Escuela de Ingeniería en Computadores

Algoritmos y Estructuras de Datos II

Profesor: Jose Isaac Ramírez Herrera

TE #2: Propuesta de investigación

Alejandro Arias
alearias@estudiantec.cr

Fabricio González
fabgonzalez@estudiantec.cr

Steven Aguilar Alvarez
s.aguilar.3@estudiantec.cr

PLANTEAMIENTO GENERAL DEL PROYECTO

1. Descripción general del proyecto:

El proyecto tiene como objetivo la implementación de una calculadora avanzada capaz de evaluar expresiones matemáticas y lógicas de cualquier longitud mediante el uso de árboles de expresión binarios, en el contexto de una arquitectura distribuida cliente-servidor. Este trabajo se desarrolla como parte del curso “Algoritmos y Estructuras de Datos I” del Instituto Tecnológico de Costa Rica, y representa una oportunidad para integrar diversos conceptos del área de ciencias de la computación en una solución funcional.

La motivación principal que origina este proyecto es fortalecer las competencias técnicas en la construcción y manipulación de estructuras de datos como los árboles binarios, así como en la aplicación práctica de la notación postfija para la conversión de expresiones infijas. Esta notación, elimina la ambigüedad en el orden de evaluación de los operadores, lo que permite representar y procesar de forma más efectiva expresiones complejas. Además, el desarrollo del proyecto requiere tener un gran conocimiento en la comunicación mediante sockets TCP, al tener que manejar múltiples clientes conectados simultáneamente al servidor.

El problema central que se pretende resolver es la evaluación estructurada, segura y precisa de expresiones compuestas por múltiples operaciones, sin depender del uso manual de paréntesis que pueden ser malinterpretadas por los usuarios o mal implementadas en soluciones lineales. Las expresiones pueden incluir operaciones aritméticas (+, -, *, /, %, **) y lógicas (and, or, xor, not), lo cual incrementa el grado de complejidad. Para solucionarlo, se utilizarán árboles de expresión binarios, donde cada nodo interno representa un operador y cada nodo hoja representa un operando.

Adicionalmente, el sistema implementado tendrá funcionalidades complementarias que mejoran la experiencia del usuario y aportan trazabilidad al proceso. El cliente contará con una interfaz gráfica que le permitirá ingresar expresiones de forma sencilla. Una vez enviada la expresión al servidor, este se encargará de construir el árbol, evaluarlo y devolver el resultado al cliente, quien lo visualizará junto con su historial. El servidor, por su parte, registrará cada operación realizada en un archivo `.csv` que incluirá la expresión evaluada, el resultado y la fecha de procesamiento. Esta funcionalidad no solo permite al usuario revisar expresiones previas, sino que también facilita el análisis posterior de uso o verificar cómo se comporta el sistema.

Finalmente, el proyecto se gestionará utilizando metodologías ágiles apoyadas por herramientas como Azure DevOps, donde se planificarán tareas a través de historias de usuario distribuidas en sprints. Cada historia definirá una acción deseada por el usuario y el beneficio correspondiente. Asimismo, se hará uso obligatorio de sistemas de control de versiones con Git y plataformas colaborativas como GitHub para garantizar un trabajo ordenado, seguro y transparente entre los integrantes del equipo.

2. Justificación

El desarrollo de una calculadora distribuida basada en árboles de expresión binarios y arquitectura cliente-servidor responde a la necesidad creciente de construir sistemas computacionales capaces de procesar grandes volúmenes de información de manera estructurada, eficiente y remota. En un contexto donde las aplicaciones de este tipo, y el procesamiento lógico-matemático se vuelven cada vez más fundamentales en sectores como la educación, la industria y las finanzas, este proyecto ofrece una solución aplicable, actual y con un gran valor didáctico y tecnológico.

La pertinencia de este proyecto está en su enfoque práctico y su potencial formativo. Por un lado, permite aplicar conceptos avanzados de estructuras de datos, diseño de interfaces gráficas y planificación ágil de proyectos, áreas clave en la formación de profesionales en ingeniería en Computadores. Por otro lado, el abordaje propuesto, la evaluación de expresiones mediante árboles binarios en una arquitectura distribuida desarrollada en C# no solo refleja problemáticas reales en el desarrollo de software, sino que también ofrece una base replicable para soluciones más complejas.

Como investigación aplicada, el proyecto tiene como principales beneficiarios a estudiantes y docentes de la carrera de ingeniería en Computadores, quienes podrán usar la herramienta como recurso educativo para reforzar la comprensión en el curso de algoritmos y estructuras de datos I. También puede beneficiar a desarrolladores de software que requieran módulos reutilizables para el procesamiento lógico y matemático de expresiones complejas. A nivel institucional, refuerza el compromiso del Instituto Tecnológico de Costa Rica con la formación de profesionales capaces de crear soluciones innovadoras.

Desde el punto de vista económico y tecnológico, este tipo de desarrollos contribuye al fortalecimiento de las competencias digitales del país, al fomentar el uso de tecnologías de software libre, el trabajo colaborativo mediante control de versiones y el enfoque de desarrollo ágil. Socialmente, promueve la familiarización tecnológica, la resolución de problemas mediante el pensamiento computacional y el acceso abierto a herramientas de formación, alineándose con políticas nacionales como la Estrategia Nacional de Sociedad y Economía del Conocimiento (MICITT) y los objetivos del Plan Nacional de Desarrollo y de Inversión Pública 2023-2026, que planea promover la innovación en los procesos productivos para la mejora de la competitividad y el desarrollo de país

Por lo tanto, esta propuesta representa una gran oportunidad para aportar al crecimiento académico de nosotros como estudiantes y al avance de las capacidades tecnológicas del Instituto Tecnológico de Costa Rica, como del país.

3. Marco Teórico:

El desarrollo de una calculadora avanzada que evalúe expresiones matemáticas y lógicas de longitud arbitraria, utilizando árboles de expresión binarios en una arquitectura cliente-servidor basada en sockets TCP, requiere la comprensión y aplicación de diversos conceptos fundamentales en ciencias de la computación y desarrollo de software.

Árboles de Expresión Binarios

Los árboles de expresión binarios son estructuras de datos que representan expresiones aritméticas y lógicas de manera jerárquica. En estos árboles, los nodos internos corresponden a operadores, mientras que los nodos hoja representan operandos. La evaluación de un árbol de expresión se realiza mediante un recorrido recursivo, evaluando los subárboles izquierdo y derecho antes de aplicar el operador en el nodo actual. Esta metodología garantiza una evaluación correcta respetando la jerarquía de operaciones [1].

Notación Postfija

La notación postfija, también conocida como notación polaca inversa, es una forma de escribir expresiones en la que los operadores siguen a sus operandos, eliminando la necesidad de paréntesis y facilitando la evaluación de expresiones mediante estructuras como pilas o árboles de expresión [1] [2].

Programación de Sockets TCP y Arquitectura Cliente-Servidor

La arquitectura cliente-servidor es un modelo de diseño en el que múltiples clientes se comunican con un servidor central para solicitar servicios o recursos. La comunicación entre cliente y servidor puede implementarse mediante sockets TCP, que proporcionan una conexión confiable y orientada a la transmisión de datos en secuencia. La implementación de sockets TCP en sistemas modernos ha evolucionado para permitir extensiones y personalizaciones [3].

Manejo de Archivos CSV

El formato CSV (Comma-Separated Values) es utilizado para almacenar y transferir datos en forma de texto plano, donde cada línea representa un registro y los campos están separados por comas. Este tipo de archivo es compatible con una gran variedad de programas, como hojas de cálculo (por ejemplo, Microsoft Excel o Google Sheets) y sistemas de bases de datos [4].

Interfaces Gráficas de Usuario (GUI)

Las interfaces gráficas de usuario permiten una interacción más intuitiva entre el usuario y la aplicación. Para el desarrollo de esta, C# ofrece tecnologías como Windows Forms y Windows Presentation Foundation (WPF). Donde WPF es un marco de interfaz de usuario que permite crear aplicaciones de escritorio con interfaces modernas con diferentes funcionalidades. Utiliza el lenguaje XAML para definir la interfaz de usuario de manera declarativa, lo que facilita la separación entre el diseño y la lógica de la aplicación [5].

Metodologías Ágiles y Azure DevOps

La gestión de proyectos de software se beneficia del uso de metodologías ágiles, como Scrum, que promueven la entrega incremental y la colaboración continua. Azure DevOps es una plataforma que facilita la planificación, y colaboración en proyectos, permitiendo la creación de historias de usuario, planificación de sprints y seguimiento del progreso del equipo. La integración de prácticas ágiles con Azure DevOps mejora la colaboración, velocidad y gestión de flujos de trabajo, brindando mejores resultados [6].

Control de Versiones con Git y GitHub

El control de versiones es una práctica fundamental en el desarrollo de software que permite gestionar y registrar los cambios realizados en el código fuente a lo largo del tiempo. Git es un sistema de control de versiones distribuido que facilita la colaboración entre múltiples desarrolladores, permitiendo que cada uno trabaje en paralelo sin interferir en el trabajo de los demás. Ampliando esto último, se puede hacer mediante la creación de ramas ya sea para desarrollar nuevas funcionalidades o corregir errores de forma aislada, etc. Integrándolas posteriormente al proyecto principal mediante fusiones [7].

Colaboración y Flujo de Trabajo en Equipos de Desarrollo

La colaboración efectiva en equipos de desarrollo de software es esencial para el éxito de los proyectos. Implica la coordinación de esfuerzos, la comunicación clara y la gestión adecuada de tareas y responsabilidades. Un flujo de trabajo bien definido establece las etapas del desarrollo, desde la planificación hasta la entrega, y asigna roles y tareas específicas a los miembros del equipo. Esto no solo mejora la productividad, sino que también reduce la posibilidad de errores [8].

4. Objetivo General

- Desarrollar una aplicación distribuida en lenguaje C# que permita evaluar expresiones matemáticas y lógicas de cualquier longitud mediante árboles de expresión binarios, implementada bajo una arquitectura cliente-servidor utilizando sockets TCP, con capacidad para múltiples clientes y un sistema de registro y consulta de resultados.

5. Objetivos Específicos

- Diseñar e implementar una arquitectura cliente-servidor en C# utilizando sockets TCP, que permita a múltiples clientes conectarse simultáneamente al servidor para el envío de expresiones a evaluar.
- Construir y evaluar árboles de expresión binarios a partir de expresiones matemáticas y lógicas en notación postfija, utilizando estructuras recursivas eficientes para representar y calcular los resultados.
- Desarrollar una interfaz gráfica en C#, que facilite el ingreso de expresiones, la visualización de resultados y el acceso al historial de operaciones realizadas por cada cliente.
- Registrar en archivos CSV las expresiones procesadas por el servidor, incluyendo el resultado obtenido y la fecha de evaluación.
- Organizar y planificar el desarrollo del proyecto mediante metodologías ágiles, utilizando Azure DevOps para la gestión de tareas, planificación de sprints e implementación de historias de usuario.

6. Metodología:

1. Enfoque de solución

El proyecto se abordará desde un enfoque incremental y modular. Se empleará el paradigma de programación orientada a objetos en C#, junto con principios de diseño limpio, para garantizar una arquitectura mantenible y escalable. La solución se dividirá en tres componentes principales:

- Cliente: aplicación de escritorio con interfaz gráfica que permita el ingreso de expresiones y visualización de resultados e historial.

- Servidor: componente encargado de recibir las expresiones, construir y evaluar los árboles binarios y registrar los resultados.

- Canal de comunicación: implementación de sockets TCP que permita la conexión de múltiples clientes concurrentes.

Cada módulo será probado de forma independiente mediante pruebas unitarias y posteriormente integrado para pruebas funcionales completas.

2. Recolección, sistematización y análisis de información

Dado que se trata de un proyecto de desarrollo, la recolección de información se centrará en el registro automatizado de las operaciones realizadas por los usuarios. Para ello, el servidor guardará en archivos .csv los siguientes campos por cada evaluación:

- Expresión ingresada

- Resultado obtenido

- Fecha de evaluación

Estos datos serán utilizados para la verificación del correcto funcionamiento del sistema, para análisis de uso y para la retroalimentación del equipo desarrollador.

3. Suposiciones del proyecto

Este desarrollo se basa en una serie de suposiciones clave que delimitan su alcance. En primer lugar, se asume que los usuarios introducirán expresiones válidas en notación infija, sin errores de sintaxis ni operadores desconocidos. También se presupone que el servidor será ejecutado en un entorno controlado, accesible por los clientes a través de una red local o mediante dirección IP fija. Además, se considera que las expresiones estarán compuestas exclusivamente por los operadores definidos en el sistema, que incluyen operaciones aritméticas como suma, resta, multiplicación, división, módulo y potencia, así como operaciones lógicas como AND, OR, NOT y XOR.

4. Equipos y técnicas utilizadas

- Lenguaje y entorno: C# con .NET 8.0 o superior; Visual Studio como entorno de desarrollo.
- Gestión de versiones: Git y GitHub para seguimiento del código y trabajo colaborativo.
- Planificación: Azure DevOps, donde se definirán historias de usuario agrupadas en sprints, con tareas asignadas.

6. Diseño experimental

Se diseñará un entorno de prueba controlado en el cual se simularán múltiples clientes realizando solicitudes al servidor. Se generará una muestra de al menos 50 expresiones distintas, evaluadas automáticamente para comparar los resultados esperados con los generados por el sistema.

- Muestras: Muestreo intencional de expresiones de diferentes longitudes y niveles de complejidad.
- Análisis: Las expresiones serán contrastadas con resultados obtenidos desde calculadoras analógicas para verificar su exactitud.

7. Plan de Acción:

PLAN DE ACCIÓN				
Objetivo Específico	**Productos**	**Actividades**	**Período de ejecución**	**Responsable por actividad**
Diseñar e implementar la arquitectura cliente-servidor utilizando sockets TCP	Sistema cliente-servidor funcional	- Diseño del protocolo de comunicación - Programación del servidor y manejo de múltiples clientes - Pruebas de conexión TCP	Semana 1 a Semana 2	Fabricio, Steven
Construir y evaluar árboles de expresión binarios a partir de expresiones postfijas	Módulo de evaluación de expresiones Árbol binario funcional	- Implementación del parser a notación postfija - Construcción del árbol binario - Desarrollo del evaluador recursivo	Semana 2 a Semana 3	Fabricio
Desarrollar una interfaz gráfica de usuario en C# (WPF)	Aplicación cliente con interfaz funcional	- Diseño de la interfaz gráfica - Integración con el sistema cliente - Validación de entrada y visualización de resultados	Semana 3 a Semana 4	Alejandro, Steven
Registrar operaciones en archivos CSV desde el servidor	Sistema de registro de operaciones	- Diseño del formato de registro - Implementación del generador de CSV	Semana 4 a Semana 5	Alejandro
Organizar y planificar el desarrollo con metodologías ágiles usando Azure DevOps	Tablero de tareas en DevOps Sprints y cronograma definidos	- Creación del tablero - Redacción de historias de usuario - Asignación de tareas	Semana 1 (inicio del proyecto)	Todos

8. Definición del cronograma:

Semana	Actividades	Responsable
1	- Definición de requerimientos funcionales - Diseño del protocolo cliente-servidor TCP - Creación del repositorio Git y tablero en Azure DevOps - Redacción de historias de usuario	Todos
2	- Programación del servidor y manejo de múltiples clientes - Desarrollo del cliente TCP - Implementación del parser a notación postfija	Todos
3	- Construcción del árbol de expresión binario - Desarrollo del evaluador recursivo - Diseño e implementación de la interfaz gráfica (WPF)	Todos
4	- Integración de módulos cliente-GUI-servidor - Registro de operaciones en CSV - Revisión general y validación de funciones	Todos
5	- Pruebas funcionales completas con casos variados - Simulación de múltiples clientes - Revisión final del código y documentación	Todos

9. Divulgación y transferencia de tecnología:

Entre las principales acciones de divulgación se contempla el repositorio público en GitHub, donde se estará el código fuente del sistema con su documentación técnica, manual de usuario, y ejemplos de uso. Esto permitirá que otros estudiantes, docentes o investigadores puedan estudiar, modificar y extender el proyecto, promoviendo la cultura del software libre y la colaboración académica.

Además, se piensa en la presentación de los resultados en actividades institucionales, tales como ferias vocacionales, congresos estudiantiles, etc. Organizados por el ITCR. En estas actividades, se mostrará el funcionamiento del sistema y se explicará su arquitectura, los principios técnicos aplicados y las oportunidades de mejora o adaptación.

Desde la perspectiva de transferencia tecnológica, la aplicación podría ser adoptado como material educativo para docentes de cursos como Algoritmos y Estructuras de Datos I. Para ello, se elaboraría una guía de implementación, donde se expliquen los objetivos de aprendizaje que el sistema puede apoyar, así como posibles ejercicios para clases/talleres.

10. **Presupuesto:**

Concepto	Cantidad	Precio Unitario	Precio Total	Justificación
Servicio de internet mensual	**3**	**10,000.00**	**30,000.00**	**Conectividad mensual estimada por estudiante para desarrollo y pruebas.**
Papelería y materiales de apoyo	**1**	**2,000.00**	**2,000.00**	**Material para anotaciones, planificación y revisión entre pares.**
Energía eléctrica	**3**	**3,000.00**	**9,000.00**	**Estimación del costo energético durante el tiempo total de desarrollo.**

11. Bibliografía

[1] J. I. Ramirez Herrera, *Algoritmos y estructuras de datos* [eBook]. Costa Rica, 2025.

[2] A. Borbón Alpízar, "¿Cómo evaluar expresiones matemáticas en el computador?", *Revista Digital: Matemática, Educación e Internet*, vol. 7, no. 2, 2014. [En línea]. Disponible en: https://revistas.tec.ac.cr/index.php/matematica/article/view/2054

[3] H. J. Fuquene Ardila, «IMPLEMENTACIÓN CLIENTE SERVIDOR MEDIANTE SOCKETS», *Rev. Vínculos*, vol. 8, n.º 2, pp. 48–59, dic. 2011.

[4] *CSV File Guide*, Noruega: Publifye AS, 2025.

[5] Microsoft, "Aplicación Hola mundo con WPF en C# - Visual Studio (Windows)," 2024. [En línea]. Disponible en: https://learn.microsoft.com/es-es/visualstudio/get-started/csharp/tutorial-wpf?view=vs-2022

[6] J. Rossberg, "Agile Project Management in Azure DevOps and TFS," en *Agile Project Management with Azure DevOps*, Apress, 2019, pp. 251–306.

[7] Arias, A. P. (2015). *Control de versiones de software con GIT*. Createspace Independent Publishing Platform.

[8] Caiza, J. C., Guamán, D. S., & López, G. R. (2015). Herramientas de desarrollo con soporte colaborativo en Ingeniería de Software. *Enfoque UTE*, 6(2), 102–116.