Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 2do. Semestre 2025

SIMULADOR PARA LA OPTIMIZACIÓN DE SISTEMAS DE RIEGO ROBÓTICO MEDIANTE LA IMPLEMENTACIÓN DE TIPOS DE DATOS ABSTRACTOS Y PROGRAMACIÓN ORIENTADA A OBJETOS

202200147 – Mario Rodrigo Balam Churunel

Resumen

La agricultura de precisión es una tendencia tecnológica global que busca optimizar la gestión de cultivos. En Guatemala, su adopción puede incrementar la competitividad del sector agrícola, aunque la inversión en infraestructura, como las redes de sensores, representa un impacto económico significativo. Este ensayo presenta una solución de software que aborda este desafío, optimizando la cantidad de estaciones base de recolección de datos mediante un algoritmo de agrupamiento. La solución, implementada en Python bajo el paradigma de Programación Orientada a Objetos, procesa datos de configuración en formato XML e identifica patrones de conectividad idénticos entre estaciones para fusionarlas. La principal conclusión es que es factible reducir significativamente la infraestructura física necesaria sin pérdida de información, logrando un sistema más rentable y escalable. La novedad del enfoque radica en el cumplimiento de estrictas restricciones de desarrollo, como la implementación manual de todos los Tipos de Datos Abstractos

Palabras clave

Agricultura de Precisión, Optimización, Agrupamiento, Python, XML.

Abstract

Precision agriculture is a global technological trend aimed at optimizing crop management. In Guatemala, its adoption can increase the competitiveness of the agricultural sector, although the investment in infrastructure, such as sensor networks, represents a significant economic impact. This essay presents a software solution that addresses this challenge by optimizing the number of data collection base stations through a clustering algorithm. The solution, implemented in Python under the Object-Oriented Programming paradigm, processes configuration data in XML format and identifies identical connectivity patterns among stations to merge them. The main conclusion is that it is feasible to significantly reduce the required physical infrastructure without information loss, achieving a more cost-effective and scalable system. The novelty of the approach lies in its adherence to strict development constraints, such as the manual implementation of all Abstract Data Types.

Keywords

Precision Agriculture, Optimization, Clustering, Python, XML.

Introducción

La modernización del sector agrícola es fundamental para la sostenibilidad y la competitividad económica. La automatización del riego mediante sistemas robóticos emerge como una solución prometedora, pero su implementación efectiva depende de una planificación logística rigurosa. ¿Es posible diseñar un sistema de software que permita simular y validar la eficiencia de diferentes planes operativos antes de su costoso despliegue en el campo? Este ensayo tiene como propósito describir el desarrollo de un simulador que responde afirmativamente a esta interrogante. El sistema utiliza un enfoque de simulación por eventos discretos para modelar el comportamiento de una flota de drones, basándose en la premisa de que la eficiencia de un plan puede medirse en términos de tiempo total y consumo de recursos. A continuación, se detallará la arquitectura del software, las estructuras de datos implementadas y el algoritmo de simulación que constituye el núcleo del proyecto.

Desarrollo del tema

El núcleo de la solución se basa en un proceso estructurado que transforma la descripción de un invernadero y sus planes de riego en un reporte detallado de simulación. Este proceso se divide en cuatro fases principales: carga y modelado, estructuración de tareas, simulación concurrente y generación de reportes.

a. Carga y Modelado de Datos

El sistema inicia con la lectura de un archivo XML que define la topología del invernadero: drones, plantas con sus requerimientos específicos y planes de riego. Para gestionar esta información en memoria, se utilizó el paradigma de Programación Orientada a Objetos, modelando cada entidad (Dron, Planta, Invernadero, PlanRiego) como una clase con atributos y comportamientos definidos. Una restricción fundamental del proyecto fue la prohibición de usar estructuras de datos nativas de Python. Por ello, se implementó desde cero un Tipo de Dato Abstracto (TDA) de Lista Enlazada para manejar todas las colecciones de objetos, garantizando una gestión de memoria dinámica y un control preciso sobre las operaciones.

b. Estructuración de Tareas con una Cola

Una vez cargados los datos, el plan de riego, que es una secuencia de texto (ej. "H1-P2, H2-P1,..."), debe ser transformado en una estructura procesable. Para esto, se implementó un TDA de Cola sobre la Lista Enlazada. La elección de una Cola es conceptualmente idónea, ya que un plan de riego sigue un principio estricto de "primero en entrar, primero en salir" (FIFO). Cada tarea (ej. "H1-P2") se encola en la estructura, y el simulador siempre atenderá la tarea que se encuentre al frente. La Figura 1 muestra una representación visual de este TDA, donde el estado de las tareas pendientes puede ser analizado en cualquier momento.

Figura 1. Representación gráfica del TDA Cola (FIFO)

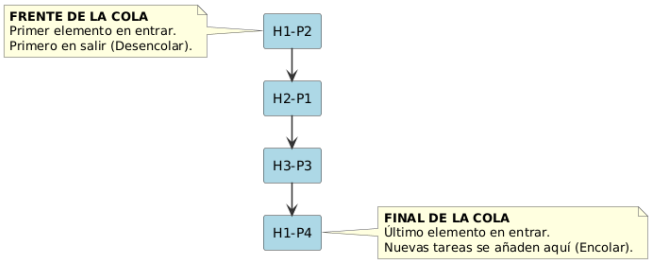

Figura 1. Representación gráfica del TDA Cola utilizado para gestionar el plan de riego.

Fuente: elaboración propia.

Esta estructura es la clave del algoritmo de simulación, pues garantiza que el orden de las operaciones se respete rigurosamente.

c. Algoritmo de Simulación por "Ticks" de Tiempo.

El algoritmo de simulación es el corazón del proyecto. En lugar de procesar las tareas de forma puramente secuencial, emula un entorno concurrente donde el tiempo avanza en unidades discretas ("ticks" o segundos). El proceso en cada segundo es el siguiente:

1. Se identifica la tarea prioritaria, que es la que se encuentra al frente de la Cola.
2. Se itera sobre todos los drones activos. Cada dron toma una decisión independiente:
    - Si su próxima tarea es la prioritaria y está en la posición correcta, su acción es Regar.
    - Si no está en la posición de su próxima tarea, su acción es Moverse (Adelante o Atrás).
    - Si está en posición, pero no es su turno de regar, su acción es Esperar.
3. Si un dron ejecutó la acción de Regar, la tarea prioritaria se desencola de la Cola.
4. Si un dron ejecutó la acción de Regar, la tarea prioritaria se desencola de la Cola.

Este ciclo se repite hasta que la Cola de tareas queda vacía.

d. Generación de Reportes y Archivos de Salida

La fase final consiste en consolidar los datos registrados durante la simulación. El sistema genera múltiples salidas para el análisis:

- **Reportes en la Interfaz Web:** La aplicación, construida con **Flask**, muestra dinámicamente tablas con la secuencia de acciones por segundo y el consumo final de recursos por dron.
- Reporte HTML General: Se genera una página HTML estática que resume los resultados de todos los planes de riego de todos los invernaderos cargados.
- Archivo XML de Salida**:** Se construye un archivo salida.xml que contiene un reporte detallado y estructurado de cada simulación, cumpliendo con el formato especificado en el enunciado. La Tabla I muestra un extracto de cómo se estructuran los datos de salida.

Ejemplo de la estructura de instrucciones en el archivo salida.xml.

Tiempo(s)	Dron DR01	Dron DR02
1	Adelante(H1P1)	Adelante(H2P1)
2	Adelante(H1P2)	Regar

Fuente: elaboración propia.

Conclusiones

El desarrollo de este proyecto demuestra que los simuladores de software son herramientas poderosas para la planificación y optimización de sistemas logísticos complejos, como la automatización agrícola. La solución no solo cumple con su objetivo funcional de analizar la eficiencia de los planes de riego, sino que también valida la importancia del diseño de software basado en TDA y POO para crear sistemas modulares y escalables.

La principal postura que se desea transmitir es que la simulación previa permite identificar cuellos de botella y optimizar operaciones, lo que puede generar un impacto económico directo al reducir el tiempo operativo y el consumo de recursos en el despliegue real. Queda como tema abierto a la reflexión cómo este simulador podría extenderse para incluir variables estocásticas, como fallos de drones o variaciones climáticas, para un análisis de riesgo más completo.

Referencias bibliográficas

Pallets Projects. (2025). *Flask Web Framework*. The Pallets Projects. Recuperado de https://flask.palletsprojects.com/en/stable/
.

Python Software Foundation. (2025). *Python Language Reference*. Python Software Foundation. Recuperado de https://www.python.org/

The Bootstrap Team. (2025). *Bootstrap CSS Framework*. The Bootstrap Team. Recuperado de https://getbootstrap.com/

Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 2do. Semestre 2025

Diagrama de Clase:

Figura 1. Diagrama de Clases del Sistema de Simulación

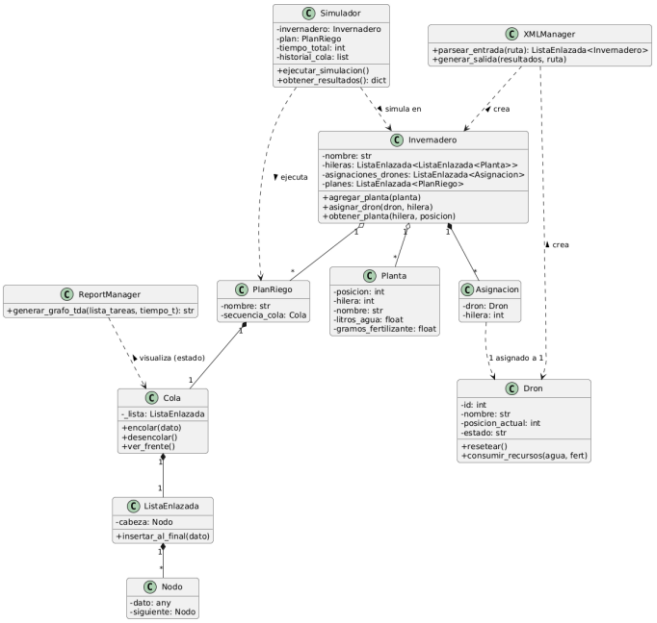

Diagrama de Actividades General.

Figura 2. Diagrama de Actividades del Proceso de Simulación

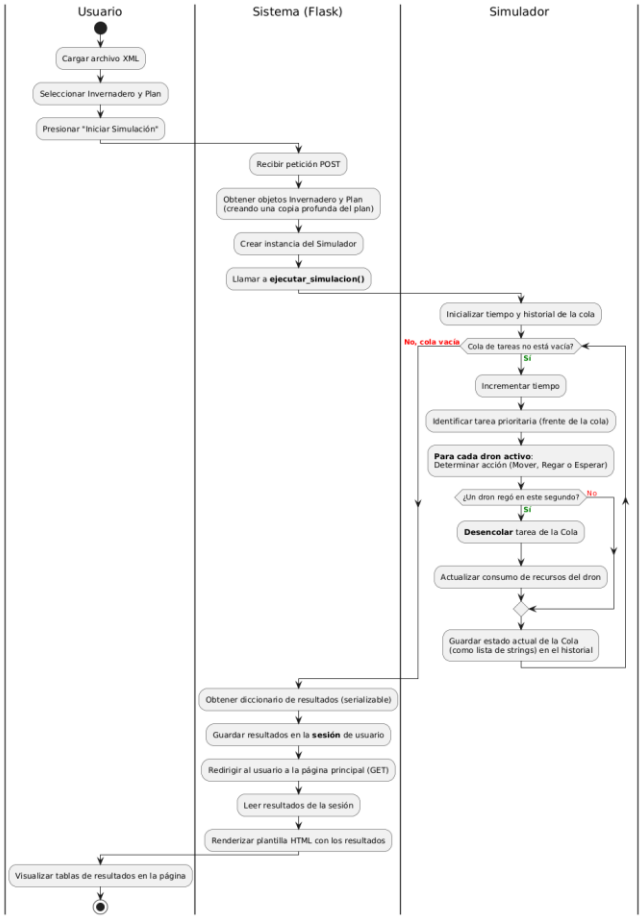

Diagrama de actividades del proceso de carga de archivo.

Figura 3. Diagrama de Actividades del Proceso de Carga de Archivo