

---

## PROYECTO 1

---

202401198 – José Bernardo Méndez Juárez  
202405532 – Marcos Manuel Chip Castillo

### Resumen

El presente ensayo aborda el desarrollo de un sistema de gestión de centros de datos utilizando estructuras de datos dinámicas implementadas manualmente, específicamente listas doblemente enlazadas. El sistema permite administrar centros de datos, máquinas virtuales y contenedores, así como calcular el consumo y disponibilidad de recursos como CPU, memoria RAM y almacenamiento. La propuesta resulta vigente debido al crecimiento del uso de infraestructuras virtualizadas y servicios en la nube a nivel nacional e internacional. Desde una perspectiva técnica, se analizan los beneficios de emplear Tipos de Datos Abstractos personalizados en lugar de estructuras nativas del lenguaje, fortaleciendo la comprensión de la gestión de memoria y la organización de datos. A nivel académico, el proyecto promueve el pensamiento lógico y estructurado, mientras que a nivel social y económico resalta la importancia de optimizar recursos computacionales. Como conclusión, se evidencia que el uso correcto de estructuras dinámicas permite construir sistemas escalables, eficientes y alineados con problemas reales de la ingeniería en sistemas.

### Palabras clave

Centros de datos, listas enlazadas, máquinas virtuales, gestión de recursos, virtualización.

### Abstract

*This essay presents the development of a data center management system using manually implemented dynamic data structures, specifically doubly linked lists. The system manages data centers, virtual machines, and containers, while calculating resource consumption and availability such as CPU, RAM, and storage. This proposal is relevant due to the increasing use of virtualized infrastructures and cloud services both nationally and internationally. From a technical perspective, the benefits of using custom Abstract Data Types instead of native language structures are analyzed, strengthening the understanding of memory management and data organization. Academically, the project encourages logical and structured thinking, while from a social and economic standpoint it highlights the importance of optimizing computational resources. In conclusion, the correct use of dynamic structures enables the development of scalable, efficient systems aligned with real-world engineering problems.*

### Keywords

*Data centers, linked lists, virtual machines, resource management, virtualization.*

## Introducción

La virtualización y la administración eficiente de recursos computacionales son pilares fundamentales en la infraestructura tecnológica moderna. Los centros de datos concentran grandes cantidades de recursos físicos que deben ser gestionados de forma óptima para garantizar disponibilidad, escalabilidad y rendimiento. En este contexto, el uso de estructuras de datos adecuadas resulta clave para modelar y administrar dichos sistemas. El presente ensayo tiene como objetivo analizar el desarrollo de un sistema de gestión de centros de datos implementado mediante listas doblemente enlazadas, sin recurrir a estructuras nativas del lenguaje Python. A través de este enfoque, se busca fortalecer la comprensión de los Tipos de Datos Abstractos y su aplicación práctica en problemas reales de ingeniería. El trabajo explora cómo estas estructuras permiten organizar centros, máquinas virtuales y contenedores, así como calcular dinámicamente el uso y disponibilidad de recursos.

## Desarrollo del tema

El desarrollo del proyecto se centró en la implementación de un sistema de gestión de centros de datos utilizando **Tipos de Datos Abstractos (TDAs)** construidos manualmente, cumpliendo con la restricción de no emplear estructuras nativas avanzadas del lenguaje. Para ello, se diseñó una arquitectura basada en **listas doblemente enlazadas**, colas y nodos, que permiten modelar de forma jerárquica los elementos principales del sistema: centros de datos, máquinas virtuales, contenedores y solicitudes.

- **Estructura de datos y jerarquía del sistema**

La estructura principal del sistema parte de una **lista doblemente enlazada de centros de datos**, donde cada nodo contiene un objeto de tipo **Centro**. Cada centro administra internamente su propia lista doblemente enlazada de **máquinas virtuales**, y a su vez, cada máquina virtual mantiene una lista de **contenedores**. Esta organización jerárquica permite representar de forma clara la relación:

Centro → Máquinas Virtuales → Contenedores

El uso de listas doblemente enlazadas facilita operaciones como inserción, eliminación, búsqueda y recorrido secuencial, sin depender de librerías externas, reforzando el aprendizaje de estructuras dinámicas implementadas desde cero.

- **Gestión de recursos**

Uno de los aspectos más importantes del desarrollo fue el control de recursos de cada centro de datos. Para ello, cada centro almacena sus recursos originales (CPU, RAM y almacenamiento) y mantiene atributos adicionales para los recursos usados y disponibles. A través de los métodos `CalcularConsumoCentro()` y `CalcularRecursosDisponibles()`, el sistema calcula dinámicamente:

- Recursos consumidos por las máquinas virtuales asociadas.
- Recursos restantes disponibles para nuevas asignaciones.

Este enfoque evita la modificación directa de los valores originales y garantiza que los cálculos sean consistentes en cualquier momento del programa, lo cual es fundamental para validar la creación o migración de máquinas virtuales.

- **Funcionalidades del menú de centros de datos**

Dentro de la gestión de centros de datos se implementaron funciones clave como:

- **Listado de centros:** recorre la lista principal y muestra información detallada de cada centro, incluyendo porcentajes de uso de recursos.
- **Búsqueda de centro por ID:** permite localizar un centro específico y visualizar su estado actual de recursos.
- **Centro con más recursos disponibles:** mediante un recorrido secuencial, se identifica el centro con mayor capacidad disponible, el cual resulta útil para asignaciones automáticas de máquinas virtuales.

Estas funciones demuestran el uso práctico de recorridos en listas enlazadas y la comparación de atributos calculados dinámicamente.

- **Gestión de máquinas virtuales**

El sistema permite buscar máquinas virtuales por ID de forma global, recorriendo todos los centros y sus respectivas listas de máquinas virtuales. Además, se puede listar todas las máquinas virtuales pertenecientes a un centro específico, mostrando información relevante como sistema operativo, recursos asignados, estado, IP y cantidad de contenedores desplegados.

Para la creación de máquinas virtuales, se implementó un proceso de validación que verifica si el centro seleccionado cuenta con los recursos suficientes antes de realizar la inserción. Esto se logra mediante la función `validarRecursosDisponibles`, la cual previene asignaciones inválidas y mantiene la integridad del sistema.

- **Migración de máquinas virtuales**

La migración de máquinas virtuales entre centros es una de las funcionalidades más complejas del sistema. Este proceso incluye:

1. Validación de la existencia del centro origen y destino.
2. Eliminación controlada de la máquina virtual del centro origen.
3. Verificación de recursos disponibles en el centro destino.
4. Inserción de la máquina virtual en el centro destino o reversión en caso de falta de recursos.

Este flujo garantiza que no se pierdan datos y que los recursos se mantengan consistentes, incluso ante escenarios de error.

- **Gestión de solicitudes mediante colas**

Para la administración de solicitudes se utilizó una **cola implementada manualmente**, donde cada nodo representa una solicitud con atributos como prioridad, tipo y recursos requeridos. El sistema permite:

- Agregar nuevas solicitudes a la cola.
- Procesar la solicitud de mayor prioridad.
- Procesar múltiples solicitudes de forma secuencial.
- Visualizar el estado actual de la cola.

La selección de solicitudes por prioridad refuerza el uso de recorridos y comparaciones dentro de una estructura dinámica, aplicando conceptos fundamentales de colas y planificación.

- **Procesamiento automático de solicitudes**

Finalmente, el sistema integra la gestión de solicitudes con la creación automática de máquinas virtuales. Dependiendo del tipo de solicitud (Deploy o Backup), se crea una máquina virtual con un estado específico y se asigna al centro con mayor disponibilidad de recursos. Esta integración demuestra cómo los diferentes módulos del sistema interactúan entre sí de forma coherente.

En conjunto, el desarrollo del proyecto evidencia la aplicación práctica de estructuras de datos, validaciones lógicas y diseño modular, cumpliendo con los requerimientos establecidos y fortaleciendo la comprensión de los TDAs en un contexto realista.

## **Conclusiones**

El desarrollo de este proyecto permitió consolidar los conocimientos teóricos sobre Tipos de Datos Abstractos mediante su aplicación práctica en un sistema funcional. La implementación manual de listas doblemente enlazadas y colas facilitó una comprensión más profunda del manejo de estructuras dinámicas, así como de sus ventajas frente a estructuras estáticas.

Asimismo, la gestión de recursos de los centros de datos demostró la importancia de realizar cálculos dinámicos y validaciones constantes para garantizar la consistencia del sistema. Funciones como la asignación automática de máquinas virtuales al centro con mayor disponibilidad y la migración controlada entre centros reflejan situaciones reales de administración de infraestructura.

Finalmente, la integración del manejo de solicitudes con prioridades reforzó el uso de colas y recorridos secuenciales, permitiendo simular procesos de planificación y atención eficiente. En conclusión, el proyecto cumple con los objetivos planteados, fortalece las habilidades de diseño estructurado y sienta bases sólidas para el desarrollo de sistemas más complejos en el futuro.

## Referencias bibliográficas

- Aho, A. V., Hopcroft, J. E., & Ullman, J. D. (2008). *Estructuras de datos y algoritmos*. Pearson Educación.
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2014). *Introducción a los algoritmos* (3.<sup>a</sup> ed.). McGraw-Hill.
- Deitel, P., & Deitel, H. (2016). *Cómo programar en Python*. Pearson Educación.
- Joyanes Aguilar, L. (2011). *Fundamentos de programación: Algoritmos, estructuras de datos y objetos*. McGraw-Hill.
- Joyanes Aguilar, L. (2011). *Fundamentos de programación: Algoritmos, estructuras de datos y objetos*. McGraw-Hill.