Universidad Politécnica de Pachuca

Codeland-oop

M. en C. Alicia Oriz Montes

Proyecto de Investigación

Integrantes:
Juan Antonio Ayola Cortes
Juan Daniel Soto Dimas
Miguel Ángel Vite Hernández

Ingeniería en Software

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

La programación orientada a objetos es uno paradigma de suma importancia dentro de la programación por lo que es basico conocerlo, y resulta muy útil que los estudiantes que cursan carreras técnicas o universitarias enfocadas al ámbito de la programación tengan un alto conocimiento de esto, sin embargo a pesar de llevarla como materia, les es difícil aprenderla por los amplios conceptos que maneja. Esto tambien conlleva a otro problema, que es la deserción de los alumnos de la carrera. La reprobación estudiantil es un problema añejo y complejo en las instituciones de educación superior. El propósito de este proyecto es aumentar el egreso de estudiantes.

1.1. Problematica

1.1.1. Ingeniería es el área con mayor tasa de abandono escolar universitario, particularmente aquellas vinculadas con la informática

El mercado laboral sufre el dilema vocacional porque las carreras más demandadas son las menos elegidas por los estudiantes después del primero año, ya que el abandono es mayor en el área de ingeniería y especialmente en informática. El foro profesional de los ingenieros técnicos en España advierte que el déficit de profesionales en el área de software y telecomunicaciones derivará en la importancia masiva de ingenieros dentro de una década.

1.1.2. Problemas del aprendizaje de la programación orientada a objetos

El problema del aprendizaje de la programación orientada a objetos se manifiesta toda vez que es una materia compleja que implica la integración de muchos elementos como son el paradigma orientado a objetos, el lenguaje de programación, el entorno de desarrollo, la metodología de desarrollo, el lenguaje de modelado, los patrones de desarrollo y la lógica de programación. Por lo tanto los alumnos se encuentran ante una cantidad abrumadora de conceptos en un periodo corto de tiempo, lo que dificulta su asimilación y el desarrollo de las habilidades para generar líneas de código como lo explica Spigariol and Passerini (2013):

> “Los docentes veían en los estudiantes que el uso del lenguaje representaba una curva de aprendizaje abrupta en los primeros momentos de la materia ya que requieren el manejo de una cantidad ámplia de conceptos antes de poder realizar algo relativamente sencillo (...). La disociación entre teoría y práctica que se generaba era ciertamente contraproducente y dificultaba el proceso de aprendizaje."

Debido a que los conceptos que se tratan en la asignatura de programación orientada a objetos son muchos y en algunos casos difíciles de comprender por el alto nivel de abstracción, esto se configura como un factor que dificulta el aprendizaje, esto se ha visto reflejado por

ejemplo cuando los estudiantes generalmente no distinguen entre lo que es una clase y un objeto.

Además la forma de enseñar la asignatura de programación orientada a objetos es muy similar a la de la programación estructurada, primero se tratan temas básicos del lenguaje de programación, como son la declaraciones de los tipos de datos, las estructuras de control y las sentencias de condición, para posteriormente enseñar lo que son las clases, objetos y los principales temas propios del paradigma orientada a objetos, esto contribuye a que los estudiantes sientan que continúan con el mismo paradigma de programación estructurada.

En virtud de que los estudiantes no logran asimilar este cambio de paradigmas, usan el lenguaje de programación orientada a objetos como un lenguaje de programación estructurada, por consiguiente no resuelven problemas diseñando clases.

Así pues los estudiantes al llegar a la asignatura de programación orientada a objetos se enfrentan de entrada a dos problemas, la percepción que tienen de dificultad y por otro lado el proceso de transición del paradigma estructurado hacia el paradigma orientado a objetos.

1.2. Objetivo general

Desarrollar una aplicación móvil con la herramienta de Android Studio, para que las personas interesadas en la programación puedan aprender de una mejor manera el paradigma de programación orientada a objetos, y así puedan adquirir y/o reforzar su conocimiento de este paradigma.

1.3. Objetivos específicos

- Proporcionar a los estudiantes y personas interesadas en la programación, información acerca del paradigma de programación orientada a objetos.
- Evaluar el conocimiento de los usuarios de la aplicación con exámenes al finalizar un tema.
- Diseñar una interfaz de tal forma que el usuario pueda aprender a utilizar la aplicación en el menor tiempo posible.
- Programar la aplicación de tal forma de que no consuma muchos recursos al usarla en un celular.

1.4. Justificación

1.4.1. Las TIC en la educación

De acuerdo con Palacios et al. (2011) en la educación el uso de las nuevas tecnologías de la información ha pasado por las siguientes etapas:

Esta evolución muestra como cada vez el aprendizaje va utilizando más tecnología; pasando de ser ésta simplemente una herramienta de apoyo, hasta ser la plataforma a través de la cual se presentan los contenidos y evalúan los conocimientos.

Mix-Learning. La etapa posterior al e-learning es la aplicación de una mezcla de sus herramientas con sistemas educativos tradicionales. La finalidad es dirigir más específicamente los contenidos a los estudiantes. Es así que el Blend Learning, Mix Learning o Hybrid Learning se presenta como “la combinación efectiva de los diferentes modelos de reparto, modelos de enseñanza y modelos de aprendizaje"(Olivares Carmona et al., 2016).

1.4.2. M-Learning en la educación

Según la Organización de las Naciones Unidas para la Educación la Ciencia y la Cultura (Dykes and Knight, 2012) existen 5.9 billones de suscripciones de teléfonos móviles en el

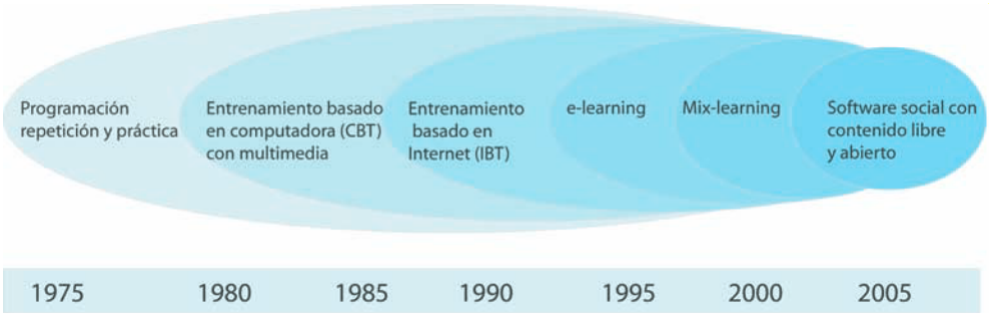

Figura 1.1: Evolución de las TIC en la educación. (Basado en Palacios et al. (2011))

mundo, contra los 7.04 billones de habitantes. Además, en el año 2020 los dispositivos móviles serán la principal herramienta de conexión a internet para la mayoría de la población; en Japón, actualmente el 75 % de su población tiene un dispositivo móvil como primer medio de acceso a internet (Camacho and Lara, 2011).

Asimismo, en años recientes el uso de la tecnología móvil para fines educativos, conocido como m-learning, ha tenido un grán desarrollo en la educación superior, ya que existen universidades de Europa y América que cuentan con sistemas de educación móvil (Traxler, 2007).

1.4.3. La importancia de la programación orientada a objetos

La programación Orientada a Objetos surge como el paradigma que permite manejar ampliamente las nuevas plataformas que garanticen desarrollar aplicaciones robustas, portables y reutilizables que puedan ofrecer una solución a largo plazo en un mundo donde los cambios se dan a cada momento.

El desarrollo de programas orientados a objetos es un enfoque diferente del mundo informático. Implica la creación de modelos del mundo real y la construcción de programas informáticos basados en esos modelos.

La importancia de esta programación radica en que, favorece la creación de programas de calidad, fuerza en mantenimiento, en extensión y reutilización de programas. Está basada en el modo de pensar del hombre y en el modo de trabajar de la máquina.

Es muy importante que los estudiantes sean capaces, no sólo de manejar los conceptos de orientación a objetos, sino también de aplicarlos de manera efectiva en el desarrollo de programas.

Capítulo 2

Estado del arte

Aplicaciones existentes para aprender programación orientada a objetos:

Nombre	Temas	Juegos	Tamaño	Idioma
Programación Orientada a Objetos	-Tipo de dato anónimo -Tipo abstracto -SOLID -RAII -Proxy -Programación basada en prototipos -Problema del diamante -Principio de sustitución de Liskov -Principio de segregación de la inerfaz -Principio de responsabilidad única-Objeto -Método -Metaclase -Encapsulamiento -Destructor -Delegación -Clase -Campo	No	3.87M	Español
Object Oriented Programming	-Object -Class -Constructor -Destructor -Get -Set -toString -Private Method -Protected Method -Public Method -Inheritance -Interfaces -Adbstrct -Plymorphism -Encapsulation	No	9.64M	Ingles

Object Oriented Programming	-OOP in PHP - What is Object -OOP in PHP - Intro to Class -OOP in PHP - Inheritance -OOP in PHP - Interfaces -OOP in PHP - Abstract -OOP in PHP - Constructor -OOP in PHP - Polymorphism -OOP in PHP - Encapsulation -OOP in PHP - Destructor -OOP in PHP - Private Method -OOP in PHP - Protected Method -OOP in PHP - Public Method	No	9.71M	Ingles
Object Oriented Programming	-Introduction to Object. -Class in OOP and its detail with syntax code. -Abstraction concepts in OOP -What is Polymorphism in OOP -Inheritance concepts with syntax code. -Method overloading vs. overriding -Encapsulation concept -Keywords in java -Constructor in Java etc. . . -Java Programming Statements with code syntax.	No	2.8M	Ingles
OPP for Beginners	-Naming convention -Object and class -Method overloading -Constructor -Static keyword -Inheritance -Aggregation -Method overrridong -covariant return type -Abstract class -Package -object class	No	8.1M	Ingles

Cuadro 2.1: Comparación de las distintas aplicaciones para aprender programación orientada a objetos.

Capítulo 3

Marco teórico

3.1. Programación Orientada a Objetos

La programación orientada a objetos, ha tomado las mejores ideas de la programación estructurada y los ha combinado con varios conceptos nuevos y potentes que incitan a contemplar las tareas de programación desde un nuevo punto de vista. La programación orientada a objetos, permite descomponer más fácilmente un problema en subgrupos de partes relacionadas del problema. Entonces, utilizando el lenguaje se pueden traducir estos subgrupos a unidades auto contenidas llamadas objetos.. (Izquierdo, 2007).

3.2. Programa

Un programa es una secuencia lógica de instrucciones escritas en un determinado lenguaje de programación que dicta a la computadora las acciones que debe llevar a cabo.

3.3. Tipo de dato

Es la representación simbólica de un atributo de una entidad; en programación, los datos expresan características de las entidades sobre las que opera un algoritmo. Los datos representan hechos, observaciones, cantidades o sucesos y pueden tomar la forma de números, letras o caracteres especiales (Izquierdo, 2007).

3.4. Objeto

Una estructura de datos y conjunto de procedimientos que operan sobre dicha estructura. Una definición más completa de objeto es: una entidad de programa que consiste en datos y todos aquellos procedimientos que pueden manipular aquellos datos; el acceso a los datos de un objeto es solamente a través de estos procedimientos, únicamente estos procedimientos pueden manipular, referenciar y/o modificar estos datos.

Para poder describir todos los objetos de un programa, conviene agrupar éstos en clases (Izquierdo, 2007).

3.5. Clase

Podemos considerar una clase como una colección de objetos que poseen características y operaciones comunes. Una clase contiene toda la información necesaria para crear nuevos objetos (Izquierdo, 2007).

3.6. Encapsulación

Es una técnica que permite localizar y ocultar los detalles de un objeto. La encapsulación previene que un objeto sea manipulado por operaciones distintas de las definidas. La encapsulación es como una caja negra que esconde los datos y solamente permite acceder a ellos de forma controlada (Izquierdo, 2007).

3.7. Abstracción

En el sentido más general, una abstracción es una representación concisa de una idea o de un objeto complicado. En un sentido más específico, la abstracción localiza y oculta los detalles de un modelo o diseño para generar y manipular objetos (Izquierdo, 2007).

3.8. Objetos

Un objeto es una entidad lógica que contiene datos y un código que manipula estos datos; el enlazado de código y de datos, de esta manera suele denominarse encapsulación. Cuando se define un objeto, se está creando implícitamente un nuevo tipo de datos (Izquierdo, 2007).

3.9. Polimorfismo

Esta característica es la capacidad que objetos similares tienen para responder de diferentes formas al mismo mensaje, y permite al programador implementar múltiples formas de un mismo método, dependiendo cada una de ellas de la clase sobre la que se realice la implementación. Esto permite acceder a varios métodos distintos utilizando el mismo medio de acceso (el mismo nombre). El polimorfismo está muy relacionado con la herencia (Velarde de Barraza et al., 2006).

3.10. Herencia

Según Joyanes Aguilar (1996), la herencia "Es la capacidad para crear nuevas clases de objetos que se construyen basados en clases existentes". La herencia es una propiedad que permite a un objeto poseer propiedades de otras clases. Además, a estos nuevos objetos creados es posible asignarles nuevos atributos y métodos.

La clase que puede ser heredada se denomina clase base (superclase) y la clase que hereda se denomina clase derivada (subclase) (Velarde de Barraza et al., 2006).

3.11. Atributos

Describen las características de los objetos: tipo de acceso (privado, protegido, publico) y tipo de dato (entero, real, booleano, etcétera) (Izquierdo, 2007).

3.12. Métodos

Describen lo que puede hacer la clase; es decir, el método define las instrucciones necesarias para realizar un proceso o tarea específicos. La definición del método se compone de tipo de acceso, tipo de retorno, nombre del método, parámetros, si los requiere, y el cuerpo del método (Izquierdo, 2007).

3.13. Algoritmo

Se define como una técnica de solución de problemas que consiste en una serie de instrucciones paso por paso y que produce resultados específicos para un problema determinado (Izquierdo, 2007).

3.14. Variable

Es un área de almacenamiento temporal a la que se ha asignado un nombre simbólico y cuyo valor puede ser modificado a lo largo de la ejecución de un programa (Izquierdo, 2007).

3.15. Constante

Es un valor definido que no cambia durante la ejecución de un programa (Izquierdo, 2007).

3.16. Parámetros formales

Son variables que reciben valores desde el punto de llamada que activa al método (Izquierdo, 2007).

3.17. Java

Java es uno de los lenguajes de programación más populares del mundo. Es un lenguaje orientado a objetos, potente, versátil y mutiplataforma (corre en cualquier sistema operativo moderno). Java fue elegido como el lenguaje para el entorno de desarrollo de Android, el sistema operativo móvil líder en smartphones y tablets (Izquierdo, 2007).

3.18. Android

Android es un sistema operativo inicialmente pensado para teléfonos móviles, al igual que iOS, Symbian y Blackberry OS. Lo que lo hace diferente es que está basado en Linux, un núcleo de sistema operativo libre, gratuito y multiplataforma (Studio, 2016).

3.19. SQLite

SQLite es un sistema de gestión de bases de datos relacional compatible con ACID, contenida en una relativamente pequeña (275 kiB) biblioteca escrita en C (Owens and Allen, 2010).

3.20. Material Design

Material Design es un concepto, una filosofía, unas pautas enfocadas al diseño utilizado en Android.

3.21. Git

Git es un sistema de control de versiones distribuido cuyo objetivo es el de permitir mantener una gran cantidad de código a una gran cantidad de programadores eficientemente (Alvarez and Alcázar, 2014).

3.22. UML

UML son las siglas de “Unified Modeling Language” o “Lenguaje Unificado de Modelado”. Se trata de un estándar que se ha adoptado a nivel internacional por numerosos organismos y empresas para crear esquemas, diagramas y documentación relativa a los desarrollos de software.

3.23. Android Studio

Es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJIDEA (Studio, 2016).

Capítulo 4

Desarrollo

Este proyecto se desarrollará con la metodología SCRUM con 8 sprints de una revision por semana, haciendo uso de historias de usuario y de la grafica del quemado para llevar un control de que historias de usuario se quemaron.

4.1. Recursos

4.1.1. Recursos humanos

Rol	Personas requeridas	Salario
Programador	1	$15,000
Diseñador	1	$10,000
Tester	1	$10,000
Lider de proyecto	1	$15,000

Cuadro 4.1: Pesonal requerido.

4.1.2. Recursos materiales

Equipo	Precio	Unidades necesarias
Computadoras	$8,000	2
Celular Android	$5,000	1

Cuadro 4.2: Recursos materiales.

4.1.3. Recursos tecnológicos

Concepto	Precio
GanttProject	$0
Pencil	$0
Android Studio	$0

Cuadro 4.3: Recursos tecnológicos

4.1.4. Recursos administrativos

Concepto	Precio
Agua	$100 al mes
Electricidad	$300 al mes
Internet	$500 al mes

Cuadro 4.4: Recursos administrativos

4.2. Diagrama de Gantt

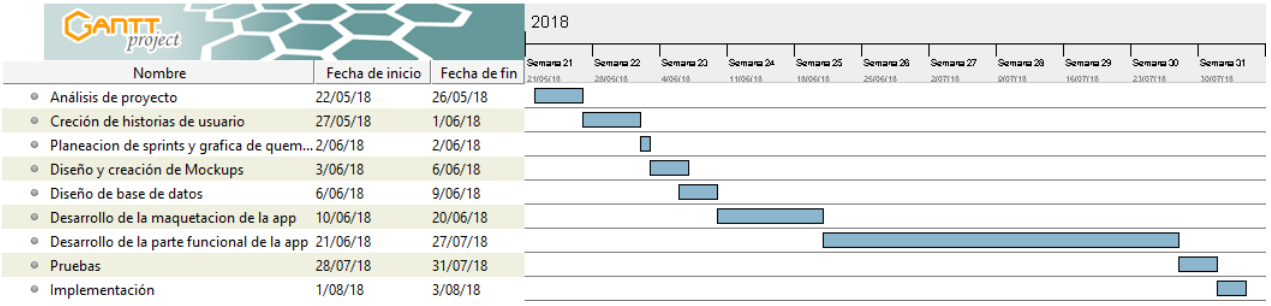

Figura 4.1: Cronograma de actividades a seguir.

4.3. Encuesta

Se realizó una encuesta a los alumnos de la carrera de Ingenieria en Software del 3° cutatrimestre de la Universidad Politecnica de Pachuca, con el fin de que con los resultados poder formular las historias de usuario. Se obtuvieron los siguientes resultados:

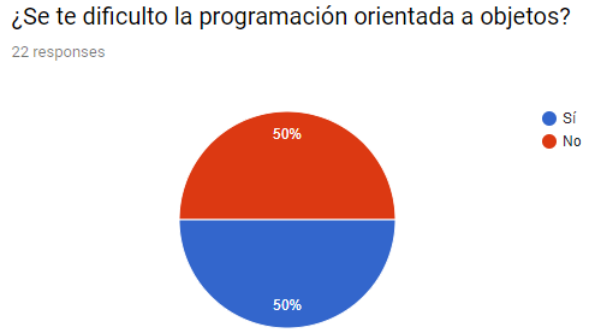

Figura 4.2: Primera pregunta de la encuesta..

En la primera pregunta se puede observar que a un 50 % de los alumnos de les dificulta la programación orientada a objetos, mientras que al 50 % restante no se le dificulta.

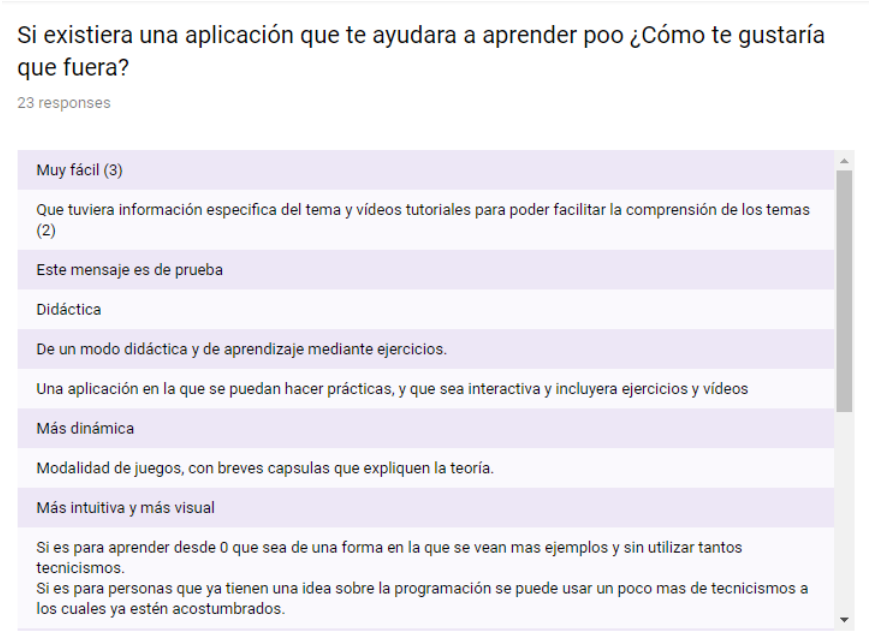

Figura 4.3: Segunda pregunta del cuestionario (parte 1).

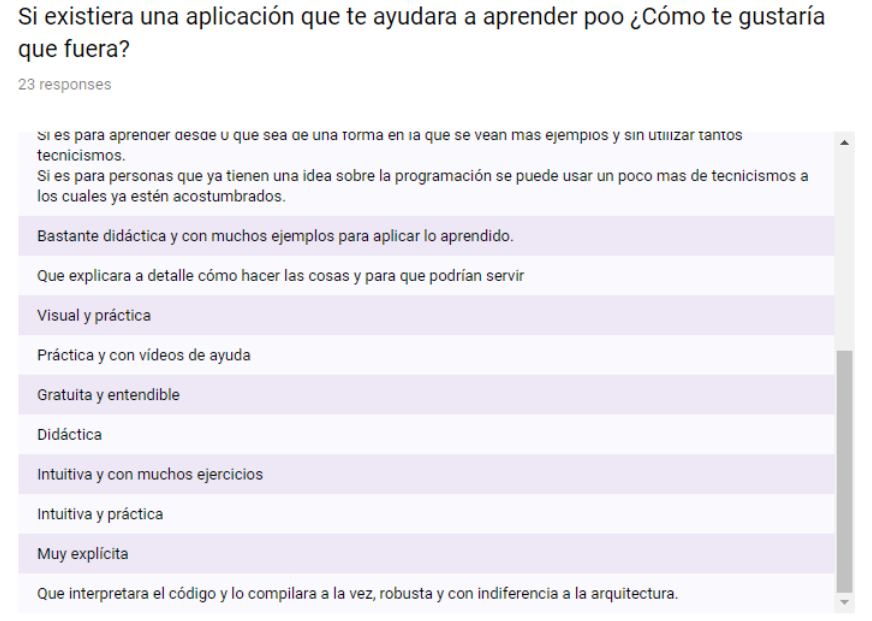

Figura 4.4: Segunda pregunta del cuestionario (parte 2).

En la segunda pregunta, tres alumnos respondiéron que la aplicación fuera "muy facil,"mientras que dos alumnos respondieron que la aplicación contuviera información muy especifica del tema y tutoriales para facilitar la comprensión del tema, mientras que los demás alumnos respondieron con respuestas como que sea didactica, dinámica, intuitiva, con una modadlidad de juegos, que no contenga muchos tecnisismos y con videos de ayuda.

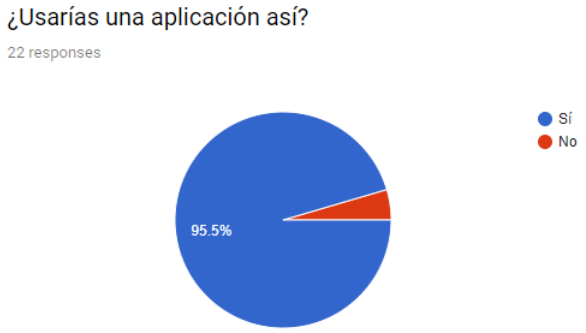

Figura 4.5: Tercera pregunta del cuestionario.

En la tercera pregunta del cuestionario, el 95.5 % de los alumnos repondieron aafirmativamente a que si utilizarían una aplicación de acuerdo a lo que respondieron anteriormente, mientras que el 4.5 % restante respondio de forma negativa.

4.4. Historias de usuario

1. Aplicación didactica		
Como usuario quisiera que la aplicación fuera de un modo didáctica y de aprendizaje mediante ejercicios.		
Estimación: 7	semanas Valor: 50	Dependencias:
Condiciones de satisfacción: ▪ La aplicación contendrá información selecta y bien explicada. ▪ Se incluirán algunos ejercicios los cuales deberán resolverse en la computadora.		

Cuadro 4.5: Historia de usuario 1

2. Explicaciónes detalladas		
Como usuario quisiera que explicara a detalle cómo hacer las cosas y para que podrían servir.		
Estimación: 7	semanas Valor: 60	Dependencias:
Condiciones de satisfacción: ▪ Se dará una aplicación detallada de cada tema así como ejemplos de uso.		

Cuadro 4.6: Historia de usuario 2

3. Modalidad de juegos		
Como usuario quisiera que la aplicación tuviera la modalidad de juegos, con breves capsulas que expliquen la teoría.		
Estimación: 6	semanas Valor: 60	Dependencias:
Condiciones de satisfacción: ▪ Se pondrá información sobre algunos conceptos y se hará un pequeño test de manera que ayude a recordar.		

Cuadro 4.7: Historia de usuario 3

4. Información especifica		
Como usuario quisiera que tuviera información específica del tema y vídeos tutoriales para poder facilitar la comprensión de los temas.		
Estimación: 6	semanas Valor: 30	Dependencias:
Condiciones de satisfacción: ▪ Se dará la información de manera específica pero en lugar de videos se incluirán algunos ejemplos.		

Cuadro 4.8: Historia de usuario 4

5. Visual y practica		
Como usuario quisiera que la aplicación fuera visual y práctica.		
Estimación: 4	semanas Valor: 20	Dependencias:
Condiciones de satisfacción: ▪ La aplicación será visual y de fácil utilización.		

Cuadro 4.9: Historia de usuario 5

6. Intuitiva		
Como usuario quisiera que la aplicación fuera intuitiva y con muchos ejercicios.		
Estimación: 3	semanas Valor: 20	Dependencias:
Condiciones de satisfacción: ▪ Se creará una interfaz amigable con el usuario y fácil de aprender.		

Cuadro 4.10: Historia de usuario 6

7. Con ejemplos y pocos tecnicismos		
Como usuario quisiera que si es para aprender desde 0 que sea de una forma en la que se vean más ejemplos y sin hacer uso excesivo de tecnicismos.		
Estimación: 3	semanas Valor: 20	Dependencias:
Condiciones de satisfacción: ▪ Se intentará utilizar pocas palabras técnicas, más sin embargo estas irán apareciendo para que después el usuario las pueda entender. ▪ Se incluirán varios ejemplos y ejercicios.		

Cuadro 4.11: Historia de usuario 7

4.5. Grafica de quemado

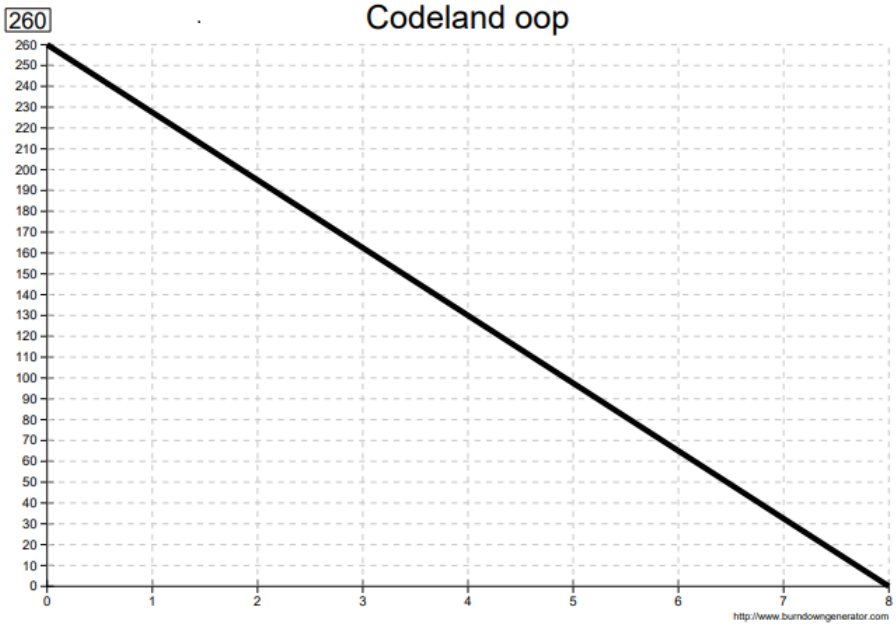

Figura 4.6: Grafica de quemado de las historias de usuario.

4.6. Maquetado

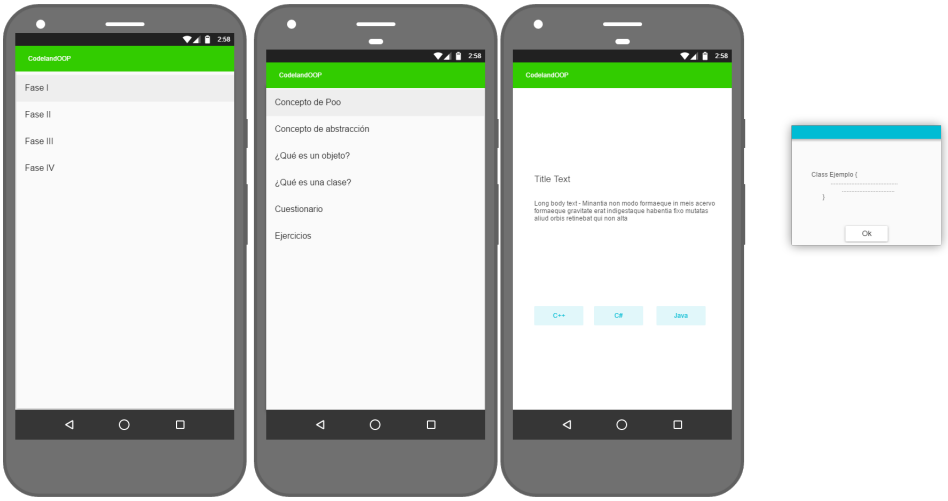

Figura 4.7: Pantallas de las fases, temas, tema, ejercicios y ejemplo.

Capítulo 5

Plan de pruebas

5.1. Historial de versiones

5.2. Fases

5.3. Alcance de las pruebas

5.3.1. Elementos de prueba

5.3.2. Pruebas de regresión

5.3.3. Funcionalidades a no probar

5.3.4. Enfoque de pruebas

5.4. Criterios de aceptacion o rechazo

5.4.1. Criterios de aceptación

5.4.2. Criterios de rechazo

5.4.3. Criterios de suspensión

5.4.4. Criterios de reanudación

5.5. Recursos

5.5.1. Requerimientos de entorno - Hardware

5.5.2. Requerimientos de entorno - Software

5.5.3. Herramientas requeridas

5.6. Planificacion y organizacion

5.6.1. Procedimientos

5.6.2. Cronograma

5.6.3. Dependencias y riesgos

5.7. Resultados de las pruebas

Capítulo 6

Resultados

Anexos

Referencias

Alvarez, M. and Alcázar, I. (2014). Introducción a git y github.

Camacho, M. and Lara, T. (2011). M-learning en españa, portugal y américa latina. *Recuperado de: http://scopeo.usal.es/sites/all/files/scopeom003.pdf*.

Dykes, G. and Knight, H. R. (2012). Mobile learning for teachers in europe: Exploring the potential of mobile technologies to support teachers and improve practice. *United Nations Educational, Scientific and Cultural Organization (UNESCO), UNESCO Working Paper Series on Mobile Learning, France*.

Izquierdo, L. R. (2007). Introducción a la programación orientada a objetos. *Publicación on line*.

Joyanes Aguilar, L. (1996). *Programación orientada a objetos*. Number 004.652. 5. McGraw-Hill Interamericana,.

Olivares Carmona, K. M., Angulo Armenta, J., Torres Gastelú, C. A., and Madrid García, E. M. (2016). Las tic en educación: metaanálisis sobre investigación y líneas emergentes en méxico. *Apertura (Guadalajara, Jal.)*, 8(2):100–115.

Owens, M. and Allen, G. (2010). *SQLite*. Springer.

Palacios, G. E. D., Gutiérrez, D., and Vargas, E. V. (2011). Tics en la educación.

Spigariol, L. and Passerini, N. (2013). Enseñando a programar en la orientación a objetos. *UTN FRC, Córdoba, Argentina*.

Studio, A. (2016). Conoce android studio. *Recuperado de: https://developer.android.com/studio/index.html*.

Traxler, J. (2007). Defining, discussing and evaluating mobile learning: The moving finger writes and having writ.... *The International Review of Research in Open and Distributed Learning*, 8(2).

Velarde de Barraza, O., Murillo de Velásquez, M., Gómez de Meléndez, L., and Castillo de Krol, F. (2006). Introducción a la programación orientada a objetos.