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

PROYECTO 1. CIOUDSYNC.

202308208 – Rodrigo Andres Guay Minera
202002019 – Oliverio Macz Coc

Resumen

El presente trabajo documenta la implementación de CloudSync, una plataforma de simulación para la gestión distribuida de recursos en la nube. El sistema aborda la problemática de la administración eficiente de Centros de Datos, Máquinas Virtuales (VMs) y Contenedores, así como el manejo de solicitudes de servicio con diversas prioridades.

El proyecto demuestra cómo la programación orientada a objetos(POO) y el diseño modular pueden crear un sistema escalable basado en Python y la estructura de archivos XML con la aplicación de estructuras de datos. Así mismo la generación de archivos de salida con visualización gráfica del estado del sistema mediante Graphviz.

La solución más acertada según las instrucciones es la utilización de estructura de datos no nativas de Python, incluyendo listas simples enlazadas para la jerarquía de recursos(Centros de Datos,Máquinas Virtuales,contenedores) y una cola de prioridad para la administración de solicitudes.

La principal conclusión es que la selección adecuada de estructura de datos es fundamental para optimizar las operaciones de gestión,la migración y balanceo en carga en entornos virtuales complejos.

Palabras clave

Estructuras de Datos, Cloud Computing,Python, Graphviz, XML.

Abstract

This paper documents the implementation of CloudSync, a simulation platform for distributed cloud resource management. The system addresses the challenges of efficiently managing data centers, virtual machines (VMs), and containers, as well as handling service requests with varying priorities.

The project demonstrates how object-oriented programming (OOP) and modular design can create a scalable system based on Python and XML file structures, using data structures. It also demonstrates the generation of output files with graphical visualization of the system's status using Graphviz.

The most effective solution, according to the instructions, is the use of non-native Python data structures, including singly linked lists for the resource hierarchy (data centers, virtual machines, containers) and a priority queue for request management.

The main conclusion is that the appropriate selection of data structures is fundamental for

optimizing management operations, migration, and load balancing in complex virtual environments.

Keywords
Data Structures, Cloud Computing, Python, Graphviz, XML.

Introducción

CloudSync es una plataforma desarrollada para simular la gestión de recursos en la nube: El cual fue desarrollado únicamente con el uso y diseño manual de estructuras de datos dinámicos y lecturas de datos mediante xml.

Uno de los objetivos fundamentales es emular el comportamiento real de una plataforma de administración, enfrentando desafíos cítricos como la asignación dinámica de recursos y la priorización de solicitudes.

La interacción con el sistema será únicamente en consola y con el uso de XML para la obtención de los datos, los cuales definirán la estructura inicial de la plataforma.

Desarrollo del tema

El sistema CloudSync modela la infraestructura de la nube utilizando una composición jerárquica de objetos gestionada exclusivamente con la lista simple enlazada.

Nivel 1 Centros de Datos

- Se utiliza una lista para almacenar los objetos y atributos en la clase “CentroDatos”.
- Las operaciones de gestión (listar, buscar, ver centro con mayores recursos) se realizan mediante recorridos de esta lista.

Nivel 2 Máquinas Virtuales

- Cada objeto “CentroDatos” contiene su propia Lista Simple Enlazada para las “MaquinaVirtual”.
- La migración de VMs entre centros implica el uso de un algoritmo de búsqueda (buscar_dato_por_id en la lista global) y una doble operación de eliminación del centro de origen y adición al centro de destino, asegurando la validación de recursos en el destino.

Nivel 3 Contenedores

- Cada objeto MaquinaVirtual contiene una lista simple enlazada de Contenedor.La gestión de contenedores se realiza accediendo primero a la VM, y luego manipulando la lista anidada

Nombre	Categoría
CentroDatos	Clase Modelo
Contenedor	Clase Modelo
Instrucciones	Clase Modelo
MaquinaVirtual	Clase Modelo
Solicitud	Clase Modelo

Tabla 1. Modelos

Fuente: elaboración propia

a.Gestión de solicitudes y Cola de Prioridad

- Estructura: La clase “ControladorSolicitudes” utiliza esta Cola de Prioridades, donde los objetos Solicitud son encolados basándose en su atributo prioridad (según del 1 a 10).

- Procesamiento: El método "procesar_siguiente()" garantiza que la solicitud con la prioridad más alta (el elemento en el frente de la cola) sea atendida primero.

b.Carga y Reportes XML

- Carga: el módulo "Lector" se encarga de parsear el archivo "entrada.xml" extrayendo la configuración inicial y las instrucciones operacionales las cuales son delegadas a los controladores respectivos.
- Generación: el sistema genera un archivo XML de salida que registra el resultado de las operaciones realizadas así como la ejecución de las instrucciones, sirviendo como un registro de auditoría.

c.Reportes Gráficos con Graphviz

- Reportes de centros de datos: Visualiza la lista simple de centros mostrando la capacidad total de recursos de cada nodo
- Reporte de VMs por centro: muestra la relación de jerarquía y el listado de las máquinas virtuales alojadas en un centro seleccionado incluyendo sus recursos asignados.
- Reporte de contenedores por VM: Representa la composición listando los contenedores desplegados en una VM específica destacando su estado mediante distintos colores.
- Reporte de la cola de solicitudes: muestra el orden de elementos de la cola prioridad donde el color del nodo representa la urgencia del mismo, más exactamente en tonalidades del azul facilitando la identificación de las tareas prioritarias.

Conclusiones

El uso y la implementación de listas enlazadas, sin depender de las estructuras nativas del lenguaje de programación python nos permitió un control bastante preciso sobre la gestión de los centros de datos, máquinas virtuales y sobre todo en la ejecución de las solicitudes. Esto nos demuestra que la elección de las estructuras al momento de iniciar cualquier proyecto es crítico y fundamental para los resultados queremos obtener.

El desarrollo de los algoritmos adecuados también evidenció la eficiencia que puede llegar a tener el uso de las listas enlazadas.

También se logró demostrar que un sistema de simulación de gestiones en la nube no solo debe almacenar recursos si no también debe ser capaz de tomar decisiones dinámicamente como las migraciones de máquinas virtuales o la ejecución de solicitudes.

Referencias bibliográficas

Estructura de Datos y Algoritmos
(Hernandez, Lázaro, Dormido, Ros)

Fundamentos de Programación: Algoritmos, Estructura de Datos (Luis Joyanes Aguilar)

Nombre	Categoría
ControladorCentros	Clase Controlador
ControladorContenedor	Clase Controlador
ControladorInstrucciones	Clase Controlador
ControladorMaquinaVirtual	Clase Controlador
ControladorSolicitud	Clase Controlador

Tabla 2. Controladores

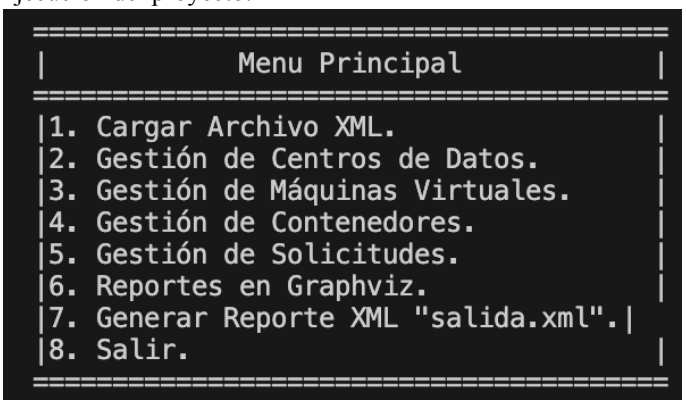
Fuente: elaboración propia

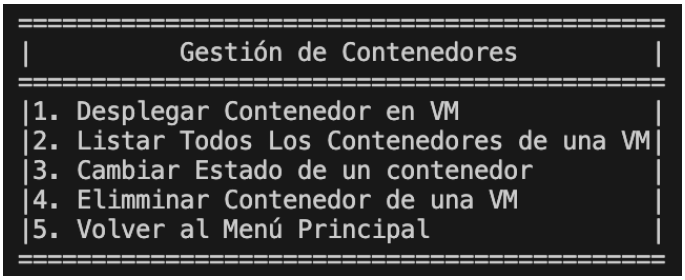
Nombre	Categoría
ListaPrioridades	Clase Lista
ListaSimpleEnlazada	Clase Lista
Nodo	Clase Nodo

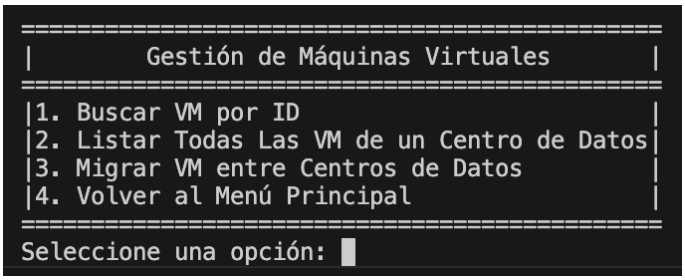
Tabla 2. Listas

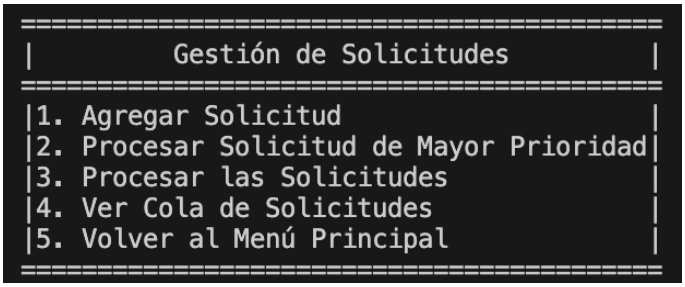
Fuente: elaboración propia

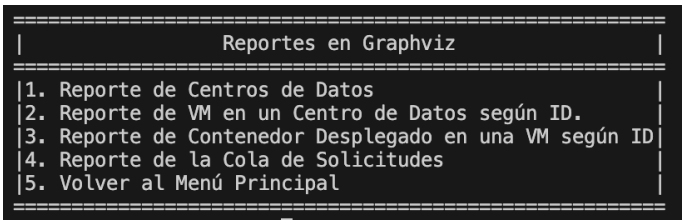
Ejecucion del proyecto:











Repositorio de Github:https://github.com/rguaynaa/IPC2DIC_Proyecto1_11.git