



ADD: MODULO CORE

Carlos Andres Castaño Bustos
2023-09-10

Índice de contenidos

1	Requisitos	2
1.1	Requisitos funcionales	2
1.2	Requisitos no funcionales	2
2	Motivadores de la iteración	3
3	Elementos a refinar	4
4	Conceptos que satisfacen los motivadores	5
5	Elementos de la arquitectura y responsabilidades	6
6	Diseño de vistas	9
7	Validación y análisis de resultados	18

1 Requisitos

1.1 Requisitos funcionales

- El módulo debe permitir crear, obtener y editar configuraciones de Video, Audio e Idioma.
- El módulo debe permitir ejecutar cambios en la configuración de video en tiempo de ejecución desde cualquier otro módulo.
- El módulo debe permitir ejecutar cambios en la configuración de audio en tiempo de ejecución desde cualquier otro módulo.
- El módulo debe permitir ejecutar cambios en la configuración de idioma en tiempo de ejecución desde cualquier otro módulo.
- El módulo debe permitir crear, obtener y editar Contenido del juego (Por ejemplo, Textos, Modos de juego, Roles, Logros, etc).
- El módulo debe permitir obtener los contenidos del juego configurados en tiempo de ejecución desde cualquier otro módulo.
- El módulo debe permitir la gestión de errores mediante eventos.

1.2 Requisitos no funcionales

- La interfaz de usuario debe ser intuitiva.
- La implementación debe de ser modular y escalable, permitiendo que se puedan implementar cambios o nuevos contenidos al juego, sin que esto requiera cambios en el código principal del módulo.

2 Motivadores de la iteración

MOTIVADOR	DESCRIPCIÓN
Sistema modular y escalable	<ul style="list-style-type: none">• Debe de ser independiente de los demás módulos del sistema.• Debe de funcionar como modulo central a los demás módulos, para ejecutar funciones comunes como el Video, Audio, Idiomas y Contenidos.• Debe de estar en la capacidad de cambiar o agregar nuevo contenido sin necesidad de cambiar el código principal del sistema.
Sistema basado en eventos	<ul style="list-style-type: none">• Debe invocar eventos que puedan ser escuchados por los demás módulos.

Table 1: Componentes Motivadores. Fuente propia

3 Elementos a refinar

COMPONENTE	DESCRIPCIÓN
Módulo Core	<p>Este componente estará encargado de varios aspectos clave como:</p> <ul style="list-style-type: none">• Manejo de Video.• Manejo del Audio.• Manejo de Idiomas.• Manejo de Errores.• Gestión de Contenido.

Table 2: Elementos a refinar. Fuente propia

4 Conceptos que satisfacen los motivadores

ESTILO ARQUITECTURA	JUSTIFICACIÓN
Basado en eventos (Publish & Subscribe)	En este enfoque, los diferentes componentes del modulo (en este caso, los managers de Video, Audio, Idioma y Errores) interactúan con los demás módulos principalmente a través de eventos. Cuando ocurre un cambio en las configuraciones, se emiten eventos que otros módulos pueden escuchar y responder en consecuencia. Esta arquitectura basada en eventos permite una comunicación desacoplada y eficiente entre los componentes, ya que no necesitan conocer los detalles internos de los demás para funcionar correctamente.

Table 3: Estilo de arquitectura. Fuente propia

PATRON DE DISEÑO	JUSTIFICACIÓN
Patrón Singleton	Al utilizar el patrón Singleton, se garantiza que exista solo una instancia de cada manager en la memoria en cualquier momento. Esto permite un control centralizado sobre las configuraciones de Video, Audio, Idioma y Contenidos, evitando posibles inconsistencias que podrían surgir si hubiera múltiples instancias independientes.
Principio Open/Closed	Este principio es fundamental, especialmente en la gestión de contenidos, ya que permite la expansión continua y la introducción de nuevos elementos sin alterar las estructuras base existentes. Se podría crear un sistema de contenidos dinámico y adaptable. Los nuevos contenidos, como modos de juego, roles, pisos y desafíos, pueden ser añadidos sin modificar mucho el código original.

Table 4: Patrones y principios de diseño. Fuente propia

5 Elementos de la arquitectura y responsabilidades

NOMBRE	RESPONSABILIDAD	RELACIONES
VideoManager	VideoManager es el componente encargado de manipular las configuraciones de video de la aplicación. Su función principal es recibir los eventos de cambio de configuración en video y realizar dichos cambios en tiempo de ejecución.	<ul style="list-style-type: none">• SettingsManager• Módulos
AudioManager	AudioManager es el componente encargado de manipular las configuraciones de audio de la aplicación. Su función principal es recibir los eventos de cambio de configuración en audio y realizar dichos cambios en tiempo de ejecución. También es el encargado de la reproducción de sonidos en tiempo de ejecución.	<ul style="list-style-type: none">• SettingsManager• Módulos
ErrorManager	El ErrorManager es el componente encargado la gestión de errores de la aplicación. Su función principal es recibir todos los eventos de errores de la aplicación y re-direccionar dicho evento al componente interesado en dicho evento de error en tiempo de ejecución.	<ul style="list-style-type: none">• Módulos
GameManager	GameManager es el componente encargado la gestión de contenidos(Por ejemplo, Modos de juego, Roles, Logros, Desafíos, etc.) de la aplicación. Su función principal es proporcionar la información de los contenidos configurados a los módulos que las requieran en tiempo de ejecución.	<ul style="list-style-type: none">• GameConfigurationService• Módulos

Continúa en la siguiente página.

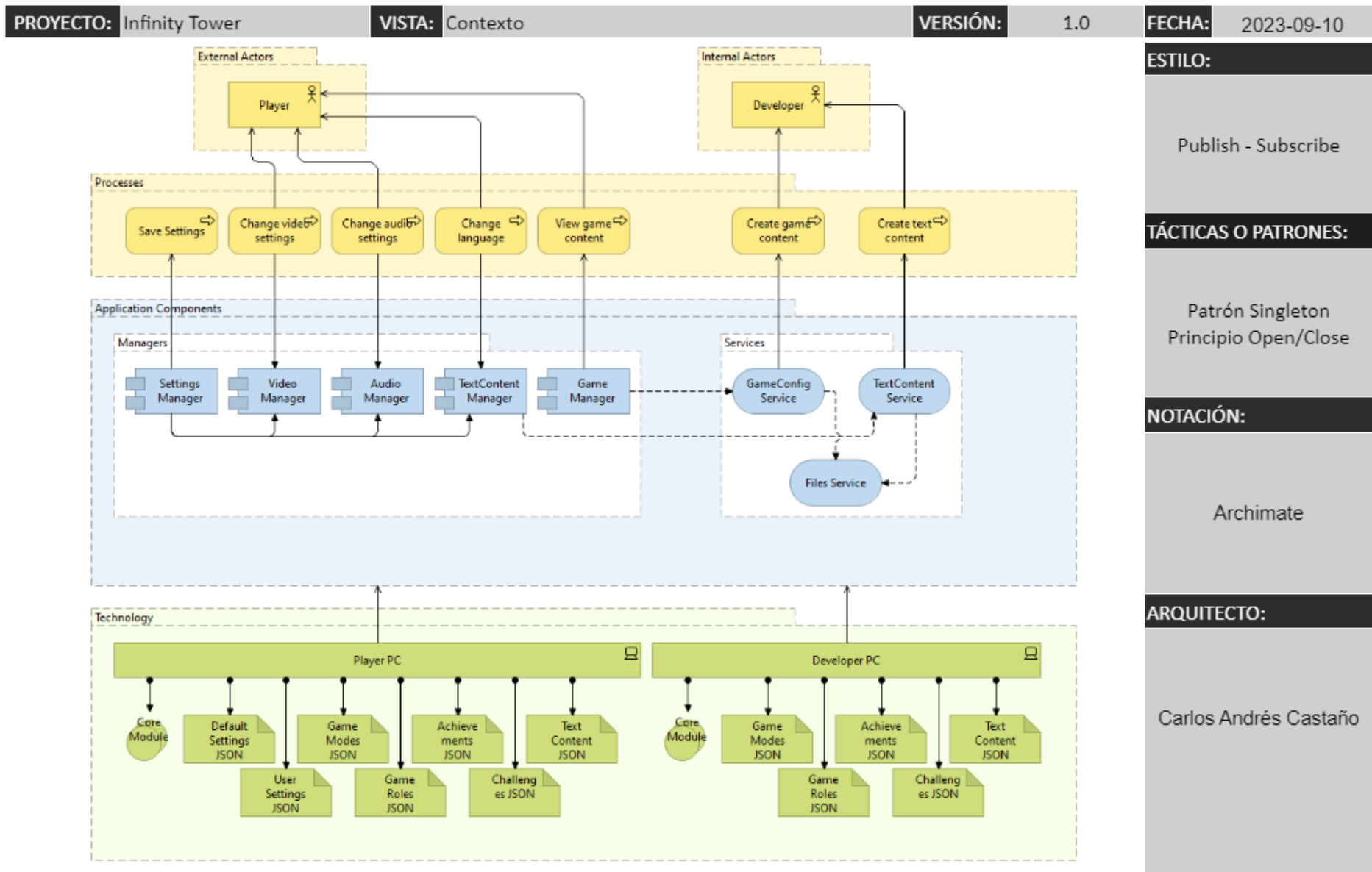
NOMBRE	RESPONSABILIDAD	RELACIONES
TextContentManager	El TextContentManager es el componente encargado la gestión de contenidos de texto(Traducciones de textos estáticos) de la aplicación. Su función principal es proporcionar las traducciones de los textos estáticos de la aplicación a los módulos que las requieran en tiempo de ejecución.	<ul style="list-style-type: none"> • TextContentService • Módulos
SettingsManager	El SettingsManager es el componente encargado de almacenar y gestionar las configuraciones de Audio, Video e Idioma del usuario. Su función principal es proporcionar las dichas configuración del usuario a los módulos y managers que las requieran en tiempo de ejecución.	<ul style="list-style-type: none"> • FilesService • Módulos
GameConfigurationService	El GameConfigurationService es el servicio encargado de cargar y guardar los contenidos(Por ejemplo, Modos de juego, Roles, Logros, Desafíos, etc.) de la aplicación.	<ul style="list-style-type: none"> • FilesService • GameManager
TextContentService	El TextContentService es el servicio encargado de cargar y guardar los contenidos de texto(Traducciones de textos estáticos) de la aplicación.	<ul style="list-style-type: none"> • FilesService • TextContentManager

Continúa en la siguiente página.

NOMBRE	RESPONSABILIDAD	RELACIONES
FilesService	El FilesService es un servicio encargado de cargar y guardar los archivos de tipo JSON de la aplicación.	<ul style="list-style-type: none"> • GameConfigurationService • TextContentService

Table 5: Componentes del sistema. Fuente propia

6 Diseño de vistas



ESTILO:

Publish - Subscribe

TÁCTICAS O PATRONES:

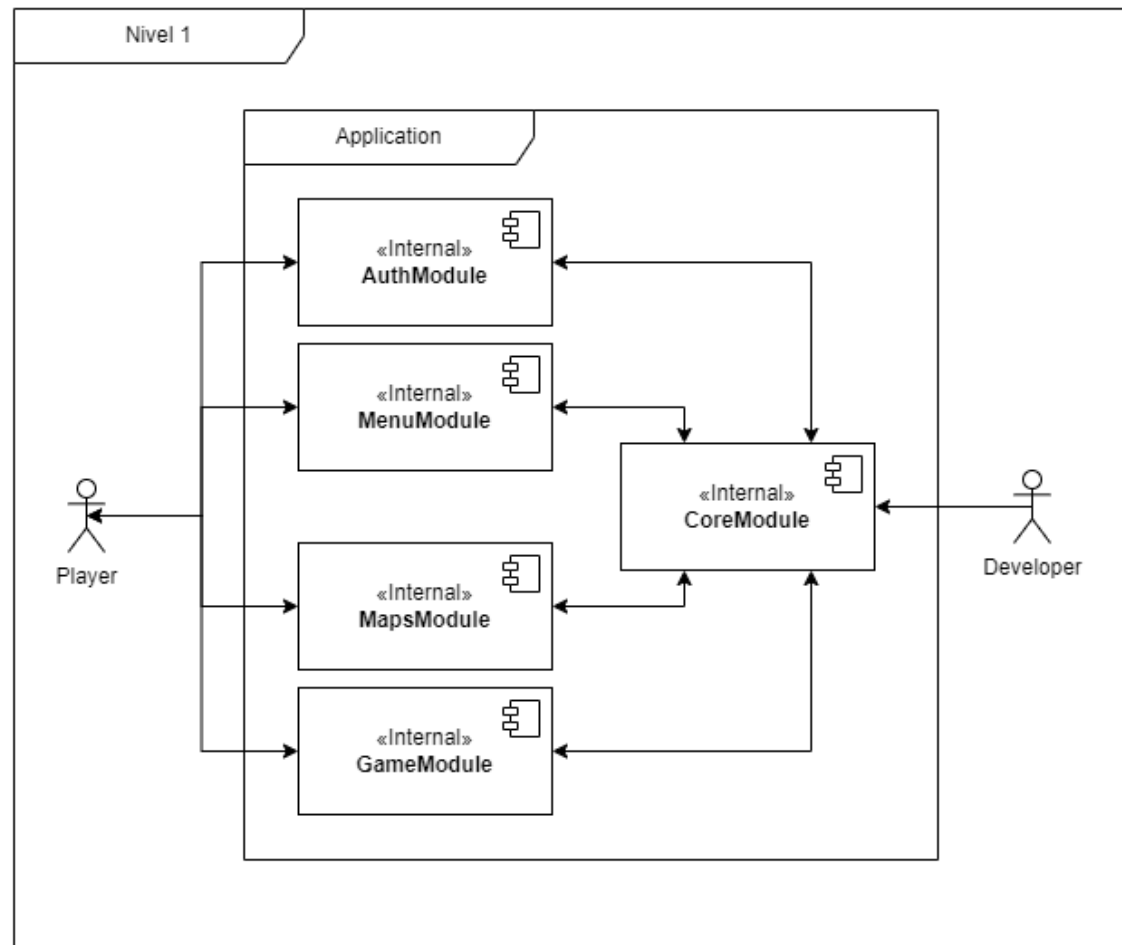
Patrón Singleton
Principio Open/Close

NOTACIÓN:

UML

ARQUITECTO:

Carlos Andrés Castaño



ESTILO:

Publish - Subscribe

TÁCTICAS O PATRONES:

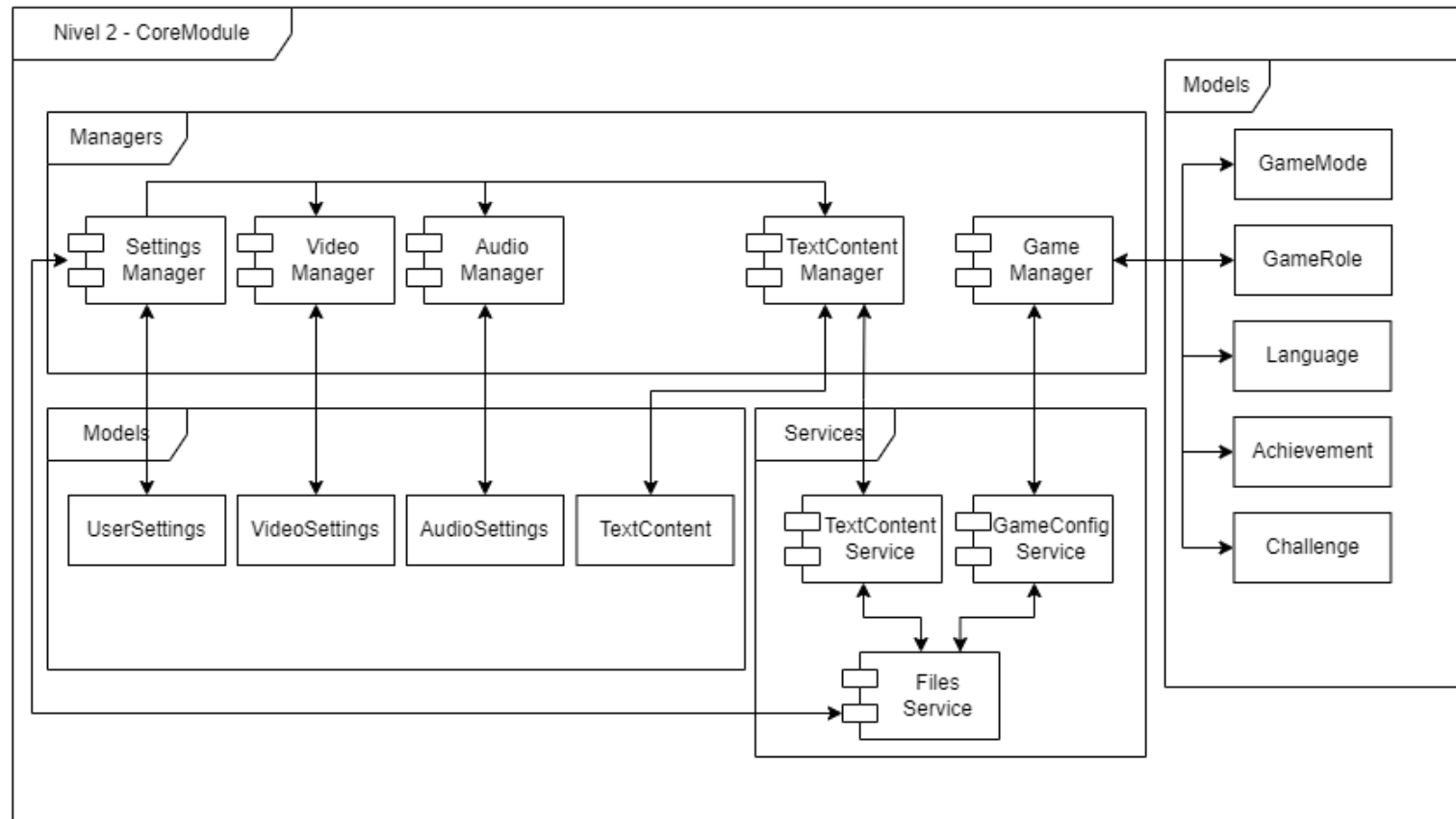
Patrón Singleton
Principio Open/Close

NOTACIÓN:

UML

ARQUITECTO:

Carlos Andrés Castaño



PROYECTO: Infinity Tower**VISTA:** Información**VERSIÓN:** 1.0**FECHA:** 2023-09-10**Object:GameMode**

string Id
string Name
bool IsEnabled
List<Translation> ContentNames
List<Translation> Description
string Image
bool RankingEnabled
List<Translation> RankingDescription

Object:GameRole

string Id
string Name
bool IsEnabled
List<Translation> ContentNames
List<Translation> Description
string SquareImage
string CircleImage

Object:Challenge

string Id
string Name
bool IsEnabled
List<Translation> ContentNames
List<Translation> Description
string Image

Object:Achievement

string Id
string Name
bool IsEnabled
List<Translation> ContentNames
List<Translation> Description
string IconLocked
string IconUnlocked
AchievementCategory Category

Object:TextContent

string Code
List<Translation> Translations
TextContentCategory Category

Object:Language

string Id
string Name
bool IsEnabled
List<Translation> ContentNames
List<Translation> Description

Object:Translation

string Key
string Value

Enum:AchievementCategory

Progress = 0
Abilities = 1
Challenges = 2

Enum:TextContentCategory

General = 0
Menubar = 1
Settings = 2
Profile = 3
Achievements = 4
Auth = 5
Maps = 6
Game = 7

ESTILO:

Publish - Subscribe

TÁCTICAS O PATRONES:Patrón Singleton
Principio Open/Close**NOTACIÓN:**

UML

ARQUITECTO:

Carlos Andrés Castaño

PROYECTO: Infinity Tower

VISTA: Desarrollo

VERSIÓN: 1.0

FECHA: 2023-09-10

ESTILO:

Publish - Subscribe

TÁCTICAS O PATRONES:

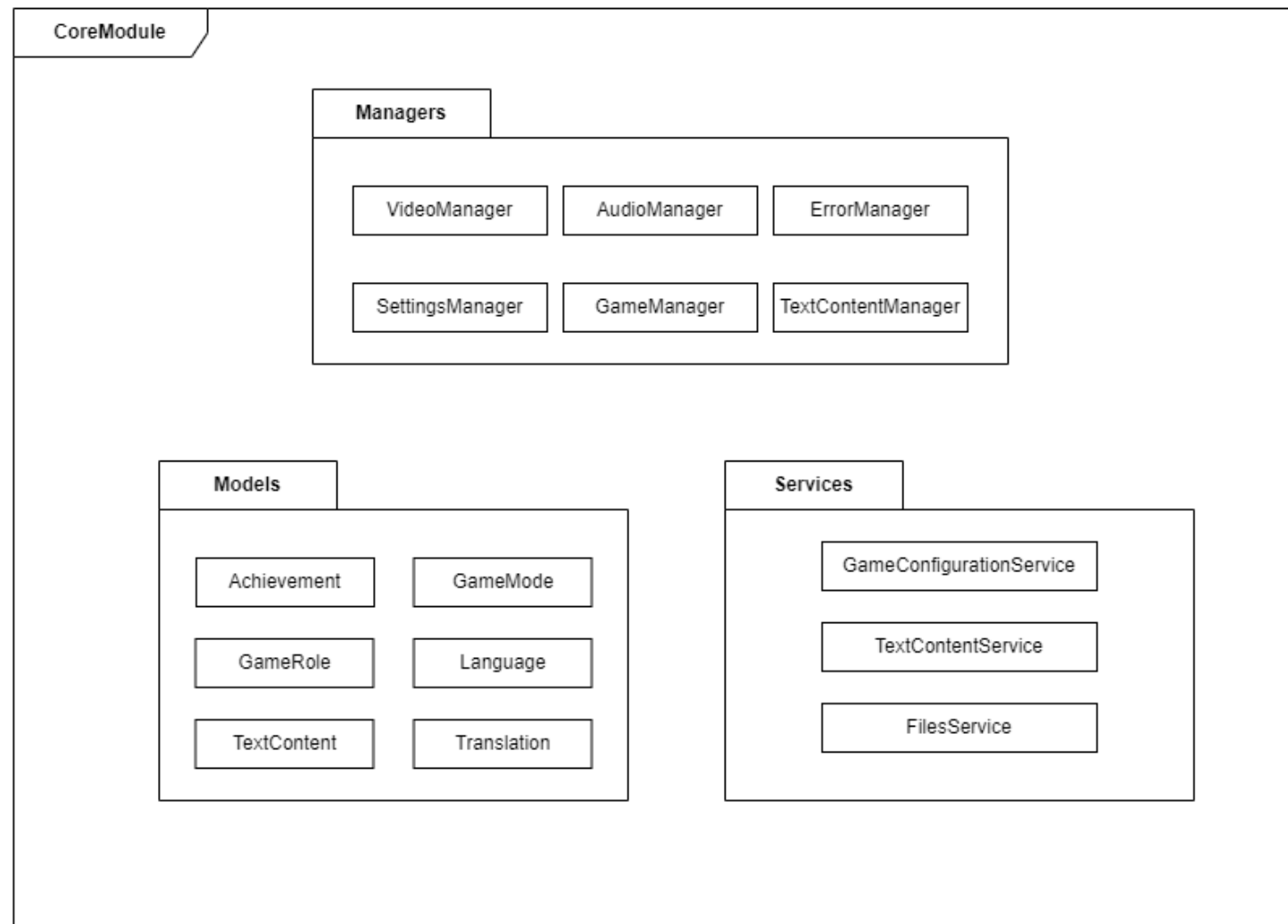
Patrón Singleton
Principio Open/Close

NOTACIÓN:

UML

ARQUITECTO:

Carlos Andrés Castaño



ESTILO:

Publish - Subscribe

TÁCTICAS O PATRONES:

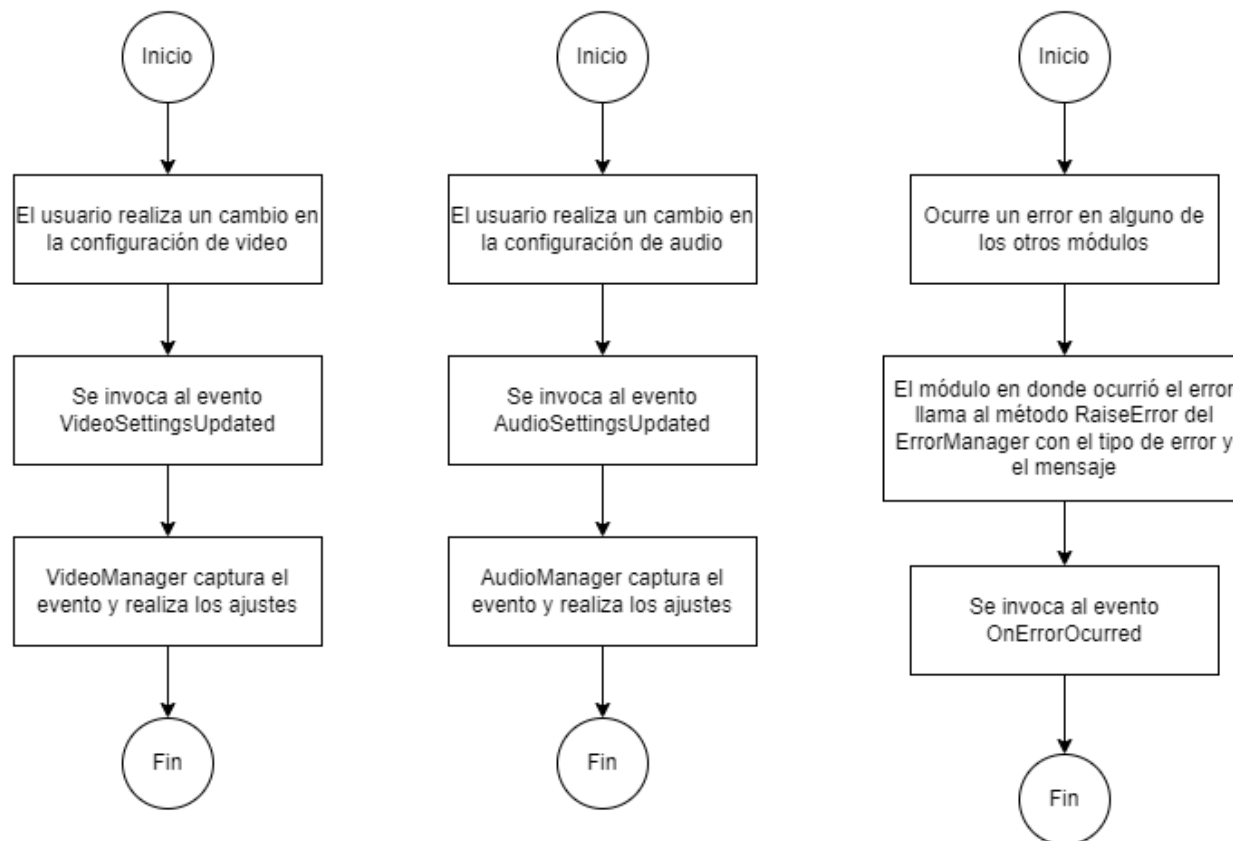
Patrón Singleton
Principio Open/Close

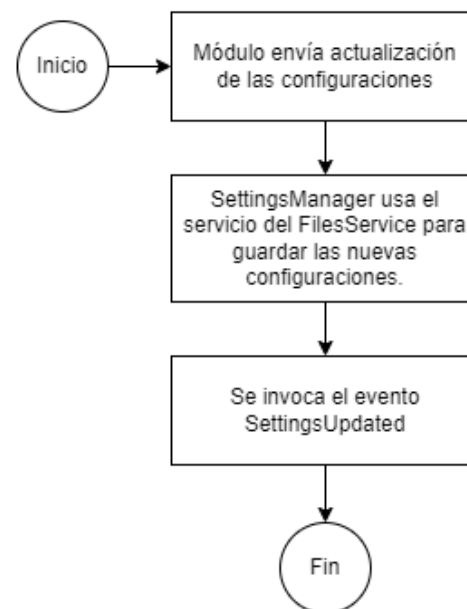
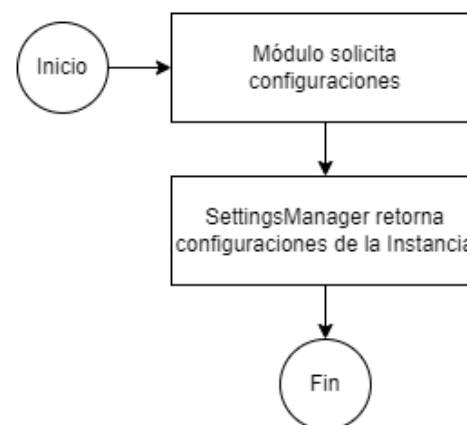
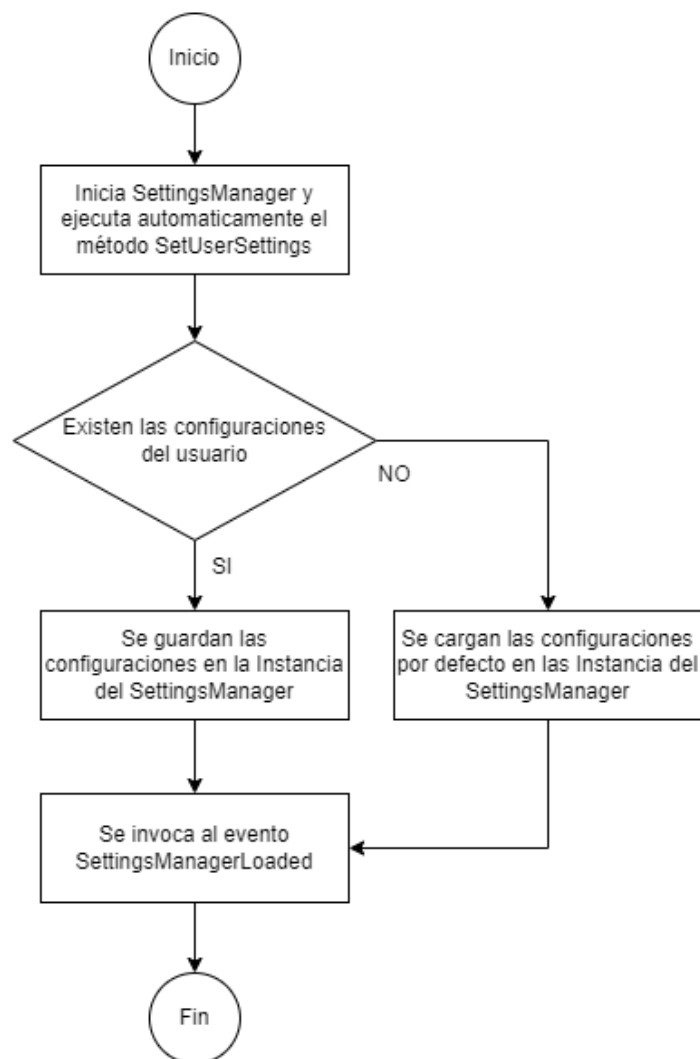
NOTACIÓN:

Diagrama de flujo

ARQUITECTO:

Carlos Andrés Castaño




ESTILO:

Publish - Subscribe

TÁCTICAS O PATRONES:

Patrón Singleton
Principio Open/Close

NOTACIÓN:

Diagrama de flujo

ARQUITECTO:

Carlos Andrés Castaño

ESTILO:

Publish - Subscribe

TÁCTICAS O PATRONES:

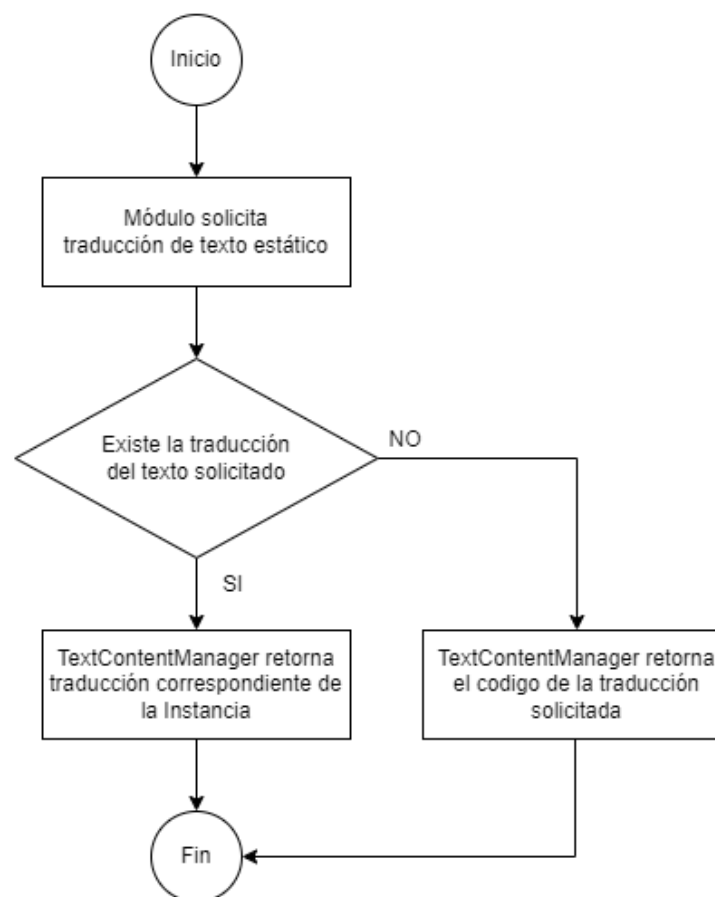
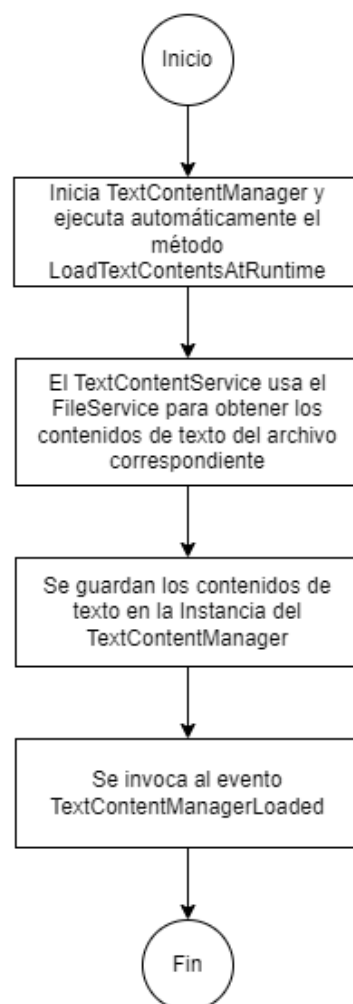
Patrón Singleton
Principio Open/Close

NOTACIÓN:

Diagrama de flujo

ARQUITECTO:

Carlos Andrés Castaño



ESTILO:

Publish - Subscribe

TÁCTICAS O PATRONES:

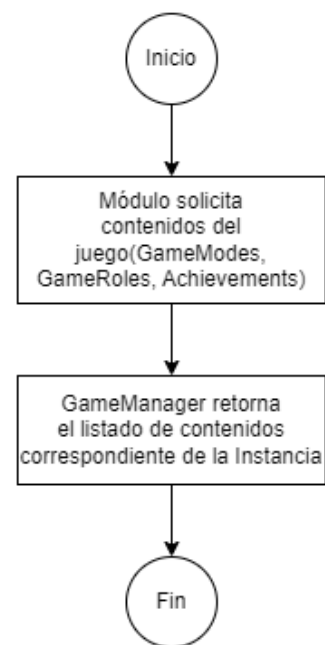
Patrón Singleton
Principio Open/Close

NOTACIÓN:

Diagrama de flujo

ARQUITECTO:

Carlos Andrés Castaño



7 Validación y análisis de resultados

	DESCRIPCIÓN
RESULTADOS	<p>El diseño de la arquitectura del módulo core ha demostrado ser efectivo al implementar patrones que han simplificado el código de manera significativa. La aplicación exitosa de estos patrones ha llevado a un código claro, comprensible y fácilmente mantenible. La modularidad y la claridad han sido los pilares, permitiendo que cada componente del módulo se desarrolle de forma independiente y sea fácilmente entendido por cualquier persona. Además, la implementación de eventos ha proporcionado una forma robusta y desacoplada para que el módulo core se comunique con otros módulos de la aplicación.</p>
ACCIONES A SEGUIR	<p>Dado que los resultados obtenidos han sido satisfactorios, lo siguiente es continuar con la implementación completa del módulo core utilizando esta arquitectura diseñada. Las acciones futuras deben centrarse en aprovechar al máximo las ventajas proporcionadas por la arquitectura basada en eventos y la modularidad.</p> <p>Las pruebas rigurosas también deben llevarse a cabo para validar que la implementación se ajusta a las expectativas y que todas las funcionalidades operan como se espera.</p>

Table 6: Resultados y acciones a seguir. Fuente propia