Instituto Politécnico Nacional

Escuela Superior De Computo

Desarrollo De Aplicaciones Móviles Nativas

Tarea 5

Propuesta de Proyecto Final:

“FlowCode: Compilador de Diagramas de Flujo a Código C para Dispositivos Móviles”

Nombre Del Alumno:

García Quiroz Gustavo Ivan | 2022630278

Grupo: 7CV3

Nombre Del Profesor: Hurtado Avilés Gabriel

Fecha De Entrega: 10/04/2025

Índice

FlowCode: Compilador de Diagramas de Flujo a Código C para Dispositivos Móviles

Resumen – La propuesta de proyecto Final propone el desarrollo de una aplicación móvil que permita a los usuarios diseñar algoritmos mediante diagramas de flujo en base a plantillas y traducirlos automáticamente a código en lenguaje C. La herramienta está orientada principalmente a estudiantes y docentes de programación, facilitando la transición entre el diseño algorítmico y la implementación en código. El sistema contará con un editor visual intuitivo para la creación de diagramas, un analizador para validar la estructura lógica, y un generador de código que producirá programas funcionales en C. Este proyecto busca cerrar la brecha entre la representación visual de algoritmos y su implementación práctica, sirviendo como herramienta educativa para la enseñanza de programación y como utilidad para el desarrollo rápido de prototipos algorítmicos.

1. Introducción

Contexto y Antecedentes

La programación es una habilidad fundamental en la formación de ingenieros en sistemas computacionales. Tradicionalmente, el proceso de desarrollo de software inicia con el diseño de algoritmos mediante representaciones visuales como los diagramas de flujo, para posteriormente implementarlos en un lenguaje de programación específico. Esta transición del diseño a la implementación representa un desafío significativo para estudiantes y programadores novatos, quienes deben dominar tanto los conceptos algorítmicos como la sintaxis específica del lenguaje de programación.

Los diagramas de flujo han sido utilizados durante décadas como herramientas pedagógicas en la enseñanza de la programación, ya que permiten representar

visualmente la secuencia de operaciones de un algoritmo. Sin embargo, la conversión manual de estos diagramas a código fuente es propensa a errores y consume tiempo considerable, especialmente para algoritmos complejos.

Planteamiento del Problema

Actualmente existen pocas herramientas específicamente diseñadas para dispositivos móviles que permitan generar código a partir de diagramas de flujo de manera eficiente y confiable. Las soluciones existentes suelen estar limitadas a entornos de escritorio, ser poco intuitivas, o generar código en lenguajes específicos que no siempre coinciden con las necesidades educativas.

La ausencia de una aplicación móvil eficiente para este propósito limita las oportunidades de comprensión y desarrollo para estudiantes, quienes podrían beneficiarse de una herramienta que les permita validar sus diseños algorítmicos y visualizar su implementación en código real desde cualquier lugar, usando dispositivos de uso cotidiano como teléfonos inteligentes o tabletas.

Propuesta de Solución

Este proyecto final propone desarrollar "FlowCode", una aplicación móvil que permita a los usuarios crear diagramas de flujo mediante una interfaz táctil intuitiva y generar automáticamente código en lenguaje C a partir de estos diagramas. La herramienta incluirá funcionalidades de validación para detectar errores de sintaxis, una representación intermedia para garantizar la correcta traducción, y un generador de código optimizado para producir programas funcionales y legibles.

La solución propuesta busca facilitar el proceso de aprendizaje y desarrollo de software, sirviendo como puente entre el diseño conceptual y la implementación práctica en un entorno móvil accesible.

A continuación, se presentan herramientas similares:

Nombre	Plataforma	Características principales	Limitaciones	Diferenciadores de FlowCode
Flowgorithm	Escritorio	Generación de código en varios lenguajes, simulación paso a paso	No disponible en móviles, interfaz no optimizada para táctil	Movilidad, interfaz táctil nativa, optimización para C
Raptor	Escritorio	Editor visual simple, ejecución de diagramas	Solo disponible para Windows, no genera código en C	Multiplataforma móvil, generación específica para C
PSeInt	Escritorio	Conversión de pseudocódigo a diagrama, múltiples idiomas	No tiene editor visual directo, no es para móviles	Editor visual táctil, enfoque directo en diagramas
Draw.io	Web/Móvil	Creación de diagramas generales	No genera código, no valida lógica algorítmica	Validación lógica, generación de código funcional
Lucidchart	Web/Móvil	Diagramas colaborativos en la nube	Generación de código limitada, enfoque general	Enfoque específico en programación, análisis semántico

Tabla 1. Resumen de productos similares.

La aplicación propuesta busca llenar este vacío ofreciendo una solución nativa para Android que combine la potencia de la creación de diagramas basada en código con una interfaz optimizada para dispositivos móviles, permitiendo a los usuarios crear, editar y compartir diagramas técnicos de manera eficiente y sin necesidad de conexión a internet.

2. Objetivo

Desarrollar una aplicación móvil Android para crear diagramas de flujo y a partir de éstos generar código fuente funcional en lenguaje C y permitir la comprensión de los conceptos de programación y el diseño de algoritmos.

2.1 Objetivos específicos

- Diseñar e implementar una interfaz de usuario táctil que facilite la creación y edición de diagramas de flujo en dispositivos móviles.
- Desarrollar un sistema de validación que verifique la estructura sintactica y semántica de los diagramas creados, identificando errores y ambigüedades.
- Implementar un motor de análisis que traduzca los diagramas a una representación intermedia que preserve su semántica.
- Crear un generador de código que transforme la representación intermedia en código C funcional, correctamente estructurado y documentado.
- Integrar funcionalidades de guardado, carga y exportación de diagramas y código generado.
- Generar manuales de usuario, documentación técnica de la arquitectura y un informe detallado del proceso de creación, que sirvan como referencia para facilitar el mantenimiento de la aplicación.

3. Justificación

Relevancia Académica y Profesional

Este trabajo terminal presenta una solución innovadora que integra conceptos fundamentales de ingeniería de software, compiladores, interfaces hombre-máquina y programación móvil. La implementación requiere aplicar conocimientos avanzados de estructuras de datos, análisis sintáctico, generación de código y diseño de interfaces, demostrando así la capacidad del estudiante para resolver problemas complejos de ingeniería.

Impacto Educativo

La herramienta propuesta tiene un alto valor pedagógico, ya que:

- Facilita la comprensión de estructuras algorítmicas al vincular su representación visual con su implementación en código.
- Reduce la barrera de entrada para estudiantes que se inician en la programación.
- Proporciona retroalimentación inmediata sobre errores lógicos y sintácticos.
- Permite a docentes crear y compartir ejemplos algorítmicos de forma visual y ejecutable.

Originalidad e Innovación

A diferencia de las herramientas existentes, FlowCode combina:

- Un enfoque nativo para dispositivos móviles con interfaz táctil optimizada.
- Generación específica hacia lenguaje C, ampliamente utilizado en educación.
- Validación semántica de los diagramas antes de la generación de código.
- Potencial para exportar y compartir tanto diagramas como código resultante.

Factibilidad Técnica

El desarrollo de la aplicación es técnicamente viable considerando:

- Existencia de frameworks multiplataforma como Flutter o React Native para desarrollo móvil.
- Algoritmos conocidos de análisis de grafos para validación de diagramas.
- Técnicas establecidas de compiladores para la generación de código.
- Hardware móvil actual con capacidad suficiente para ejecutar las tareas requeridas.

4. Productos o resultados esperados

Al concluir el trabajo terminal, se espera obtener los siguientes productos:

- **Aplicación móvil :** Aplicación móvil funcional para dispositivos Android que permita la creación, edición y traducción de diagramas de flujo a código C.

Las herramientas y tecnologías que vamos a usar son:

- **Desarrollo móvil:** Flutter/Dart o React Native
- **Gestión de datos:** SQLite o solución de almacenamiento local
- **Análisis de diagramas:** Implementación propia basada en teoría de grafos
- **Generación de código:** Técnicas de compiladores adaptadas al contexto
- **Control de versiones:** Git/GitHub
- **Gestión del proyecto:** Trello o ClickUp

5. Arquitectura del Sistema

La arquitectura del sistema FlowCode seguirá un patrón de diseño en capas que separa claramente las responsabilidades de cada componente. La siguiente figura muestra la arquitectura propuesta:

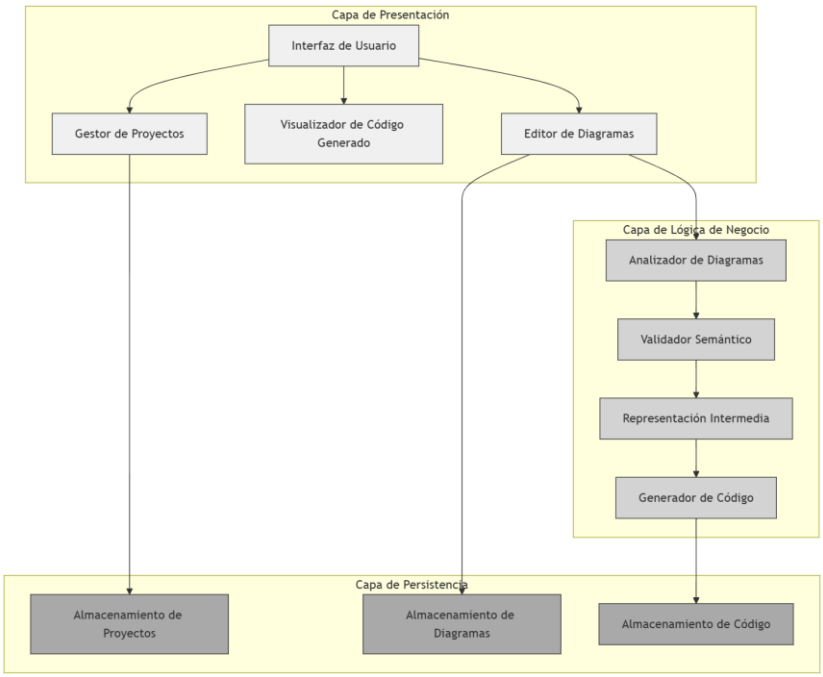

Figura 1 Arquitectura del sistema. **Fuente:** Elaboración propia.

La arquitectura del sistema se divide en tres capas principales:

1. **Capa de Presentación:**
    - Interfaz de Usuario: Maneja todas las interacciones con el usuario.
    - Editor de Diagramas: Proporciona la interfaz gráfica para la creación y edición de diagramas de flujo.
    - Visualizador de Código Generado: Muestra el código C generado con formato adecuado.

- Gestor de Proyectos: Permite administrar los diagramas y archivos de código.

2. **Capa de Lógica de Negocio:**
    - Analizador de Diagramas: Procesa la estructura del diagrama de flujo.
    - Validador Semántico: Verifica la correctitud lógica y semántica del diagrama.
    - Representación Intermedia: Traduce el diagrama a una estructura interna que facilita la generación de código.
    - Generador de Código: Produce el código C a partir de la representación intermedia.
3. **Capa de Persistencia:**
    - Almacenamiento de Proyectos: Gestiona la persistencia de los proyectos del usuario.
    - Almacenamiento de Diagramas: Almacena los diagramas de flujo en formato serializado.
    - Almacenamiento de Código: Gestiona el almacenamiento del código generado.

6. Especificación de Requisitos Funcionales

Requisitos del Editor Visual

- **RF01:** El sistema permitirá crear diagramas de flujo mediante una interfaz táctil.
- **RF02:** El sistema ofrecerá una biblioteca de símbolos estándar (inicio/fin, proceso, decisión, entrada/salida, bucles).
- **RF03:** El sistema permitirá conectar elementos del diagrama mediante gestos táctiles.
- **RF04:** El sistema implementará zoom y desplazamiento para navegación por el diagrama.
- **RF05:** El sistema permitirá editar las propiedades de cada elemento (texto, condiciones, expresiones).

Requisitos del Analizador y Validador

- **RF06:** El sistema validará la estructura del diagrama (conexiones válidas, elementos requeridos).
- **RF07:** El sistema verificará la semántica del diagrama (variables declaradas, tipos compatibles).
- **RF08:** El sistema identificará y reportará caminos sin salida o código inalcanzable.
- **RF09:** El sistema mostrará visualmente los errores en el diagrama.

- **RF10:** El sistema verificará que todos los caminos terminen en un elemento fin.

Requisitos del Generador de Código

- **RF11:** El sistema generará código C sintácticamente correcto a partir del diagrama.
- **RF12:** El sistema producirá código con formato adecuado (indentación, comentarios).
- **RF13:** El sistema incluirá declaraciones de variables necesarias.
- **RF14:** El sistema mapeará correctamente estructuras de control (if-else, while, for).
- **RF15:** El sistema permitirá visualizar el código generado en tiempo real.

Requisitos de Gestión

- **RF16:** El sistema permitirá guardar y cargar diagramas.
- **RF17:** El sistema exportará diagramas como imágenes.
- **RF18:** El sistema exportará el código generado como archivo .c o .txt.
- **RF19:** El sistema organizará los diagramas en proyectos o carpetas.
- **RF20:** El sistema permitirá compartir diagramas y código mediante servicios estándar del dispositivo.

7. Especificación de Requisitos No Funcionales

Usabilidad

- **RNF02:** El tiempo de comprensión de la aplicación para usuarios novatos será menor a 30 minutos.
- **RNF03:** Los iconos y elementos visuales seguirán estándares reconocidos de diagramas de flujo.

Rendimiento

- **RNF06:** La aplicación funcionará eficientemente en dispositivos con 2GB de RAM o superior.

Confiabilidad

- **RNF07:** El sistema realizará autoguardado cada 5 minutos para prevenir pérdida de datos.
- **RNF08:** La tasa de errores en la generación de código será menor al 1% para diagramas válidos.

Compatibilidad

- **RNF10:** La aplicación será compatible con Android 8.0 o superior.
- **RNF12:** El código generado será compatible con compiladores de C estándar (GCC, Clang).