo a la funcionalidad desarrollada como producto de esta investigación.

El uso de las aplicaciones por medio de distintos dispositivos móviles iOS y Android, permitió verificar el funcionamiento del producto desarrollado en relación a los requerimientos planteados, experiencia de usuario y estabilidad en las diversas versiones de los sistemas operativos para los cuales fueron desarrolladas las aplicaciones.

4.2 Seguimiento y Monitoreo de resultados

4.2.1 Descripción de los resultados

Los requerimientos funcionales y no funcionales, producto de la investigación previa, en conjunto con la aplicación de la metodología Scrum durante la fase de desarrollo, conllevaron a la realización de cinco Sprints. Cada Sprint resultó en subproductos funcionales correspondientes a la agrupación de funcionalidades como son: Diagramación de Flujos de Datos, Edición de Pseudocódigo, Conversión de Diagrama a Pseudocódigo, Conversión de Pseudocódigo a Diagrama y Ejecución de Algoritmos.



Para la comprobación de los resultados, las aplicaciones fueron desplegadas en un iPad Air 2 y un Motorola Z Play con las siguientes características:

CARACTERÍSTICAS iPad AIR 2	
PANTALLA	IPS LED – 9,7 Pulgadas
RESOLUCIÓN	2048x1536 píxeles (264 ppp)
PROCESADOR	A8X (64 bits, 3 núcleos) / Procesador M8
RAM	2 GB
MEMORIA	128 GB
VERSION SO	iOS 10.3.2

Tabla 25 Características iPad utilizado para las pruebas

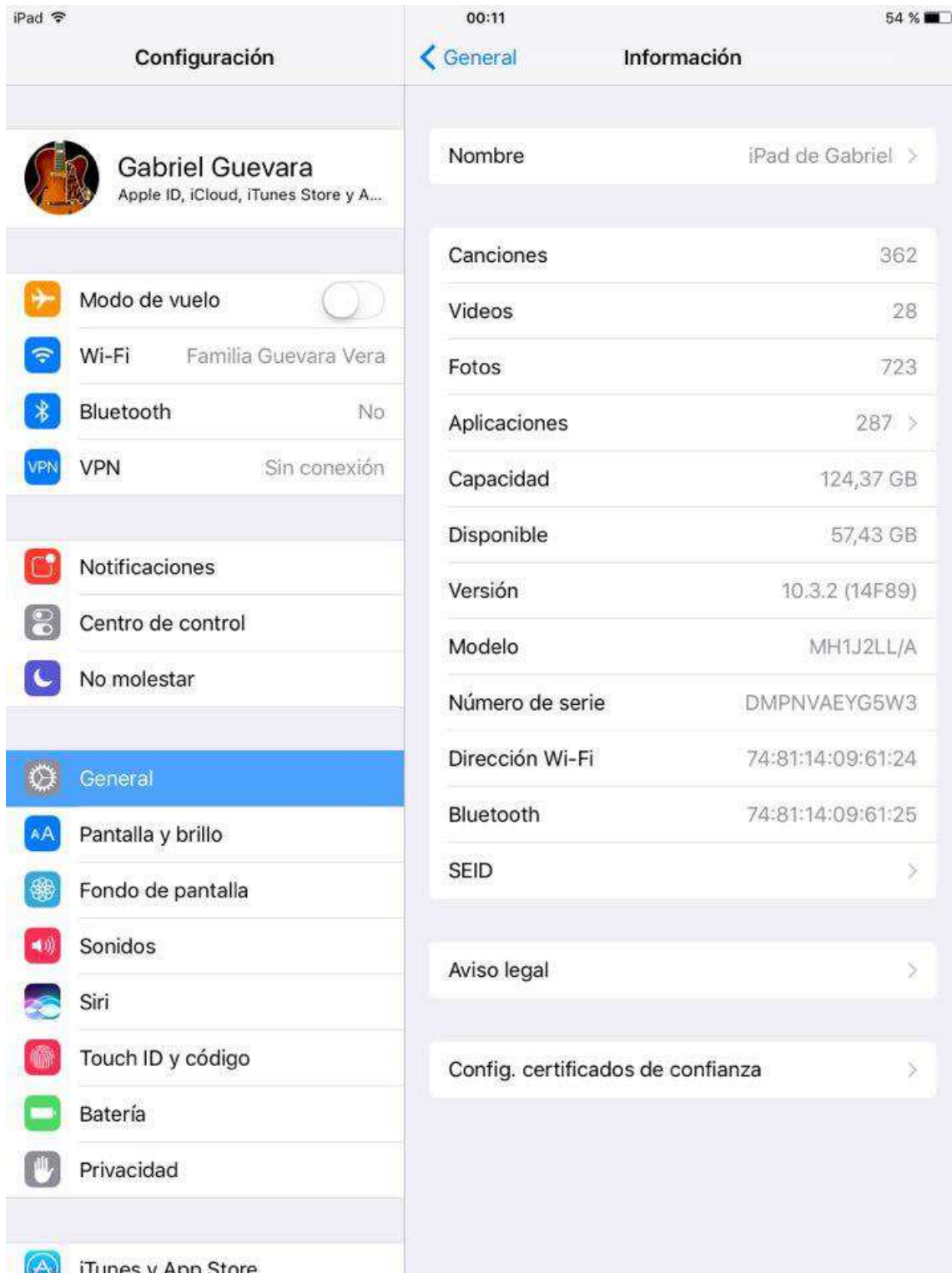


Ilustración 24 Características iPad utilizado para las pruebas



Para la implementación de la aplicación en un dispositivo iOS, se requirió su previo despliegue en la tienda de aplicaciones App Store con una cuenta de desarrollador vinculada a un Apple ID. Los usuarios requirieron poseer a su vez su Apple ID personal para poder descargar la aplicación desde la tienda. Los recursos del dispositivo para la instalación de la aplicación PseudoDF en dispositivos iOS son: 1 GB de RAM o superior y 35,7 MB de espacio de almacenamiento con un procesador Apple A7.

CARACTERÍSTICAS MOTO Z PLAY	
PANTALLA	IPS 5,5 Pulgadas
RESOLUCIÓN	1920x1080
PROCESADOR	Qualcomm Snapdragon 625 Octa-Core 64 bits 8x Cortex-A53 2GHz GPU Adreno 506
RAM	3 GB
MEMORIA	32 GB
VERSION SO	Android 7.0 Nugat

Tabla 26 Características Moto Z Play utilizado para las pruebas

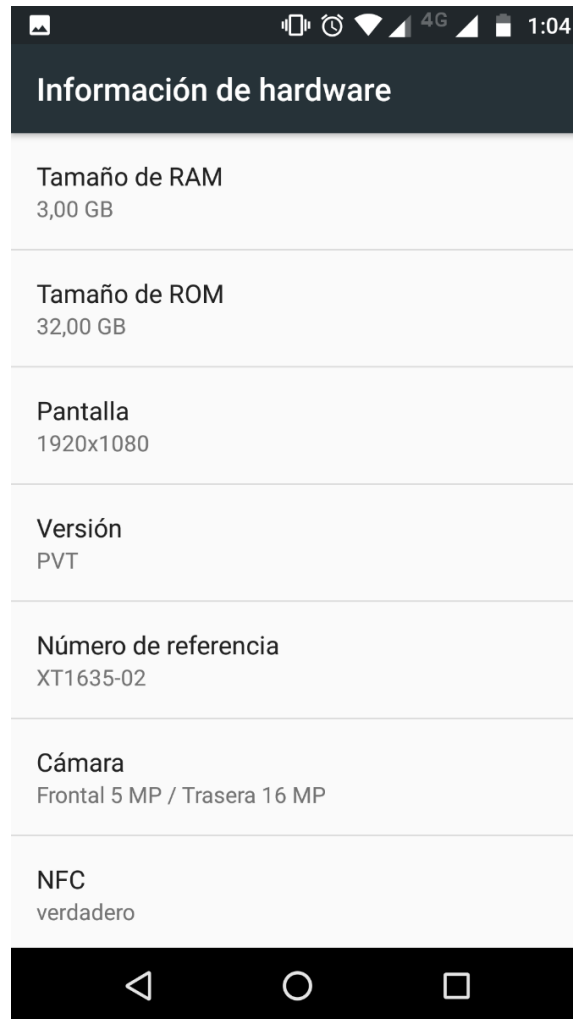


Ilustración 25 Características Moto Z Play utilizado para las pruebas

Para la implementación de la aplicación en un dispositivo Android, se requirió su previo despliegue en la tienda de aplicaciones Play Store con una cuenta de desarrollador vinculada a una cuenta de Google. Los usuarios requirieron poseer a su vez su cuenta de Google personal para poder descargar la aplicación desde la tienda. Los recursos del dispositivo para la instalación de la aplicación PseudoDF en dispositivos Android son: 1 GB de RAM o superior y 37,3 MB de espacio de almacenamiento con un procesador Dual-Core 1.2 GHz.

Grupo	Requerimiento	Prioridad	Objetivo Planteado	Resultado	Aceptación
Diseño de Algoritmos	Diagramación de Flujo de Datos	Alta	Las aplicaciones deben permitir abrir la paleta de componente y realizar drag and drop desde ella al espacio de diseño. Además, el usuario debe poder editar las sentencias dentro de cada componente haciendo doble tap sobre ellos, seleccionar los componentes, mostrando un marco de opciones para el objeto seleccionado permitiendo eliminarlos o realizar una conexión por líneas entre los componentes del lienzo.	El usuario puede abrir la paleta de componentes al requerir agregar un elemento, mostrando los componentes de diagramación disponibles, siendo posible presionar sobre ellos y arrastrarlos hacia el espacio de trabajo para poder editar su contenido (sentencia a ejecutar) al detectar un doble tap sobre él y ser conectado a otros componentes del lienzo por medio de las opciones de conexión mostradas al seleccionar el componente, así como también tener la posibilidad de eliminar el objeto, eliminando con él las líneas de conexión asociadas.	APROBADO
	Edición de Pseudocódigo	Alta	El usuario debe poder escribir en un editor de texto sentencia en estándar de pseudocódigo, reconociendo las palabras claves y resaltarlas con un color predefinido, además deber realizar tabulaciones automáticas al realizar salto de línea luego de escribir la	Las aplicaciones permiten escribir texto en un editor por medio del teclado del sistema operativo móvil, resaltando de color morado las palabras claves detectadas a medida que se van escribiendo; la tabulación es agregada al presionar la tecla retorno en el teclado cuando se ha abierto una estructura de control,	APROBADO

			apertura de una estructura de control.	incrementando la tabulación en estructuras anidadas.	
Conversión de Algoritmos	Conversión de Diagrama a Pseudocódigo	Media	El usuario requiere poder convertir el diagrama que ha diseñado previamente en el editor de diagramas de flujo a su equivalente pseudocódigo cambiando entre las vistas de edición.	Las aplicaciones proveen al usuario la posibilidad de cambiar las vistas entre el editor de diagramas de flujo y el editor de pseudocódigo, convirtiendo el diagrama de flujo a su equivalente pseudocódigo, siguiendo la secuencia de las conexiones.	APROBADO
	Conversión de Pseudocódigo a Diagrama	Media	El usuario requiere poder convertir el pseudocódigo que ha diseñado previamente en el editor de pseudocódigo a su equivalente diagrama de flujo cambiando entre las vistas de edición.	Las aplicaciones proveen al usuario la posibilidad de cambiar las vistas entre el editor de pseudocódigo y el editor de diagramas, convirtiendo el pseudocódigo a su equivalente diagrama de flujo, siguiendo la secuencia de los pasos escritos para conectar los componentes del diagrama.	APROBADO
Ejecución de Algoritmos	Ejecución de Algoritmos	Alta	Se requiere ejecutar los algoritmos tanto desde la vista de edición de diagramas de flujo como la vista de edición pseudocódigo, mostrando diálogos para las entradas y salidas solicitadas.	Las aplicaciones permiten al usuario ejecutar el algoritmo que han diseñado en la vista de diagramación o en la vista de pseudocódigo, recorriendo el algoritmo de manera secuencial, ejecutando las sentencias de cada componente o línea, siguiendo el camino de las	APROBADO

				estructuras de control o iterando según corresponda. Al solicitar datos se entrada se despliega una alerta con caja de texto para ingresar el valor de la variable y al necesitar mostrar datos se muestra una alerta con el valor requerido a mostrar. De suceder una excepción durante la ejecución se marca con un cuadro rojo el paso donde ocurrió el error.	
--	--	--	--	---	--

Análisis

En conjunto con el dueño del producto, se realizó la verificación del funcionamiento de la aplicación móvil PseudoDF para las plataformas iOS y Android en relación a los requerimientos extraídos durante la fase de investigación y refinándolos, durante el proceso de desarrollo; lo cual llevo a comprobar la experiencia de usuario y validez del funcionamiento y estabilidad de las aplicaciones al utilizar los distintos módulos de: Diagramación de Flujos de Datos, Edición de Pseudocódigo, Conversión de Diagrama a Pseudocódigo, Conversión de Pseudocódigo a Diagrama y Ejecución de Algoritmos.



Conclusiones

En base a los objetivos planteados en el presente trabajo de titulación, se formulan las siguientes conclusiones:

- En función de lo investigado y su utilización, el estándar ANSI (American National Standards Institute) para representar Diagramas de Flujo, es una de las notaciones más utilizadas para este fin, siendo claro y comprensible al momento de diseñar y revisar un algoritmo mientras que, para la representación de pseudocódigo se deben utilizar palabras claves que representen las acciones u operaciones, en un lenguaje natural, debido a que no existe un estándar a seguir.
- La metodología de desarrollo ágil Scrum es aplicable a proyectos externos al ámbito empresarial, permitiendo organizar la funcionalidad del producto final en productos entregables más pequeños, elaborados en iteraciones y permitiendo una detección temprana de errores. Además es un marco de trabajo independiente de tecnologías, permitiendo trabajar con los componentes de las plataformas iOS y Android requeridos para la construcción de la aplicación, como los SDK y JDK nativos, librerías de imágenes, interpretes, gráficos, eventos y permisos.
- La aplicación móvil PseudoDF desarrollada en el presente proyecto, es una herramienta que permite diseñar algoritmos por medio de Diagramas de Flujo o Pseudocódigo, y a su vez transformar el diseño de una técnica de representación a otra; esto permite realizar un análisis y comparación entre ambas técnicas.
- La herramienta cuenta con la características de ejecutar los algoritmos diseñados en ella, permitiendo verificar su correcto funcionamiento a través de las entradas y salidas esperadas, junto con los errores de ejecución o



excepciones producidas, que son controladas por la aplicación, deteniendo el proceso, informando al usuario mediante un mensaje de alerta e indicando el paso donde se ha producido el error.

- La herramienta PseudoDF fue desarrollada para las principales plataformas móviles como son iOS y Android, brindando acceso a ella desde una variedad de ubicaciones; donde el usuario puede diseñar algoritmos por medio de las dos técnicas de representación más utilizadas como son los Diagramas de Flujo y Pseudocódigo, además es posible verificar el funcionamiento de los algoritmos mediante su ejecución y análisis de resultados.



Recomendaciones

En base a los objetivos planteados en el presente trabajo de titulación, se proponen las siguientes recomendaciones:

- Los usuarios de la aplicación deben de conocer el funcionamiento de los diferentes componentes en las técnicas de diseño de algoritmos, antes de utilizar herramientas computarizadas o móviles, que facilitan el DEACON. Estas herramientas tienen la finalidad de agilizar el proceso de diseño para ayudar al desarrollo de la lógica.
- Incentivar a los estudiantes la realización de algoritmos computacionales para plantear una solución a un problema dado, y así clarificar las ideas previo a realizar la codificación del problema planteado utilizando la herramienta móvil PseudoDF para el desarrollo de algoritmo.
- Utilizar la característica de conversión entre las dos técnicas de representación de algoritmos que ofrece la aplicación desarrollada en el presente trabajo, y realizar un análisis en comparación de ellas para poder familiarizarse con ambas técnicas.
- Aplicar la característica de ejecución de algoritmos durante el proceso de verificación para comprobar la validez del algoritmo diseñado de forma interactiva.
- Emplear las clases de conversión de algoritmos como referencia para desarrollar un proyecto que permita transformar pseudocódigo o un diagrama de flujo a código de uno o varios lenguajes de programación.



Bibliografía

- Aniorte Hernández, N. (2016). *Nicanor Aniorte Hernández*. Obtenido de http://www.aniorte-nic.net/apunt_metod_investigac4_9.htm
- Apple Inc. (27 de Marzo de 2017). *About Swift*. Obtenido de The Swift Programming Language (Swift 3.1): https://developer.apple.com/library/content/documentation/Swift/Conceptual/Swift_Programming_Language/
- Arellano, J., Nieva, O., Solar, R., & Arista, G. (2012). Software para la enseñanza-aprendizaje de algoritmos estructurados. *Software para la enseñanza-aprendizaje de algoritmos estructurados*.
- Arias, F. G. (2012). *El Proyecto de Investigación*. Caracas: EPISTEME.
- Blaxter, L., Hughes, C., & Tight, M. (2002). *Cómo se hace una investigación*. Gedisa.
- Borrego Del Pino, S. (2008). Estadística Descriptiva e Inferencial. *Innovación y Experiencias Educativas*, 1-2.
- Caicedo Ausique, L. P., Herrera Hennessey, J. R., & Quenguan Sanabria, N. (2013). *El coaching como herramienta para el desarrollo de los recursos humanos en la empresa*. Pamplona, España: Universidad del Rosario.
- Campión, R. S., Filv, D. A., & Ochoa, A. D. (2014). ¿Pueden Las Aplicaciones Educativas De Los Dispositivos Mviles Ayudar Al Desarrollo De Las Inteligencias Mltiples? *Revista Electrónica de Tecnología Educativa*(47).
- Espinosa, R. S. (8 de Diciembre de 2010). Percepciones de estudiantes sobre el Aprendizaje mvil; la nueva generacin de la educacin a distancia. 21. Barcelona, Espaa.
- Fermn, N., Raul, S., & Javier, T. (2013). Valoraciones Del Profesorado Del rea De Fresno (California Central) Sobre La Influencia De La Tecnologa Mvil En El Aprendizaje De Sus Estudiantes.



- Ferri, C., Insfran, E., & Fernández, A. (2007). Una herramienta CASE para la mejora de la enseñanza de la ingeniería del software. *Una herramienta CASE para la mejora de la enseñanza de la ingeniería del software*.
- Fombona, J., Pascual, M., & Madeira, M. F. (Julio de 2012). Realidad Aumentada, Una Evolución De Las Aplicaciones De Los Dispositivos Mów. *Revista de Medios y Educación*(41).
- García Fernando, M., Ibañez, J., & Alvira, F. (1994). *El análisis de la realidad social: métodos y técnicas de investigación*. Madrid: Alianza Universidad. .
- García, J. C. (2009). Educación Básica - Algoritmos y Programación.
- Gomez, J. B. (2014). *Análisis y Diseño de Algoritmos*.
- Gomez, S. A., & Ramirez, C. D. (2011). Sistemas Operativos Móviles: Funcionalidades, Efectividad Y Aplicaciones Útiles En Colombia.
- Gosling, J., Joy, B., Steele, G., Bracha, G., & Buckley, A. (13 de Febrero de 2015). The Java® Language Specification Java SE 8 Edition.
- Hernandez, M. L. (Marzo de 2010). Diseño Estructurado de Algoritmos - Diagramas de Flujo y Pseudocódigos.
- Herrera, S. I., & Fennema, M. C. (Octubre de 2011). Tecnologías Móviles Aplicadas a la Educación Superior. *Tecnologías Móviles Aplicadas a la Educación Superior*.
- Inc, G. (s.f.). *Android, the world's most popular mobile platform*. Obtenido de Developers Android: <https://developer.android.com/about/android.html>
- Isaías Pérez Pérez, A. F. (2008). Estudio de la problemática presente en el diseño de algoritmos por computadora. *Estudio de la problemática presente en el diseño de algoritmos por computadora*.
- Isaías Pérez Pérez, C. A. (2010). Análisis del desempeño de los estudiantes en el diseño y construcción de algoritmos secuenciales, selectivos e iterativos.



Análisis del desempeño de los estudiantes en el diseño y construcción de algoritmos secuenciales, selectivos e iterativos.

Isaías Pérez Pérez, C. A. (2011). Nivel de utilización de las técnicas de diagramación por parte de los estudiantes, en el diseño de algoritmos secuenciales, selectivos e iterativos. *Nivel de utilización de las técnicas de diagramación por parte de los estudiantes, en el diseño de algoritmos secuenciales, selectivos e iterativos.*

Joyanes Aguilar Luis. (1996). *Fundamentos de Programación*. Madrid, España: McGRAW-HILL.

López Carreño, D. (2008). *SlideShare*. Obtenido de <https://www.slideshare.net/guestdfb3eb/introduccion-a-las-fuentes-de-informacion-especializadas>

López de Prado, R. (s.f.). *Zaguán 2000*. Obtenido de <http://www.oocities.org/zaguan2000/metodo.html>

Mejía, Ó. Á. (2012). *Android*.

Olmo, D. Y. (2013-2014). *Teorías De La Motivación Laboral Y Constructos Psicológicos Relacionados*. León, España: Universidad De León.

Parella, S. S., & Pestana, F. M. (2012). *Metodología de la Investigación Cuantitativa*. Caracas: FEDUPEL.

Pérez Pérez, I., & Monzalvo López, C. A. ((2012)). Análisis del nivel de utilización de las herramientas de diagramación manejadas dentro del diseño de algoritmos. *Análisis del nivel de utilización de las herramientas de diagramación manejadas dentro del diseño de algoritmos.*

Pérez Pérez, I., & Monzalvo López, C. A. (2012). Diseño de una propuesta de herramienta de diagramación para el desarrollo de algoritmos para computadora. *Diseño de una propuesta de herramienta de diagramación para el desarrollo de algoritmos para computadora.*



- Prado, A. d., & Lamas, N. (Diciembre de 2014). Alternativas para la enseñanza de pseudocódigo y diagrama de flujo. *Revista Electrónica Iberoamericana de Educación en Ciencias y Tecnología*, 5(3), 12.
- Quetglás, M., Toledo, F., & Cerverón, V. (2002). Fundamentos de Informática y Programación. *Fundamentos de Informática y Programación*.
- Rios, M. C. (2013). Composición Del Sistema Operativo Móvil iOS De Apple Y El Hardware Y Software Que.
- Rivera, Y. J., Cardona, J. S., & Franco, S. A. (2012). Sistema Operativo Android: Características Y Funcionalidad Para Dispositivos Móviles.
- Rivero, Y. (s.f.). *Rol de Medios*. Obtenido de <https://bloquemetodologicodelainvestigacionudo2010.wordpress.com/tecnicas-e-instrumentos-de-recoleccion-de-datos/>
- Robledo, J. (2012). Dispositivos móviles para el aprendizaje Lo que usted necesita saber.
- Rodríguez, I. A. (2016). Introducción a Scrum. *Gestión de proyectos con metodologías Ágiles y enfoques Lean*.
- Rosa, C. B., Lorena, C. C., & Adriana, O. (2015). Aplicación de una herramienta basada en Software Libre para la enseñanza de Algoritmos y Lógica de Programación. *Aplicación de una herramienta basada en Software Libre para la enseñanza de Algoritmos y Lógica de Programación*.
- Shih, G., Lakhani, P., & Nagy, P. (2010). *Is Android or iPhone the Platform for Innovation in Imaging Informatics*. Obtenido de SpringerLink: <http://link.springer.com/article/10.1007/s10278-009-9242-4>
- Silvestrini Ruiz, M. M., & Vargas Jorge, J. (Enero de 2008). *Universidad InterAmericana de Puerto Rico*. Obtenido de <http://ponce.inter.edu/cai/manuales/FUENTES-PRIMARIA.pdf>



- Sutherland, K. S. (2016). *La Guía de Scrum*. Obtenido de Scrum Guides: <http://www.scrumguides.org/docs/scrumguide/v2016/2016-Scrum-Guide-Spanish.pdf#zoom=100>
- The New Media Consortium; Universitat Oberta de Catalunya. (2012). *Perspectivas Tecnológicas: Educación Superior en Iberoamérica 2012-2017 Un Análisis Regional del Informe Horizon del NMC y la UOC*.
- Torres, M. (16 de 02 de 2014). *Métodos De Recolección De Datos Para Una Investigación*. Ciudad de Guatemala, Guatemala, Guatemala.
- tutorialspoint. (2015). *iOS Application Development*. Obtenido de tutorialspoint: https://www.tutorialspoint.com/ios/ios_tutorial.pdf
- Univ. Nacional de Entre Ríos. (s.f.). *Facultad de Ingeniería*. Obtenido de <http://bioingenieria.edu.ar/academica/catedras/metestad/muestreo.pdf>
- Universidad de Alicante. (2015). *Sistemas Operativos*.
- Universidad Nacional Experimental del Táchira. (s.f.). *UNET Virtual*. Obtenido de https://uvirtual.unet.edu.ve/pluginfile.php/10772/mod_resource/content/0/Guia_de_algoritmos.pdf
- Varelo, C. C., Redondo, M. R., & Palacín, A. S. (Junio de 2012). Tendencias actuales en el uso de dispositivos móviles en la educación. *La educación*(147).



Anexos

Anexo 1

Encuesta a los estudiantes de la FACCI.

 	UNIVERSIDAD LAICA "ELOY ALFARO" DE MANABI
	Creada el 13 de noviembre de 1985 mediante Decreto Ley No.10, publicado en el Registro Oficial No. 313
	FACULTAD DE CIENCIAS INFORMÁTICAS
	Creada, Resolución H. Consejo Universitario del 11 de Julio del 2001

El presente estudio está realizado por estudiantes de la FACCI con fin de recolectar información para su trabajo de titulación. Por el presente medio queremos garantizar que todas sus respuestas serán tratadas con la mayor confidencialidad posible.

INTRUCCIÓN: Marque con una equis su respuesta seleccionada

1. Considera usted que los Algoritmos Computacionales son la base en el Desarrollo De Software

Sí

No

2. ¿Cuál de las siguientes técnicas para el desarrollo de algoritmos conoce? (Seleccione una o más opciones).

Diagramas de Flujo

Pseudocódigo pseudocódigo

Diagramas N-S

Otras

3. De las siguientes técnicas para el desarrollo de algoritmos, seleccione su preferida

Diagrama de Flujo



Pseudocódigo

4. ¿Conoce herramientas que permitan desarrollar algoritmos por medio de un computador?

Sí

No

5. ¿Alguna de las herramientas para el desarrollo de algoritmos (de su conocimiento) le permite ejecutar y comprobar la validez del mismo (Computador)?

Sí

No

6. ¿Conoce herramientas que permitan desarrollar algoritmos por medio de un dispositivo móvil?

Sí

No

7. ¿Alguna de las herramientas para el desarrollo de algoritmos (de su conocimiento) le permite ejecutar y comprobar la validez del mismo (Dispositivo Móvil)?

Sí

No

8. ¿Cree usted que estas herramientas favorecen el proceso de aprendizaje?

Sí

No

9. ¿Considera usted relevante una herramienta que permita la conversión de diagramas de flujo a pseudocódigo y viceversa?

Sí



No

10. ¿Cuál es el Sistema Operativo de su dispositivo móvil? (Seleccione una o más opciones).

Android

iOS

Windows Phone

Symbian

Otro

11. La dificultad con la que se encuentran los estudiantes en el proceso de diseño de algoritmos computacionales, para la resolución de problema conlleva a dificultad el avance del estudiante dentro de la carrera incluso es una de las principales causas de la deserción estudiantil. ¿Está usted de acuerdo con lo manifestado en el texto anterior?

Sí

No



Anexo 2

Entrevista dirigida a los profesores de la FACCI.

1. ¿Considera fundamental el dominio en el desarrollo de algoritmos por parte de los estudiantes de la carrera de Ingeniería en Sistemas?
2. ¿Considera usted el desarrollo de algoritmos como la base esencial para el desarrollo de software?
3. ¿Considera que la dificultad en el desarrollo de algoritmos aumenta la complejidad en el avance del estudiante dentro de la carrera?
4. ¿De las técnicas de representación de algoritmos aplicadas en la institución, cual considera que es de mayor dificultad para los estudiantes?
5. ¿De las técnicas de representación de algoritmos aplicadas en la institución, cual considera que es de menor dificultad para los estudiantes?
6. ¿Considera que la ejecución de algoritmo de forma automatizada facilitaría el proceso de verificación del algoritmo?
7. ¿Que opina acerca de un software que permita diseñar un algoritmo con una técnica y convertirlo a otra que pueda representar mayor dificultad para el estudiante?
8. ¿Cuáles son las principales causas académicas que conllevan a la deserción del estudiante con respecto a la carrera de Ingeniería en Sistemas?



Anexo 3

Presentación de Encuesta realizada a los estudiantes de la FACCI.

Pregunta 1: Considera usted que los Algoritmos Computacionales son la base en el Desarrollo De Software.

Objetivo: Conocer si los estudiantes, consideran importante los algoritmos computacionales en el desarrollo de software.

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Si	150	98%
No	3	2%
Total	153	100%

Tabla 27 Tabulación de Pregunta 1 - Encuesta estudiantes FACCI

Representación Gráfica:

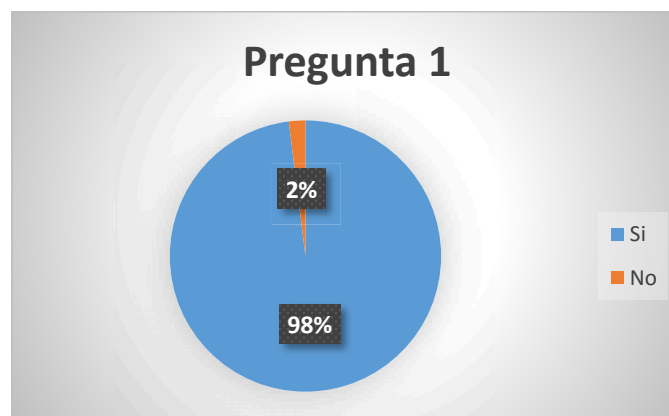


Ilustración 26 Gráfica de Pregunta 1 - Encuesta realizada de estudiantes de la FACCI

Análisis: Observando los resultados obtenidos en la presente estadística, se puede interpretar que para el 98% de los estudiantes encuestados; los algoritmos computacionales son la base del desarrollo de software, mientras que un 2% no los considera como tal.



Pregunta 2: ¿Cuál de las siguientes técnicas para el desarrollo de algoritmos conoce?

Objetivo: Determinar las técnicas más conocidas para la representación de los algoritmos computacionales.

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Diagramas de Flujo	131	45%
Pseudocódigo	131	45%
Otras	16	6%
Diagramas N-S	12	4%
Total	290	100%

Tabla 28 Tabulación de Pregunta 2 - Encuesta realizada a estudiantes FACCI

Representación Gráfica:

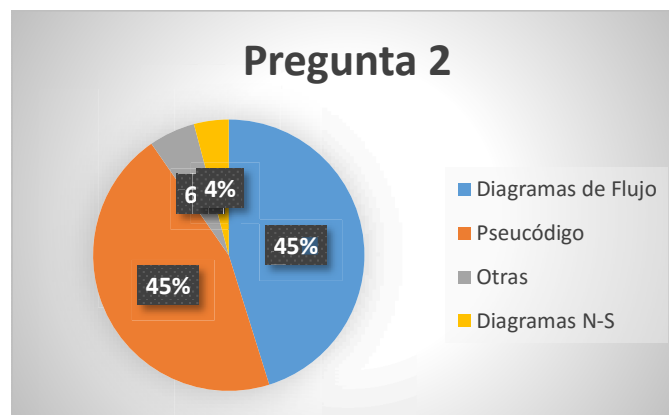


Ilustración 27 Gráfica de Pregunta 2 - Encuesta realizada a estudiantes de la FACCI

Análisis: Considerando la información anterior, se determina, que el 45% de los estudiantes encuestados conocen los Diagramas de Flujo, un 45% los Pseudocódigo, 4% los Diagramas N-S y un 6% otras técnicas no consideradas en la encuesta realizada.

Pregunta 3: De las siguientes técnicas para el desarrollo de algoritmos, seleccione su preferida.



Objetivo: Conocer la técnica preferida para los estudiantes, entre los Diagramas de Flujo y los Pseudocódigos.

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Diagrama de Flujo	82	54%
Pseudocódigo	71	46%
Total	153	100%

Tabla 29 Tabulación de Pregunta 3 - Encuesta realizada a estudiantes FACCI

Representación Gráfica:

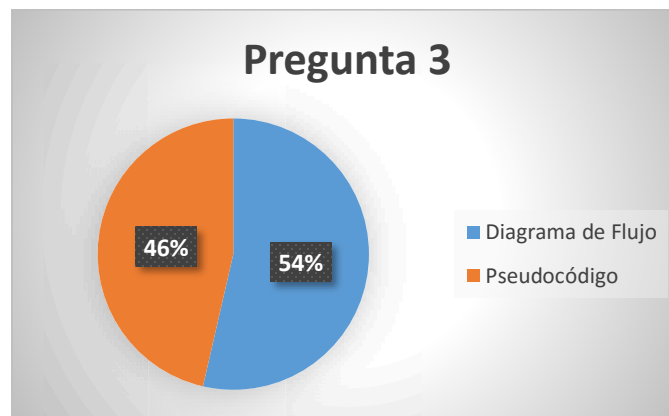


Ilustración 28 Gráfica de Pregunta 3 - Encuesta realizada a estudiantes de la FACCI

Análisis: En base a la información obtenida, se pudo determinar que el 54% de los estudiantes prefieren los Diagramas de flujos, mientras que un 46% los Pseudocódigo.

Pregunta 4: ¿Conoce herramientas que permitan desarrollar algoritmos por medio de un computador?

Objetivo: Conocer si los estudiantes están familiarizados con herramientas que permiten el desarrollo de algoritmos computacionales.

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Sí	120	78%



No	33	22%
Total	153	100%

Tabla 30 Tabulación de Pregunta 4 - Encuesta realizada a estudiantes FACCI

Representación Gráfica:

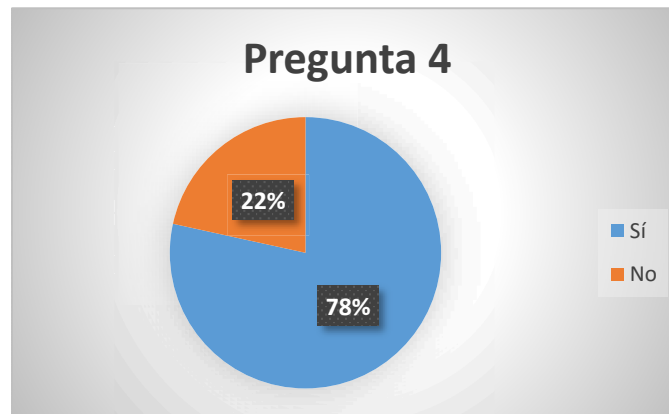


Ilustración 29 Gráfica de Pregunta 4 - Encuesta realizada a estudiantes de la FACCI

Análisis: Por medio de la estadística anterior, se conoció que el 78% de las estudiantes conocen herramientas que permiten desarrollar algoritmos computacionales en computadores, mientras que el 22% lo ignoran.

Pregunta 5: ¿Alguna de las herramientas para el desarrollo de algoritmos (de su conocimiento) le permite ejecutar y comprobar la validez del mismo (en un computador)?

Objetivo: Determinar el número de estudiantes que conocen herramientas que permitan el DEACON.

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Sí	91	59%
No	62	41%
Total	153	100%



Tabla 31 Tabulación de Pregunta 5 - Encuesta realizada a estudiantes FACCI

Representación Gráfica:

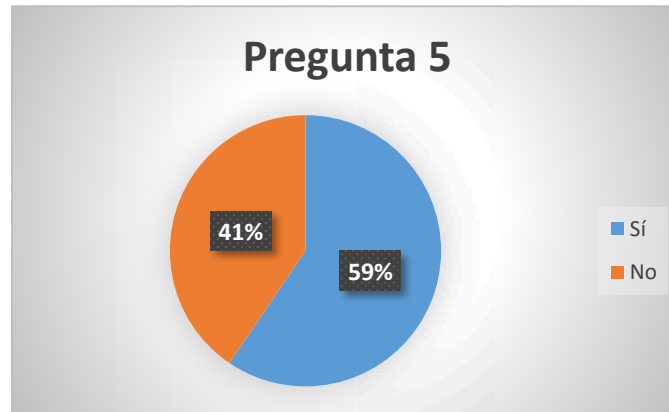


Ilustración 30 Gráfica de Pregunta 5 - Encuesta realizada a estudiantes de la FACCI

Análisis: Observando los datos estadísticos anterior, se determinó que el 59% de los alumnos conoce herramientas que permiten ejecutar y validar algoritmos computacionales en un computador, mientras que el 41% desconoce.

Pregunta 6: ¿Conoce herramientas que permitan desarrollar algoritmos por medio de un dispositivo móvil?

Objetivo: Determinar el número de estudiantes que conocen herramientas para el desarrollo de algoritmos en dispositivos móviles.

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Sí	126	82%
No	27	18%
Total	153	100%

Tabla 32 Tabulación de Pregunta 6 - Encuesta realizada a estudiantes FACCI



Representación Gráfica:

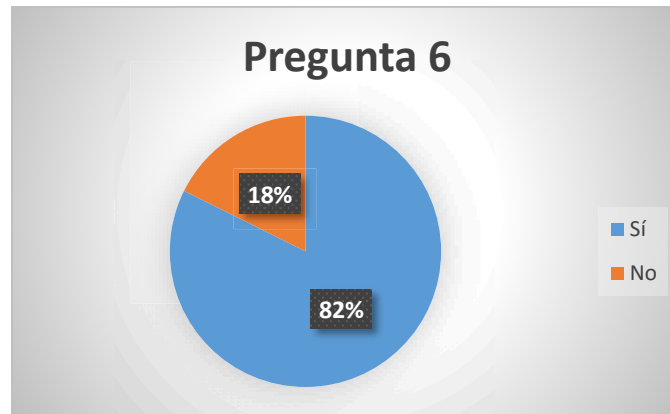


Ilustración 31 Gráfica de Preguntas 6 - Encuesta realizada a estudiantes de la FACCI

Análisis: Por medio de la estadística anterior, se conoció que el 82% de las estudiantes conocen herramientas que permiten desarrollar algoritmos computacionales en dispositivos móviles, mientras que el 18% lo ignoran.

Pregunta 7: ¿Alguna de las herramientas para el desarrollo de algoritmos (de su conocimiento) le permite ejecutar y comprobar la validez del mismo (en dispositivos móviles)?

Objetivo: Saber del conocimiento de los estudiantes con respecto a herramientas similares a las desarrolladas en el presente trabajo de titulación.

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Sí	19	12%
No	134	88%
Total	153	100%

Tabla 33 Tabulación de Preguntas 7 - Encuesta realizada a estudiantes FACCI



Representación Gráfica:

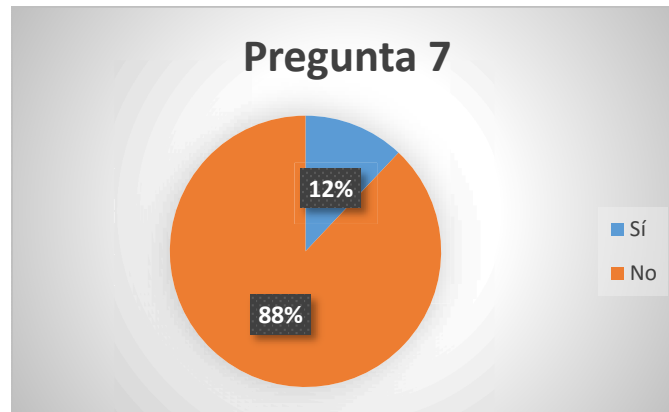


Ilustración 32 Gráfica de Pregunta 7 - Encuesta realizada a estudiantes de la FACCI

Análisis: Observando los datos estadísticos anterior, se determinó que el 88% de los alumnos desconocen herramientas que permiten ejecutar y validar algoritmos computaciones en dispositivos móviles, mientras que el 12% conocen.

Pregunta 8: ¿Cree usted, que estas herramientas favorecen el proceso de aprendizaje?

Objetivo: Determinar la postura de los estudiantes ante herramientas similares a la desarrollada en el presente trabajo de titulación

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Sí	143	93%
No	10	7%
Total	153	100%

Tabla 34 Tabulación de Pregunta 8 - Encuesta realizada a estudiantes FACCI



Representación Gráfica:

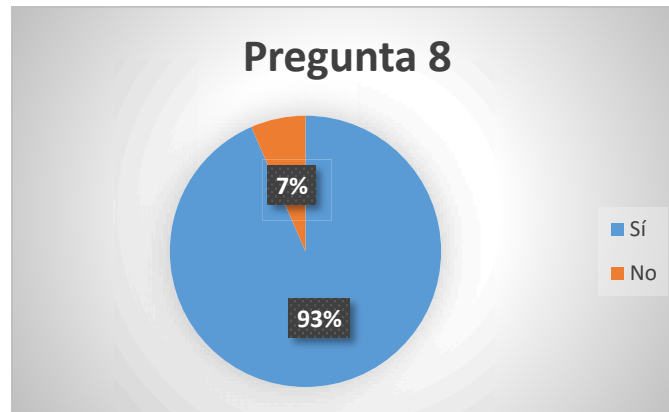


Ilustración 33 Gráfica de Preguntas 8 - Encuesta realizada a estudiantes de la FACCI

Análisis: Como se observa en la información anterior, el 93% de los estudiantes creen que las herramientas para el DEACON facilitan el aprendizaje, mientras que el 7% no lo consideran así.

Pregunta 9: ¿Considera usted relevante una herramienta que permita la conversión de diagramas de flujo a pseudocódigo y viceversa?

Objetivo: Analizar la importancia de los estudiantes con respecto a herramientas que permitan la conversión entre Diagramas de Flujo y Pseudocódigo.

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Sí	127	83%
No	26	17%
Total	153	100%

Tabla 35 Tabulación de Preguntas 9 - Encuesta realizada a estudiantes FACCI



Representación Gráfica:

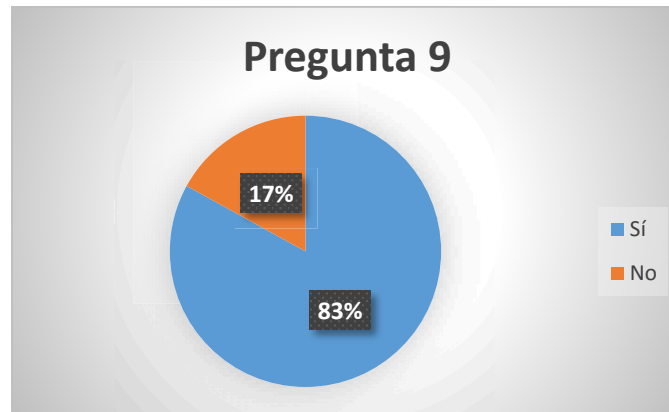


Ilustración 34 Gráfica de Pregunta 9 - Encuesta realizada a estudiantes de la FACCI

Análisis: En base a los datos obtenidos, se determinó que el 83% considera relevante una herramienta que permita la conversión de diagramas de flujo a pseudocódigo y viceversa, mientras que el 17% no lo consideran.

Pregunta 10: ¿Cuál es el Sistema Operativo de su dispositivo móvil?

Objetivo: Determinar el Sistema Operativo móvil más utilizado por los estudiantes.

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Android	131	79%
iOS	19	12%
Windows Phone	11	7%
Otro	4	2%
Symbian	0	0%
Total	165	100%

Tabla 36 Tabulación de Pregunta 10 - Encuesta realizada a estudiantes FACCI



Representación Gráfica:

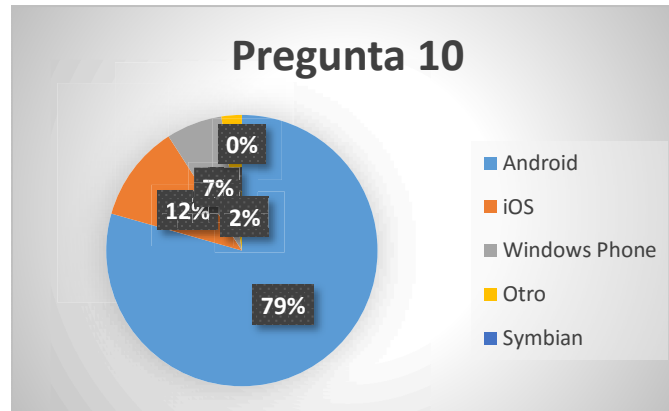


Ilustración 35 Gráfica de Pregunta 10 - Encuesta realizada a estudiantes de la FACCI

Análisis: Como se puede observar en la información anterior, el 79% de los estudiantes utilizan dispositivos móviles con Android, un 12% con iOS, un 7% con Windows Phone, un 2% utilizan otro sistema operativo diferente a los considerados en la encuesta y un 0% utiliza Symbian.

Pregunta 11: La dificultad con la que se encuentran los estudiantes en el proceso de diseño de algoritmos computacionales, para la resolución de problema conlleva a dificultad el avance del estudiante dentro de la carrera incluso es una de las principales causas de la deserción estudiantil. ¿Está usted de acuerdo con lo manifestado en el texto anterior?

Objetivo: Conocer si los estudiantes estiman que los Algoritmos Computacionales generan una dificultad académica en el proceso de aprendizaje.

Cuadro Referencia:

Alternativas	Frecuencia	Porcentaje
Sí	114	75%
No	39	25%



Total	153	100%
-------	-----	------

Tabla 37 Tabulación de Pregunta 11 - Encuesta realizada a estudiantes FACCI

Representación Gráfica:

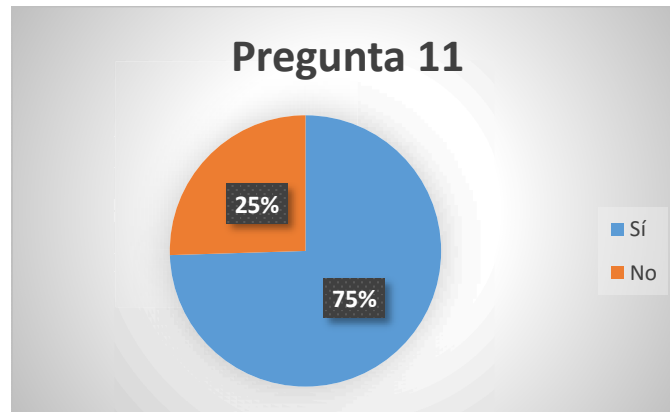


Ilustración 36 Gráfica de Pregunta 11 - Encuesta realizada a estudiantes de la FACCI

Análisis: En base a la estadística anterior, el 75% de los estudiantes están en acuerdo con lo manifestado, mientras que el 25% está en desacuerdo.



Anexo 4

Manual de Usuario