

---

## IPC2 - MARKET

---

202200341 – Rebeca Ayline Torres Del Cid  
202112109 – Carlos David De León Barrios

### Resumen

La aplicación de escritorio IPC2 market es un app Commerce que facilita a los usuarios realizar compras de productos para su consumo diario. Cuenta con una interfaz intuitiva y amigable. Se distingue entre dos tipos de usuarios: Administrador y Comprador. La carga de información se limita a archivos XML y se almacena en memoria dinámica. Los usuarios deben estar registrados por el administrador para acceder al sistema. El único usuario quemado en el sistema es el Administrador, cuyas credenciales son Usuario: AdminIPC2 y Contraseña: IPC2VJ2024.

### Palabras clave

- AppCommerce: Aplicación que combina comercio electrónico y funcionalidades móviles o de escritorio para compras en línea.
- Programación Orientada a Objetos (POO): Paradigma de programación basado en objetos para modularidad y reutilización de código.
- Listas Enlazadas: Estructuras de datos con nodos conectados por referencias para operaciones eficientes como inserción y eliminación.
- Archivos XML: Formato de almacenamiento jerárquico y legible para intercambio de datos entre sistemas.
- Memoria Dinámica: Asignación de memoria durante la ejecución para estructuras flexibles y eficiencia en el uso de recursos.

### Abstract

*The IPC2 market desktop application is an app Commerce platform that enables users to make purchases of daily consumption products. It features a user-friendly and intuitive interface. There are two user types: Administrator and Buyer. Information loading is limited to XML files and stored using dynamic memory. Users must be registered by the Administrator to access the system. The only preloaded user is the Administrator, with credentials Username: AdminIPC2 and Password: IPC2VJ2024.*

### Keywords

- *AppCommerceApplication combining e-commerce and mobile/desktop functionalities for online shopping.*
- *Object-Oriented Programming (OOP)\*\*: Programming paradigm based on objects for modularity and code reusability.*
- *Linked ListsData structures with nodes connected by references for efficient operations like insertion and deletion.*
- *XML FilesHierarchical and readable storage format for data exchange between systems*
- *Dynamic MemoryAllocation of memory during runtime for flexible structures and efficient resource usage.*

Introducción

En el entorno actual, la tecnología juega un papel fundamental en todos los aspectos de nuestra vida, incluyendo la forma en que realizamos compras y manejamos información. La convergencia entre el comercio electrónico y las aplicaciones móviles o de escritorio ha dado lugar a una nueva era de conveniencia y accesibilidad para los consumidores. En este contexto, el desarrollo de soluciones integrales como AppCommerce ha ganado relevancia, ofreciendo a los usuarios una plataforma intuitiva para realizar compras en línea de manera eficiente.

Por otro lado, la Programación Orientada a Objetos (POO) sigue siendo un pilar fundamental en el mundo del desarrollo de software, permitiendo la creación de sistemas modulares y fácilmente mantenibles. La combinación de estos elementos con tecnologías como XML para el intercambio de datos y la gestión de la memoria de manera dinámica, representa el panorama actual de la informática aplicada al comercio y la programación.

Desarrollo del tema

La Evolución del AppCommerce en el Comercio Electrónico Moderno

AppCommerce, la fusión del comercio electrónico con aplicaciones móviles y de escritorio, representa un cambio transformador en el comportamiento del consumidor y las estrategias empresariales. Este ensayo profundiza en los fundamentos teóricos, situaciones contextuales y perspectivas disciplinarias que definen la evolución del AppCommerce en el paisaje digital contemporáneo.

El Auge del AppCommerce

El advenimiento de los teléfonos inteligentes y la proliferación de la conectividad a internet han allanado el camino para el ascenso del AppCommerce. Las aplicaciones móviles ofrecen una conveniencia, personalización y accesibilidad sin precedentes, revolucionando la forma en que los consumidores interactúan con las empresas.

Programación Orientada a Objetos (POO) en AppCommerce

Detrás de la experiencia de usuario fluida del AppCommerce se encuentra el sólido marco de la Programación Orientada a Objetos (POO). Los principios de POO facilitan el desarrollo modular, la reutilización de código y la escalabilidad, elementos esenciales para construir plataformas de compras sofisticadas basadas en aplicaciones.

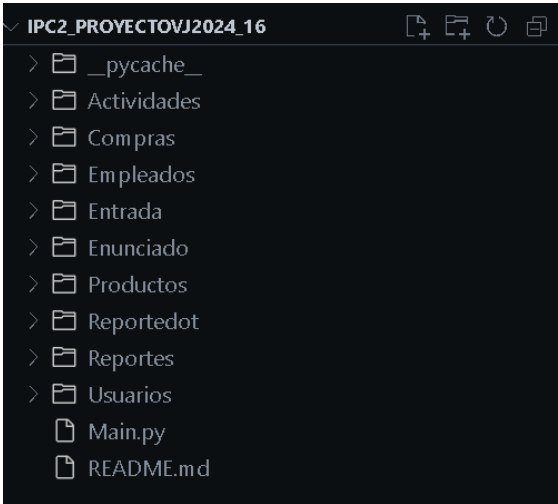
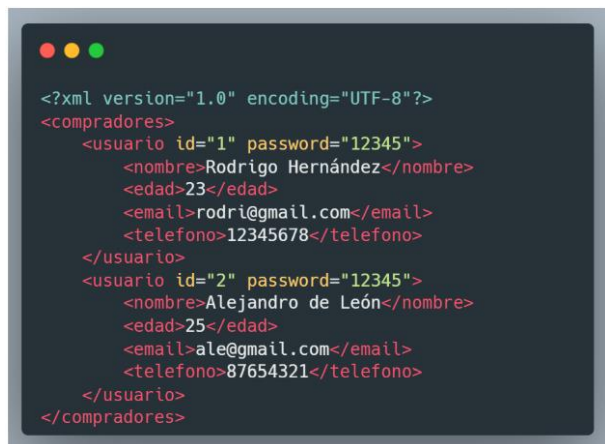


Figura 1. Estructura del proyecto, estructura POO por carpetas.

Fuente: Elaboración Propia, 2024.

## Intercambio de Datos con XML en AppCommerce

XML (Lenguaje de Marcado Extensible) desempeña un papel crucial en el AppCommerce al permitir el intercambio estructurado de datos entre sistemas. Su formato jerárquico y legibilidad aseguran un flujo de información eficiente, respaldando transacciones sin problemas y experiencias personalizadas.



```
<?xml version="1.0" encoding="UTF-8"?>
<compradores>
  <usuario id="1" password="12345">
    <nombre>Rodrigo Hernández</nombre>
    <edad>23</edad>
    <email>rodri@gmail.com</email>
    <telefono>12345678</telefono>
  </usuario>
  <usuario id="2" password="12345">
    <nombre>Alejandro de León</nombre>
    <edad>25</edad>
    <email>ale@gmail.com</email>
    <telefono>87654321</telefono>
  </usuario>
</compradores>
```

Figura 2. Estructura del archivo XML de entrada de usuarios.

Fuente: Elaboración Propia, 2024.

## Gestión de Memoria Dinámica en AppCommerce

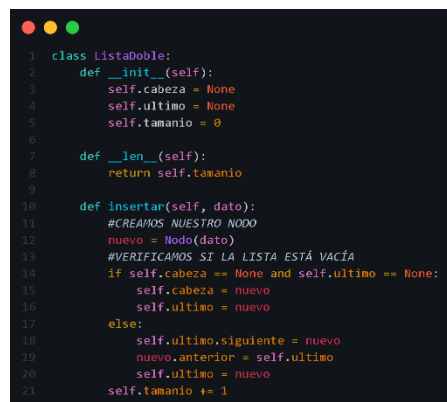
La gestión de memoria dinámica en Python ha sido fundamental para optimizar la utilización de recursos en las aplicaciones de AppCommerce. Al utilizar estructuras de datos como listas simples, dobles, dobles enlazadas, circulares, pilas y colas, se han implementado estrategias eficientes de asignación y liberación de memoria. Estas estrategias contribuyen significativamente al rendimiento de la aplicación, asegurando una respuesta rápida, estabilidad operativa y escalabilidad para manejar cargas de trabajo variables.

En el caso de las listas simples, dobles y dobles enlazadas, la asignación dinámica de memoria permite agregar y eliminar elementos de manera flexible, adaptándose a las necesidades cambiantes de la aplicación. Las listas circulares, por su parte,

optimizan el acceso a elementos mediante referencias circulares, lo que mejora la eficiencia en la búsqueda y manipulación de datos.

Las pilas y colas, también implementadas con memoria dinámica, ofrecen un manejo eficiente de datos de manera secuencial. Las pilas siguen el principio LIFO (Last In, First Out), mientras que las colas siguen el principio FIFO (First In, First Out), lo que permite una gestión ordenada y rápida de las operaciones de inserción y extracción de elementos.

En conjunto, estas estructuras de datos optimizadas con gestión de memoria dinámica en Python aseguran un funcionamiento fluido y una experiencia positiva para el usuario en las aplicaciones de AppCommerce.



```
1 class ListaDoble:
2     def __init__(self):
3         self.cabeza = None
4         self.ultimo = None
5         self.tamano = 0
6
7     def __len__(self):
8         return self.tamano
9
10    def insertar(self, dato):
11        #CREAMOS NUESTRO NODO
12        nuevo = Nodo(dato)
13        #VERIFICAMOS SI LA LISTA ESTÁ VACÍA
14        if self.cabeza == None and self.ultimo == None:
15            self.cabeza = nuevo
16            self.ultimo = nuevo
17        else:
18            self.ultimo.siguiente = nuevo
19            nuevo.anterior = self.ultimo
20            self.ultimo = nuevo
21            self.tamano += 1
```

Figura 3. Manejo de memoria dinámica, ejemplo en lista doble

Fuente: Elaboración Propia, 2024.

## Conclusión

La evolución del AppCommerce señala un cambio de paradigma en el comercio minorista y el comercio electrónico. Al aprovechar la POO, XML, la gestión de memoria dinámica e interfaces de usuario innovadoras, las empresas pueden crear experiencias de compra inmersivas y personalizadas que resuenen con los consumidores modernos. Este ensayo destaca el papel integral de la tecnología, los principios de programación y la gestión de datos en la configuración del futuro del AppCommerce y el comercio electrónico en general.

Manejo de listas según la carga XML

• Usuarios

Una lista doble utilizada para almacenar usuarios en una aplicación suele ser una estructura de datos en la que cada elemento, en este caso cada usuario, está representado por un nodo que contiene información como el nombre, la edad, el correo electrónico y otros datos relevantes. Lo que hace que una lista sea doble es que cada nodo tiene dos referencias, una al nodo anterior y otra al siguiente, permitiendo así la navegación en ambas direcciones dentro de la lista. Esto permite operaciones eficientes como la inserción y eliminación de usuarios en cualquier posición de la lista, así como el recorrido de la lista tanto hacia adelante como hacia atrás, lo que puede ser útil en diversas funcionalidades de la aplicación, como la visualización y edición de datos de usuario.



Figura 4. Lista Doble

Fuente: Elaboración Propia, 2024.

• Empleados

Una lista circular utilizada para guardar empleados en una aplicación es una estructura de datos en la que los nodos que representan a cada empleado están enlazados de manera circular, es decir, el último nodo de la lista está enlazado con el primer nodo, formando un bucle. Esto permite un acceso continuo y eficiente a los datos de los empleados, ya que una vez que se llega al final de la lista, se vuelve al principio automáticamente.

En una lista circular de empleados, cada nodo contendría información como el código del empleado, su nombre completo y el puesto que desempeña. Al ser circular, esta estructura de datos facilita operaciones como la inserción y eliminación de empleados en cualquier punto de la lista sin necesidad de recorrer toda la lista, lo que ahorra

tiempo y recursos en la gestión de datos de los empleados dentro de la aplicación.

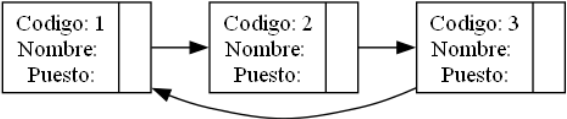


Figura 5. Lista Circular

Fuente: Elaboración Propia, 2024.

• Productos

Una lista doble circular para guardar productos en una aplicación es una estructura de datos donde cada producto se representa como un nodo enlazado a otros nodos de manera circular y doble. Esto significa que cada nodo tiene referencias tanto al nodo anterior como al siguiente en la lista, y el último nodo está enlazado al primero, cerrando así el círculo.

En esta lista, cada nodo de producto contendría información como el ID del producto, el precio, la descripción, la categoría, la cantidad disponible y la ruta de la imagen del producto. Al ser circular, esta estructura permite una navegación continua a través de los productos, facilitando operaciones como la inserción, eliminación y búsqueda de productos en cualquier posición de la lista sin necesidad de recorrer toda la lista.

La combinación de una lista doble y circular para almacenar productos en una aplicación proporciona una eficiente gestión de inventario y acceso rápido a la información de los productos, lo que contribuye a una mejor experiencia para los usuarios al realizar compras en línea.

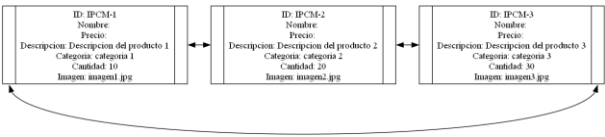


Figura 6. Lista Doblemente Circular

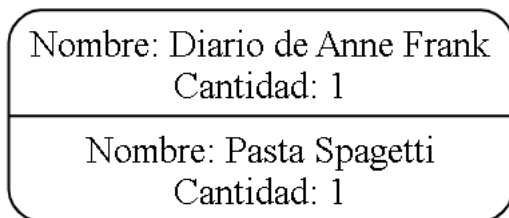
Fuente: Elaboración Propia, 2024.

## • Compras

Una pila utilizada para simular el "ver carrito" en una aplicación de comercio electrónico es una estructura de datos que sigue el principio LIFO (Last In, First Out), donde el último producto añadido al carrito es el primero en ser revisado o eliminado.

En esta pila, cada nodo representaría un producto añadido al carrito por el usuario, conteniendo información como el nombre del producto, la cantidad seleccionada y posiblemente otros detalles relevantes como el precio o la imagen. Cuando un usuario añade un producto al carrito, este se coloca en la cima de la pila. Para revisar los productos en el carrito, se puede recorrer la pila desde el último producto añadido hasta el primero.

Esta estructura es particularmente útil para gestionar el carrito de compras de manera sencilla y eficiente, permitiendo al usuario agregar productos, revisar el último producto añadido rápidamente y eliminar productos si es necesario. Además, al usar una pila, se garantiza que las operaciones de añadir y eliminar productos sean rápidas y fáciles de implementar.



Pila

Figura 3. Pila para el manejo de ver carrito de compras

Fuente: Elaboración Propia, 2024.

## Referencias bibliográficas

- **everance, C. R. (2016).** *Python for Everybody: Exploring Data in Python 3*. CreateSpace Independent Publishing Platform.
- **Lutz, M. (2013).** *Learning Python (5th ed.)*. O'Reilly Media.

- **Zelle, J. (2017).** *Python Programming: An Introduction to Computer Science (3rd ed.)*. Franklin, Beedle & Associates Inc.
- **Larman, C. (2004).** *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd ed.)*. Prentice Hall.
- **Budd, T. (2002).** *Understanding Object-Oriented Programming with Java (Updated Edition)*. Addison-Wesley.
- **McLaughlin, B., Pollice, G., & West, D. (2006).** *Head First Object-Oriented Analysis and Design*. O'Reilly Media.

## Anexos

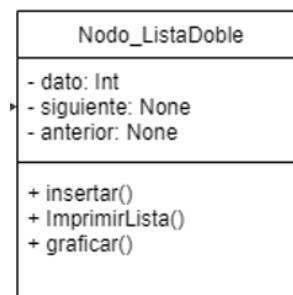


Figura 4. Modelo del tipo de dato Nodo\_ListaDoble.

Fuente: Elaboración Propia, 2024.

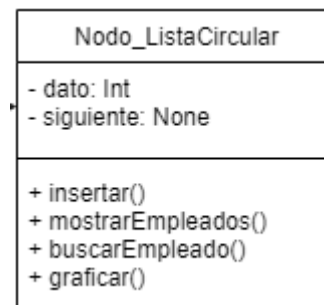


Figura 5. Modelo del tipo de dato Nodo\_Circular.

Fuente: Elaboración Propia, 2024.

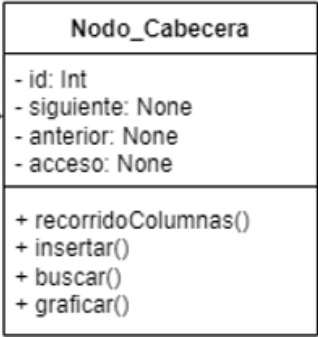


Figura 6. Modelo del tipo de dato *Nodo\_Cabecera*.  
Fuente: Elaboración Propia, 2024

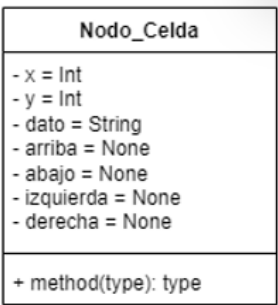


Figura 7. Modelo del tipo de dato *Nodo\_Celda*.  
Fuente: Elaboración Propia, 2024

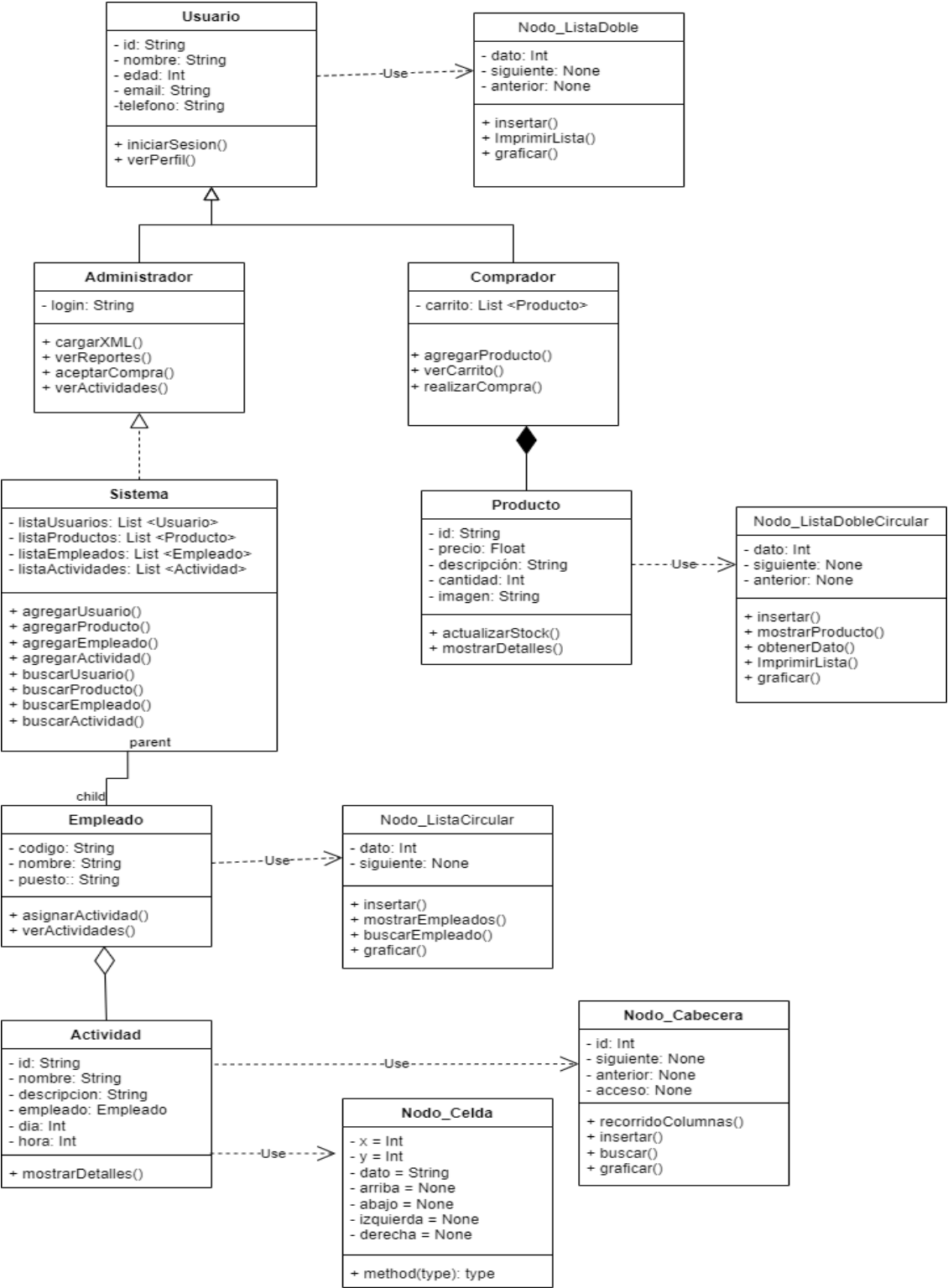


Figura 8. Diagrama de Clases del Sistema  
Fuente: Elaboración Propia, 2024