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

DESARROLLO DE UNA PLATAFORMA WEB PARA LA SIMULACIÓN DE UN SISTEMA LOGÍSTICO MEDIANTE ARQUITECTURA CLIENTE–SERVIDOR

202300537 – Katherin Yasmin Miranda Hernández
202205097 – Juan Francisco Ramirez De Paz

Resumen

El presente ensayo describe el diseño e implementación de una plataforma web orientada a la simulación de un sistema logístico, desarrollada bajo una arquitectura cliente–servidor utilizando tecnologías modernas de desarrollo de software. El sistema, denominado LogiTrack, integra un backend basado en Java con Spring Boot y un frontend desarrollado en React, permitiendo la gestión de centros de distribución, rutas, mensajeros, paquetes y solicitudes de envío a partir de una configuración inicial cargada mediante archivos XML. La solución propuesta responde a la necesidad de comprender y modelar procesos logísticos reales, enfatizando el uso de programación orientada a objetos y estructuras de datos en memoria, prescindiendo del uso de bases de datos tradicionales. Desde una perspectiva técnica, el sistema implementa validaciones, control de estados y una cola de prioridad para el procesamiento de solicitudes, garantizando coherencia y consistencia operativa. Asimismo, se analizan los impactos de la solución en el ámbito académico, destacando su valor como herramienta didáctica para el aprendizaje de arquitecturas web, diseño de APIs REST y modelado de dominios complejos. Finalmente, se presentan conclusiones relacionadas con la importancia de la separación de responsabilidades y la escalabilidad del sistema

Palabras clave
logística, API REST, React, Spring Boot, programación orientada a objetos.

Abstract

This essay describes the design and implementation of a web platform aimed at simulating a logistics system, developed under a client–server architecture using modern software development technologies. The system, called LogiTrack, integrates a backend based on Java with Spring Boot and a frontend developed in React, enabling the management of distribution centers, routes, couriers, packages, and shipping requests from an initial configuration loaded through XML files. The proposed solution addresses the need to understand and model real logistics processes, emphasizing the use of object-oriented programming and in-memory data structures, avoiding traditional database systems. From a technical perspective, the system implements validations, state control, and a

priority queue for request processing, ensuring operational consistency and coherence. Additionally, the academic impact of the solution is analyzed, highlighting its value as a didactic tool for learning web architectures, REST API design, and complex domain modeling. Finally, conclusions are presented regarding the importance of responsibility separation and system scalability.

Keywords
logistics, REST API, React, Spring Boot, object-oriented programming.

Introducción

La logística constituye uno de los pilares fundamentales en el funcionamiento de empresas modernas, especialmente en contextos donde la eficiencia en la gestión de recursos, transporte y distribución resulta determinante para la competitividad organizacional. En este escenario, el uso de sistemas informáticos para modelar y simular procesos logísticos se ha convertido en una herramienta clave tanto en el ámbito empresarial como académico. El presente ensayo aborda el desarrollo de una plataforma web orientada a la simulación de un sistema logístico, con el propósito de aplicar conceptos de programación orientada a objetos, arquitecturas cliente–servidor y servicios web. La solución planteada permite gestionar de forma integral los principales componentes de un sistema logístico, proporcionando una base sólida para el análisis y comprensión de su funcionamiento. A lo largo del ensayo se describen los fundamentos técnicos del sistema, las decisiones de diseño adoptadas y su relevancia dentro del contexto formativo de la Ingeniería en Ciencias y Sistemas.

Desarrollo del tema

a. Arquitectura del sistema y separación de responsabilidades

El sistema LogiTrack fue diseñado bajo una arquitectura cliente–servidor claramente definida, en la cual el backend y el frontend cumplen roles específicos y complementarios. El backend, desarrollado con Java y el framework Spring Boot, actúa como el núcleo lógico del sistema, siendo responsable de procesar la información, aplicar las reglas de negocio y mantener la coherencia de los datos en memoria. Por su parte, el frontend, implementado en React, se encarga exclusivamente de la presentación de la información y de la interacción con el usuario final, comunicándose con el backend a través de una API REST sobre el protocolo HTTP.

Esta separación de responsabilidades permite una mayor mantenibilidad y escalabilidad del sistema, ya que los cambios en la lógica de negocio no afectan directamente a la interfaz gráfica y viceversa. Además, este enfoque facilita el consumo del backend por parte de otros clientes, como herramientas de prueba o servicios externos, reforzando el carácter modular de la solución.

b. Modelado del dominio y programación orientada a objetos

Uno de los aspectos centrales del proyecto es el modelado del dominio logístico mediante programación orientada a objetos. El sistema identifica entidades clave como centros de distribución, rutas, mensajeros, paquetes y solicitudes, cada una representada por clases que

encapsulan atributos y comportamientos específicos. Este enfoque permite reflejar de manera fiel las relaciones y dependencias existentes en un sistema logístico real, tales como la asignación de mensajeros a centros o la asociación de paquetes a solicitudes de envío.

El uso de enumeraciones para representar los estados de mensajeros, paquetes y solicitudes contribuye a garantizar transiciones válidas y evita inconsistencias en el flujo del sistema. Asimismo, la implementación de servicios especializados para cada entidad refuerza el principio de responsabilidad única, promoviendo un diseño limpio y estructurado.

c. Gestión de datos en memoria y procesamiento de XML

En concordancia con las restricciones del proyecto, el sistema prescinde del uso de bases de datos tradicionales, optando por el almacenamiento de la información en estructuras de datos en memoria. Esta decisión enfatiza el manejo directo de colecciones y la implementación de mecanismos propios para la validación, búsqueda y actualización de datos, fortaleciendo las competencias del estudiante en estructuras de datos y lógica de programación.

La carga inicial del sistema se realiza mediante un archivo XML que contiene la configuración completa del dominio. El backend se encarga de validar la estructura del archivo, detectar errores y poblar las entidades correspondientes, permitiendo que el sistema inicie con un estado coherente. De igual manera, el sistema ofrece la posibilidad de generar un XML de salida que resume el estado final de la simulación, incluyendo estadísticas relevantes y resultados de las operaciones realizadas.

d. Procesamiento de solicitudes y uso de estructuras de prioridad

El procesamiento de solicitudes de envío constituye uno de los componentes más relevantes del sistema. Para ello, se implementa una cola de prioridad que permite atender primero aquellas solicitudes con mayor valor de prioridad, simulando escenarios reales donde ciertos envíos requieren atención preferencial. Este mecanismo implica la validación de múltiples condiciones, como la disponibilidad de mensajeros, la existencia de rutas y la capacidad de carga, garantizando que cada solicitud se procese de manera lógica y consistente.

El cambio de estados de paquetes y mensajeros durante el procesamiento de solicitudes refuerza la idea de un sistema dinámico, en el cual las acciones del usuario impactan directamente en el estado global del sistema. Este comportamiento resulta especialmente valioso desde una perspectiva académica, al permitir observar de forma práctica los efectos de las decisiones de diseño y las reglas de negocio implementadas.

Conclusiones

El desarrollo del sistema LogiTrack evidencia la importancia de aplicar principios sólidos de ingeniería de software en la construcción de soluciones informáticas complejas. La utilización de una arquitectura cliente–servidor, junto con un modelado adecuado del dominio y una clara separación de responsabilidades, permite construir un sistema coherente, escalable y fácil de mantener. Asimismo, el uso de programación orientada a objetos y estructuras de datos en memoria fortalece las competencias técnicas del estudiante, promoviendo una comprensión profunda de los fundamentos de la disciplina. Desde el punto de vista académico, la solución desarrollada representa una herramienta valiosa para el aprendizaje práctico de arquitecturas web y diseño de APIs REST. Finalmente, este proyecto demuestra que es posible simular procesos logísticos reales de manera efectiva, sentando las bases para futuras extensiones y mejoras del sistema.

Referencias bibliográficas

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley.

Sommerville, I. (2016). Software Engineering (10th ed.). Pearson Education.

Oracle. (2023). Java Platform, Standard Edition Documentation.

Spring. (2024). Spring Boot Reference Documentation.

React. (2024). React Official Documentation.

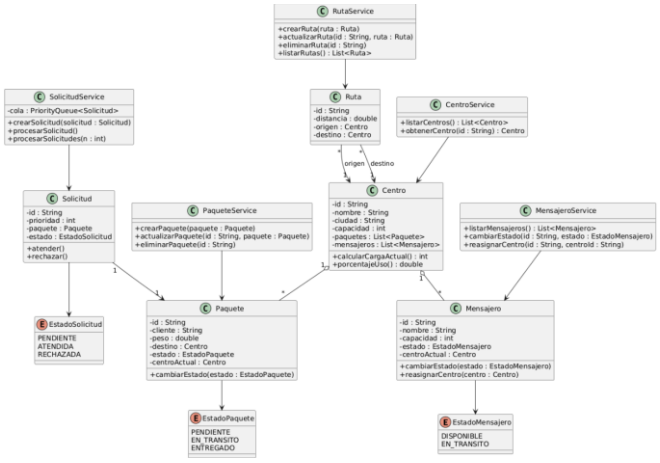

Figura 1. Diagrama de clases.

Fuente: elaboración propia

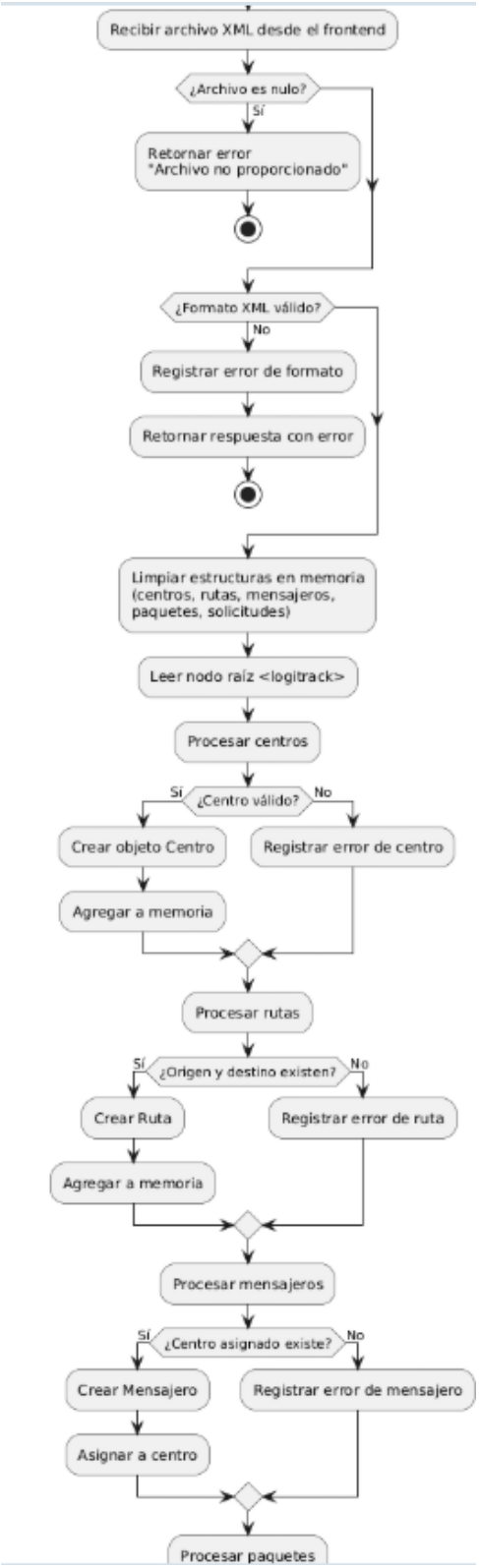

Figura 11. Diagrama de flujo carga del sistema inicial.

Fuente: elaboración propia