

UNIVERSIDAD LA SALLE

CARRERA DE INGENIERÍA DE SISTEMAS



Equipo de desarrollo:

- Ronald Choque Sillo
- Luis Angel Vertiz Blanco Tarifa
- Caleb Adrian Morales Alarcon
- Mattias Alexandre Duarte Aparicio

La Paz – Bolivia

INDICE

1. CAPITULO I.....	4
GENERALIDADES.....	4
1.1. Introducción.....	5
1.2. Justificación.....	5
1.3. Objetivos.....	6
a) Objetivo General.....	6
b) Objetivos Específicos.....	6
1.4. Arquitectura Hexagonal.....	6
a) Domino.....	7
b) Aplicación.....	7
c) Infraestructura.....	7
1.5. Principios de la Arquitectura Hexagonal.....	7
a) Separación de Preocupaciones.....	7
b) Independencia de Implementación.....	7
c) Inversión de dependencias.....	7
d) Testabilidad.....	8
e) Flexibilidad y Adaptabilidad.....	8
f) Reutilización de código.....	8
g) Modularidad.....	8
h) Simplicidad y Claridad.....	8
1.6. Ventajas y Desventajas.....	9
a) Ventajas.....	9
b) Desventajas.....	9
2. CAPITULO II.....	11
COMPONENTES DE LA ARQUITECTURA HEXAGONAL.....	11
2.1. Capa de Dominio (Nucleo).....	12
2.2. Capa de aplicación (Adaptadores).....	13

2.3. Capa de infraestructura.....	13
3. CAPITULO III.....	14
IMPLEMENTACIÓN DE LA ARQUITECTURA HEXAGONAL.....	14
3.1. Introducción.....	15
3.2. Estructura de Archivos.....	15
3.3. Componentes Clave.....	16
4. CAPITULO IV.....	17
CASOS DE ESTUDIO.....	17
4.1. Introducción.....	18
4.2. Aplicaciones de Comercio Electrónico.....	18
4.3. Sistemas de Banca en Línea.....	18
4.4. Sistemas de Gestión de Recursos Humanos (HRMS).....	19
4.5. Sistemas de Gestión de Inventario.....	19
4.6. Sistemas de Transporte y Logística.....	19
4.7. Aplicaciones de Atención Médica y Registros Electrónicos de Pacientes.....	19
4.8. Sistemas de Juegos.....	20
4.9. Ejemplo Practico del uso de la Arquitectura Hexagonal.....	20
a) Sistema de Reservas para una Biblioteca.....	20
5. CAPITULO V.....	22
CONCLUSIÓN.....	22
5.1. CONCLUSIÓN.....	23
6. CAPITULO VI.....	24
BIBLIOGRAFÍA.....	24



1. CAPITULO I

GENERALIDADES

1.1. Introducción

En el mundo de la tecnología informática, la creación de software efectivo se convirtió en un punto importante para las organizaciones y desarrolladores, debido a este contexto la arquitectura de software se rige como un pilar fundamental, el cual define la estructura por donde se estará desarrollando su robustez, eficiencia y escalabilidad del mismo.

La arquitectura de software no es simple codificación sino representa un conjunto de decisiones de diseño organizadas dirigidas hacia la viabilidad a largo plazo de algún proyecto en el que este aplicado. El campo de la arquitectura de software es multidisciplinario el cual aborda muchos desafíos, partiendo desde la organización de recursos y la adaptabilidad ante los cambios futuros, siendo seguro y eficaz en la entrega de soluciones.

1.2. Justificación

La presente investigación sobre la arquitectura de software Hexagonal se fundamenta en la necesidad de aprender sobre tipos de arquitectura más complejos y dinámicos para el desarrollo de software sustentada por los siguientes puntos:

- Tendencia hacia la Flexibilidad y Adaptabilidad de los sistemas informáticos.
- Mantenimiento Eficiente y escalabilidad de los sistemas.
- Experiencias en mejores prácticas de desarrollo.
- Impacto en el desarrollo.

1.3. Objetivos

a) Objetivo General

- Analizar, comprender y evaluar sobre la arquitectura de software Hexagonal hacia un enfoque de diseño de sistemas

b) Objetivos Específicos

- Explorar los fundamentos de la arquitectura hexagonal.
- Examinar la estructura de dicha arquitectura.
- Evaluar la independencia de implementación.

1.4. Arquitectura Hexagonal

Alistair Cockburn es conocido por sus contribuciones al desarrollo ágil de software y ser uno de los firmantes del manifiesto Ágil proponiendo la arquitectura Hexagonal en 2005.

También conocida como puertos y adaptadores, es un paradigma de diseño de software la cual propone una estructura modular y desacoplada para todo sistema informático de modo que se organice por capas concéntricas donde el centro del sistema es donde se encuentra la lógica de negocio la cual se puede comunicar con el mundo a través de puertos.

Sus componentes son los siguientes:

a) Dominio

El dominio es la capa central de la arquitectura hexagonal. Contiene la lógica de negocio y el modelo de datos de la aplicación.

b) Aplicación

La capa de aplicación proporciona la interfaz entre el dominio y la infraestructura.

c) Infraestructura

La capa de infraestructura proporciona interfaces con el mundo exterior, como bases de datos, sistemas operativos, API externas, etc.

1.5. Principios de la Arquitectura Hexagonal

a) Separación de Preocupaciones

División en capas que representan distintas preocupaciones.

b) Independencia de Implementación

El núcleo debe ser independiente de los detalles de la implementación y de los componentes externos.

c) Inversión de dependencias

Permite que el núcleo defina las interfaces (puertos) implementados por los adaptadores.

d) Testabilidad

Permite la testabilidad mediante las pruebas automatizadas al aislar el núcleo efectivamente para así no depender de los detalles de la implementación.

e) Flexibilidad y Adaptabilidad

Al ser modular y desacoplada, facilita los cambios en los requisitos y componentes externos, logrando así implementar nuevas funcionalidades sin afectar el sistema.

f) Reutilización de código

De modo que separamos el núcleo de la implementación facilitamos la reutilización de código en varias partes.

g) Modularidad

Al implementar la arquitectura hexagonal esta obliga a la creación de módulos independientes y fácilmente intercambiables.

h) Simplicidad y Claridad

Al seguir los principios de la arquitectura hexagonal, se busca simplificar la estructura y la claridad del diseño.

1.6. Ventajas y Desventajas

a) Ventajas

- **Flexibilidad y Adaptabilidad.** Al ser de estructura desacoplada la adaptación a cambios es más flexible.
- **Mantenibilidad.** Al separar la lógica de negocio de los detalles de implementación, se simplifica el mantenimiento.
- **Testabilidad.** Favorece a las pruebas automatizadas.
- **Reutilización.** Al estar separado del núcleo los componentes pueden usarse en distintas partes del sistema.
- **Modularidad.** Al hacer uso de dicha arquitectura, los módulos son independientes e intercambiables entre si.
- **Independencia Tecnológica.** Permite el uso de componentes externos o nuevas tecnologías sin afectar la lógica central del sistema, proporcionando así independencia tecnológica.

b) Desventajas

- **Complejidad.** Puede existir mayor complejidad al inicio del proyecto mediante las capas y la definición de interfaces.
- **Sobrecarga de abstracción.** La introducción de interfaces y adaptadores puede llegar a provocar una sobrecarga de abstracción.
- **Exceso de Modularidad.** La modularidad extremo puede llevar a un exceso de fragmentación y complejidad.

- **Curva de Aprendizaje.** Aquellos que no se encuentren familiarizados con dicha arquitectura pueden tardar en adaptarse.
- **Sobrecarga de Interfaces.** Al estar basado en interfaces puede llegar a una proliferación de interfaces.
- **Sobre ingeniería.** Para proyectos pequeños la arquitectura hexagonal se puede percibir como sobre ingeniería.



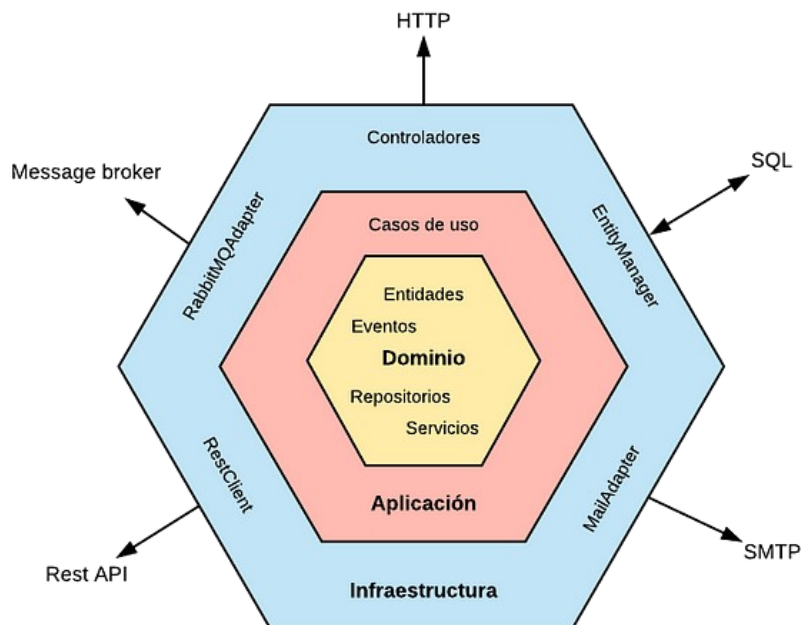
2. CAPITULO II

COMPONENTES DE LA ARQUITECTURA HEXAGONAL

A lo largo de este documento notamos que esta arquitectura ofrece una estructura modular que facilita la adaptación a los cambios en los requisitos de una historia de usuario y la incorporación de nuevas tecnologías sin afectar el núcleo del sistema.

Figura 1

Imagen de los componentes de una arquitectura hexagonal



Nota: Imagen de @edusalguero, 2020, Medium
(<https://medium.com/@edusalguero/arquitectura-hexagonal-59834bb44b7f>)

2.1. Capa de Dominio (Núcleo)

Los dominios son la capa principal de la arquitectura hexagonal. Contiene la lógica de negocio y el modelo de datos de la aplicación. El dominio es independiente de la infraestructura, por lo que se puede reutilizar en diferentes entornos.

La lógica de negocio es responsable de las reglas y cálculos que determinan el funcionamiento de la aplicación. El modelo de datos representa los datos utilizados por la aplicación.

2.2. Capa de aplicación (Adaptadores)

La capa de aplicación proporciona una interfaz entre el dominio y la infraestructura. La capa de aplicación es responsable de la comunicación entre el dominio y la infraestructura. Es esta misma capa se comunica con el dominio a través de interfaces. Las interfaces definen cómo el dominio puede ser accedido por la infraestructura.

La capa de aplicación se comunica con la infraestructura a través de adaptadores. Los adaptadores proporcionan una interfaz entre el dominio y la infraestructura específica.

2.3. Capa de infraestructura

La capa de infraestructura proporciona interfaces con el mundo exterior, como bases de datos, sistemas operativos y API externas. La capa de infraestructura es responsable de interactuar con el mundo exterior. La base de datos almacena datos de la aplicación. El sistema operativo proporciona el entorno en el que se ejecutan las aplicaciones. Las API externas brindan acceso a servicios externos.

Usando la infraestructura del adaptador. Los adaptadores proporcionan una interfaz entre un dominio y una infraestructura específica.



3. CAPITULO III

IMPLEMENTACIÓN DE LA ARQUITECTURA HEXAGONAL

3.1. Introducción

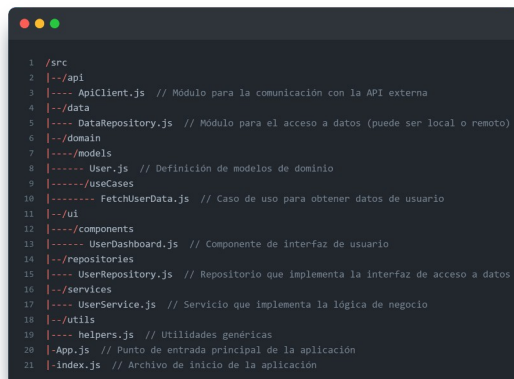
En este capítulo, se presenta la implementación de la arquitectura hexagonal en el proyecto de software [nombre del proyecto]. La arquitectura hexagonal, también conocida como "Puertos y Adaptadores", es un patrón de diseño de software que se centra en la separación de preocupaciones y la creación de componentes independientes y reemplazables. En este capítulo, se describirá cómo se aplicó esta arquitectura en el desarrollo del proyecto, incluyendo la estructura de archivos, los componentes clave y su interacción.

3.2. Estructura de Archivos

La estructura de archivos del proyecto [nombre del proyecto] se diseñó siguiendo los principios de la arquitectura hexagonal. A continuación, se presenta una descripción de la estructura:

Figura 2

Imagen de estructura de archivos



```
1 /src
2 |--/api
3 |---- ApiClient.js // Módulo para la comunicación con la API externa
4 |--/data
5 |---- DataRepository.js // Módulo para el acceso a datos (puede ser local o remoto)
6 |--/domain
7 |----/models
8 |----- User.js // Definición de modelos de dominio
9 |-----/useCases
10 |----- FetchUserData.js // Caso de uso para obtener datos de usuario
11 |--/ui
12 |----/components
13 |----- UserDashboard.js // Componente de interfaz de usuario
14 |--/repositories
15 |---- UserRepository.js // Repositorio que implementa la interfaz de acceso a datos
16 |--/services
17 |---- UserService.js // Servicio que implementa la lógica de negocio
18 |--/utils
19 |---- helpers.js // Utilidades genéricas
20 |--App.js // Punto de entrada principal de la aplicación
21 |--index.js // Archivo de inicio de la aplicación
```

anapp@com

Creditos: Mattias Alexandre Duarte Aparicio

3.3. Componentes Clave

En la implementación de la arquitectura hexagonal, se identificaron componentes clave que desempeñan un papel fundamental en la aplicación:

- **ApiClient:** Este componente se encarga de interactuar con la API externa y proporciona una capa de abstracción para las llamadas de red.
- **DataRepository:** Actúa como un adaptador para el acceso a datos, permitiendo la comunicación con una fuente de datos, ya sea local o remota.
- **User:** Define los modelos de dominio que representan los conceptos clave de la aplicación, como el usuario.
- **FetchUserData:** Caso de uso que encapsula la lógica empresarial para obtener datos de usuario.
- **UserDashboard:** Componente de interfaz de usuario que muestra los datos de usuario y se comunica con el caso de uso correspondiente.
- **UserRepository:** Define la interfaz para el repositorio de usuarios, lo que permite la comunicación con el adaptador de acceso a datos.
- **UserService:** Servicio que implementa la lógica de negocio utilizando casos de uso y adaptadores externos.



4. CAPITULO IV

CASOS DE ESTUDIO

4.1. Introducción

En este capítulo, exploraremos una serie de casos de estudio en los que la arquitectura hexagonal ha sido implementada con éxito en diversas aplicaciones y sistemas. Cada ejemplo demuestra cómo esta arquitectura ha proporcionado beneficios significativos en términos de modularidad, flexibilidad y mantenibilidad en una variedad de dominios de aplicación.

4.2. Aplicaciones de Comercio Electrónico

Las plataformas de comercio electrónico utilizan la arquitectura hexagonal para separar la lógica de negocio relacionada con la gestión de productos, carritos de compra y procesamiento de pedidos de las interfaces de usuario y los sistemas de pago. Esto permite una mayor flexibilidad para adaptarse a diferentes interfaces de usuario y métodos de pago, al tiempo que mantiene la integridad de las transacciones comerciales.

4.3. Sistemas de Banca en Línea

Los sistemas bancarios en línea utilizan la arquitectura hexagonal para garantizar que la lógica de negocio relacionada con las transacciones financieras y la seguridad se mantenga independiente de las interfaces de usuario y las operaciones de almacenamiento de datos. Esto es esencial para garantizar la seguridad y la consistencia de las transacciones financieras.

4.4. Sistemas de Gestión de Recursos Humanos (HRMS)

Las aplicaciones de HRMS a menudo implementan la arquitectura hexagonal para gestionar funciones como la administración de nóminas, la gestión de vacaciones y el seguimiento del rendimiento. Esto permite que la lógica de negocio se mantenga separada de las interfaces de usuario y los sistemas de almacenamiento de datos, lo que facilita la adaptación a las necesidades cambiantes de las empresas.

4.5. Sistemas de Gestión de Inventario

Las aplicaciones de gestión de inventario utilizan la arquitectura hexagonal para gestionar el seguimiento y la gestión de productos y existencias. La lógica de negocio relacionada con la gestión de inventario se mantiene independiente de las interfaces de usuario y las bases de datos, lo que permite una mayor eficiencia en la gestión de inventario y la adaptación a diferentes sistemas de almacenamiento.

4.6. Sistemas de Transporte y Logística

Los sistemas de gestión de flotas, seguimiento de envíos y planificación de rutas en la industria del transporte y la logística a menudo adoptan la arquitectura hexagonal. Esto facilita la comunicación con sistemas externos, como sistemas de seguimiento de GPS y sistemas de información de tráfico, al tiempo que garantiza que la lógica de negocio relacionada con la logística se mantenga aislada de las interfaces de usuario y los sistemas de almacenamiento de datos.

4.7. Aplicaciones de Atención Médica y Registros Electrónicos de Pacientes

En el sector de la atención médica, las aplicaciones de gestión de registros médicos electrónicos (EHR) utilizan la arquitectura hexagonal para asegurar la privacidad y la

seguridad de los datos del paciente. La lógica de negocio relacionada con la gestión de registros médicos se separa de las interfaces de usuario y los sistemas de almacenamiento de datos para cumplir con los estándares de seguridad y regulación de datos en la atención médica.

4.8. Sistemas de Juegos

Los motores de juegos y los sistemas de juegos a menudo implementan la arquitectura hexagonal para separar la lógica del juego, como la física y la inteligencia artificial, de las interfaces gráficas y las interacciones de los jugadores. Esto permite una mayor portabilidad del juego a diferentes plataformas y facilita la extensión y modificación de la funcionalidad del juego sin afectar a su núcleo.

Estos ejemplos ilustran cómo la arquitectura hexagonal se utiliza en una variedad de aplicaciones y sectores para lograr la modularidad, la flexibilidad y la capacidad de adaptación en el diseño de software. Esta arquitectura facilita el mantenimiento y la evolución de los sistemas a medida que cambian los requisitos y las tecnologías.

4.9. Ejemplo Practico del uso de la Arquitectura Hexagonal

a) Sistema de Reservas para una Biblioteca

Dominio: Sera el corazón del sistema además será el que contendrá todas las reglas y lógica de negocio relacionadas con la gestión de reservas de libros en una biblioteca.

Puertos: Son interfaces definidas que permiten la comunicación del dominio con el mundo exterior. Incluyen el puerto de interfaz de usuario (para recibir solicitudes de usuarios), el puerto de base de datos (para acceder a la información sobre libros y usuarios) y el puerto de notificaciones (para enviar correos electrónicos a los usuarios).

Infraestructura: Son implementaciones concretas de los puertos. El adaptador de interfaz web traduce las solicitudes de los usuarios en comandos para el núcleo y presenta las respuestas en la interfaz web. El adaptador de base de datos se encarga de interactuar con la base de datos de la biblioteca. El adaptador de correo electrónico envía notificaciones por correo a los usuarios.

Funcionamiento:

1. Un usuario utiliza la interfaz web para solicitar la reserva de un libro.
2. El adaptador de interfaz web traduce esta solicitud en un formato comprensible para el núcleo del sistema.
3. El núcleo procesa la solicitud, verifica la disponibilidad del libro y realiza la reserva si es posible.
4. Si la reserva es exitosa, el adaptador de correo electrónico envía una confirmación al usuario.
5. Si no es posible hacer la reserva, el adaptador de interfaz web informa al usuario.
6. En este ejemplo se puede ver cómo la arquitectura hexagonal permite que el núcleo del sistema se centre en la lógica de negocio, mientras que los puertos y adaptadores facilitan la comunicación con interfaces externas, como la interfaz web y la base de datos. Esto resulta en un sistema modular y adaptable que puede evolucionar sin afectar la lógica central del negocio.



5. CAPITULO V

CONCLUSIÓN

5.1. CONCLUSIÓN

La arquitectura hexagonal es especialmente útil en aplicaciones que necesitan flexibilidad y mantenibilidad, sobre todo cuando el software está sujeto a cambios frecuentes. Aunque su implementación puede ser un desafío, las ventajas que ofrece hacen que valga la pena considerarla. Ya que promueve a desarrolladores la implementación de principios SOLID y el uso de arquitectura limpia en diferentes proyectos.



6. CAPITULO VI

BIBLIOGRAFÍA

- <https://blog.hubspot.es/website/que-es-arquitectura-hexagonal>
- <https://www.udemy.com/course/proyecto-ecommerce-con-spring-boot-y-arquitectura-hexagonal/>
- <https://codigoencasa.com/arquitectura-hexagonal/>
- <https://appmaster.io/es/blog/arquitectura-hexagonal-java>
- [Arquitectura Hexagonal. O el patrón puertos y adaptadores | by Edu Salguero | Medium](#)