Catalina Patiño Forero  
Descripción de Arquitectura de Sistema y Software

SPRINT 1

Catalina Patiño Forero - Wilfer M. Salas Gonzales  
[cpatin10@eafit.edu.co](mailto:cpatin10@eafit.edu.co) - [wsalasg@eafit.edu.co](mailto:wsalasg@eafit.edu.co)   
Versión 1.0  
Abril 1 del 2018

Contenido

[1. Introducción 4](#_Toc509295480)

[2. Diseño Evolutivo de Arquitectura 5](#_Toc509295481)

[2.1 Decisiones de Diseño: 5](#_Toc509295482)

[2.2 Contexto del sistema 5](#_Toc509295483)

[Diagrama de Contexto 5](#_Toc509295484)

[2.3 Vista Funcional 6](#_Toc509295485)

[2.4 Vista de Informaciónn / Modelo de Datos 7](#_Toc509295486)

[2.4.1 Data Structure (RDBMS) 7](#_Toc509295487)

[2.4.2 Definición de Datos si utiliza modelos NoSQL 8](#_Toc509295488)

[2.5 Vista de Despliegue 9](#_Toc509295489)

[2.5.1 Modelo de Plataforma de Ejecución 9](#_Toc509295490)

[2.5.2 Dependencias de Software 9](#_Toc509295491)

[2.5.3 Modelo de Red 9](#_Toc509295492)

[2.6 Vista de Desarrollo 11](#_Toc509295493)

[2.6.1 Module Structure 11](#_Toc509295494)

[2.6.2 Diseño común 11](#_Toc509295495)

[3. Atributos de Calidad del Sistema 13](#_Toc509295496)

[3.1 Rendimiento y Escalabilidad 13](#_Toc509295497)

[3.2 Seguridad 13](#_Toc509295498)

[3.3 Disponibilidad and Resilencia 13](#_Toc509295499)

[3.4 Extensibilidad 13](#_Toc509295500)

[3.5 Otros Atributos de Calidad 14](#_Toc509295501)

[3.5.1 Accesibilidad 14](#_Toc509295502)

[3.5.2 Internacionalización 14](#_Toc509295503)

[3.5.3 Localización 14](#_Toc509295504)

[3.5.4 Regulación 14](#_Toc509295505)

[3.5.5 Usabilidad 14](#_Toc509295506)

[4. Middlewares 15](#_Toc509295507)

[4.1 Definición de Servicios Web 15](#_Toc509295508)

[4.2 Definición de Servicios Mensajería y Notificación 15](#_Toc509295509)

[4.3 Definición de Protocolos 15](#_Toc509295510)

[4.4 Definición de Datos distribuidos 15](#_Toc509295511)

[4.5 Otros middlewares 15](#_Toc509295512)

[5. Arquitectura especifica por Tipo de Sistema 15](#_Toc509295513)

# Introducción

**ARQUITECTURA BASE (SISTEMA)**

**TOMAR LAS VISTAS DE ARQUITECTURA Y PATRONES DE ARQUITECTURA USABLES PARA SU SISTEMA:**

1. **Arquitectura global / Funcional del sistema**

En el siguiente documento se hará una presentación de la arquitectura del sistema a desarrollar. Por ejemplo, se presenta una vista funcional, una física y una de desarrollo, además de un diagrama de desarrollo. Para complementar un poco, y para contextualizar al lector de qué se pretende desarrollar con la aplicación, a continuación, se presentarán dos diagramas de casos de uso que ilustran a grandes rasgos el funcionamiento del sistema.

Paquete Nivel: Actividades de entrenamiento auditivo desarrolladas en el juego.

* + Leer instrucciones: El usuario puede leer las instrucciones correspondientes de cada nivel.
  + Escuchar intervalo: Dentro del nivel se reproduce el sonido correspondiente a un intervalo el cual será escuchado por el usuario.
  + Tocar respuesta: El usuario puede tocar en el piano la respuesta que considere apropiada a un reto propuesto en el nivel.
  + Recibir puntaje: El usuario recibe puntos de acuerdo a las acciones tomadas dentro del nivel.
  + Verificar respuestas: El sistema verifica las respuestas del usuario y de acuerdo a esta información genera datos que serán almacenados posteriormente.
  + Medir tiempo de respuesta: El sistema mide el tiempo que tarda un usuario en dar una respuesta, a partir de esto se generan datos que serán almacenados posteriormente.

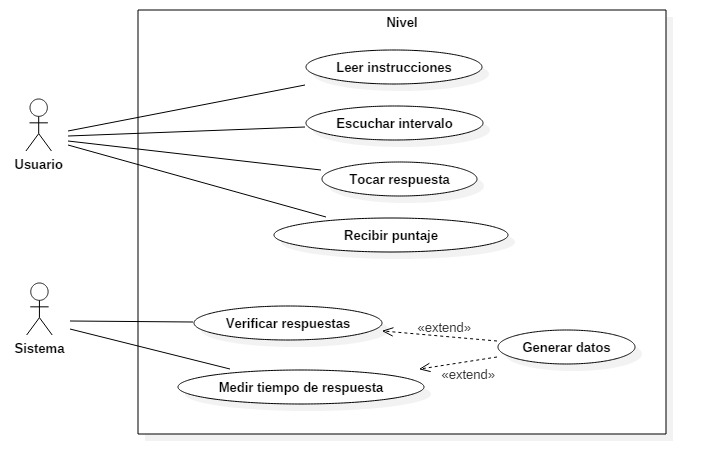


Figura 1. Caso de uso Nivel

Paquete IA: Actividades de análisis de datos, reconocimiento del tipo de usuarios que usan el juego y definición de un posible camino de aprendizaje para cada usuario.

* + Analizar datos: El sistema analiza los diferentes datos almacenados de los diferentes usuarios.
  + Buscar tendencias: De acuerdo al análisis de datos, el sistema busca tendencias que presenta el usuario actual.
  + Ajustar dificultad: De acuerdo a la tendencia del usuario, el sistema ajusta la dificultad del juego para que se acomode a éste.
  + Generar camino de aprendizaje: De acuerdo a la tendencia del usuario el sistema puede generar un camino de aprendizaje u orden de niveles que hagan más efectivo el aprendizaje del usuario.
  + Clasificar usuarios por tendencias: De acuerdo al análisis de datos, el sistema puede clasificar usuarios de acuerdo a las tendencias que presentan en el momento de jugar.

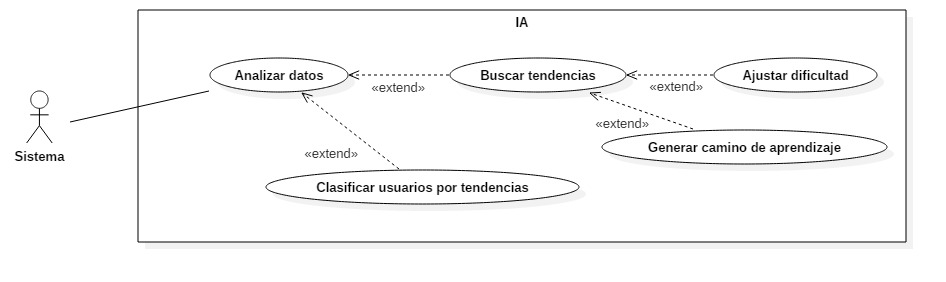


Figura 2. Caso de uso IA

1. **Arquitectura de integración entre sistemas**

En el siguiente diagrama se presenta la integración entre componentes del sistema. Todos los componentes presentados son de reuso, ninguno es de desarrollo propio.

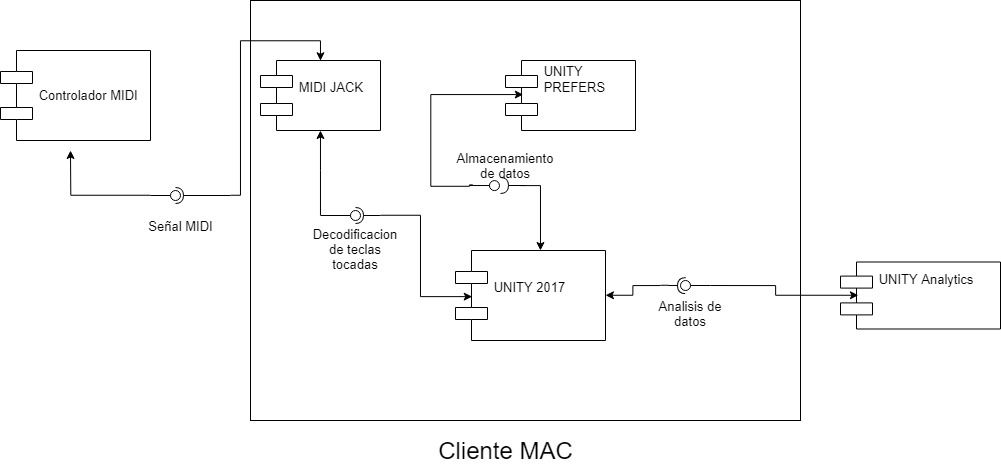


Figura 3. Diagrama de integración

1. **Arquitectura de datos:**

El sistema almacena datos en el sistema de archivos del sistema operativo, y también los envía a Unity Analytics, el cual realiza operaciones de analítica. El esquema utilizado se amplía en la sección 2.4.

1. **Consideraciones particulares por tipo de sistema**

La mayor particularidad del sistema es que todo el desarrollo se realiza dentro del ecosistema provisto por Unity. Esto presenta ventajas y desventajas. La mayor ventaja es la alta interoperabilidad del sistema, dado que la mayor parte de las herramientas están provistas por la misma interfaz, aunque esto vuelve de Unity una caja negra para el desarrollador. La mayor desventaja, es que Unity es cerrado con su código fuente, por lo que sólo se suelen encontrar las herramientas que la misma empresa proveer, u otras desarrolladas por terceros que no suelen ser las más óptimas posible. Esto dificulta la búsqueda de alternativas cuando se hace necesario, por ejemplo, para encontrar una herramienta de integración continua o de análisis de código estático.

1. **Definición Específica de la tecnología**

En términos generales el sistema consiste en una aplicación de entrenamiento auditivo para equipos con sistema operativo Mac OS, debido a que el lugar en el que se va a ejecutar la aplicación sólo posee de esta plataforma. Sin embargo, el equipo de desarrollo solo cuenta con máquinas Windows para el desarrollo. El programa recibirá entradas por medio de un mouse o por un piano o teclado electrónico. Dicha conexión se realiza por medio de MIDI. También se conecta con Unity Analytics, herramienta gratuita (debe agregarse el servicio desde el proyecto Unity) que almacena la información determinada por el desarrollador y le realiza diferentes procesos estadísticos. Aunque, también se realiza un almacenamiento más controlado por medio de la librería de Player Prefs de Unity.

Como ya se ha mencionó la codificación y desarrollo del proyecto se desarrolla con Unity 2017, el cual usa como IDE a Visual Studio (en este caso 2015 que integra Visual Studio Tools para Unity) o a Mono. Se escoge el primero por ser la herramienta por excelencia para desarrollo de código C#, uno de los dos lenguajes usados por la herramienta. El otro lenguaje es JavaScript, sin embargo, para no incrementar la curva de aprendizaje se escoge C# por su similitud con Java y C++.

Teniendo en cuenta que se va a desarrollar con Unity, se deben buscar herramientas que se integren lo suficientemente bien con éste. Para el análisis de código estático se usará PVS-Studio, que no presenta las dificultades que presentan otras herramientas al tratar con proyectos Unity. Para Integración continua se tiene Jenkins, el cual requiere del plugin de Unity3D. Y para el desarrollo de pruebas unitarias se desarrolla con Unity Test Runner, herramienta integrada a Unity.

Por otro lado, para la gestión de cambios se usa GitHub y para la gestión del proyecto Visual Studio Team Services. Además, se hace uso de StarUML y draw.io para el diseño de la arquitectura del sistema.

# Diseño Evolutivo de Arquitectura

## Decisiones de Diseño:

* Se usa un patrón arquitectónico Modelo-Vista-Controlador (MVC).
* El desarrollo se da usando programación orientada a objetos (OOP) y programación orientada a eventos. Esta última se hace necesaria ya que los scripts suelen reaccionar a eventos que ocurren, ya sea por una acción del sistema o del usuario, por ejemplo, cuando el usuario presiona una tecla, el sistema debe reaccionar acorde a este evento. También se tiene OOP, porque no todos los scripts se asocian a un MonoBehaviour (clase base de los scripts de Unity).
* Se acompaña la programación orientada a eventos haciendo uso del patrón de diseño Observer.
* Se implementa el patrón de diseño Singleton Para evitar la instanciación de clases que deben ser únicas a lo largo de los niveles y almacenan información que debe ser accedida por otras clases.
* Se usa CamelCase para el nombramiento de las variables. A menos que sea una variable constante y estática donde se escribe el nombre en mayúsculas y se usa Snake\_Case.
* Todas las variables y comentarios se escriben en inglés. A menos que se el usuario final pueda visualizarlas, caso en el que se usa español, por ejemplo, los strings correspondientes al menú principal.

## Contexto del sistema

### Diagrama de Contexto

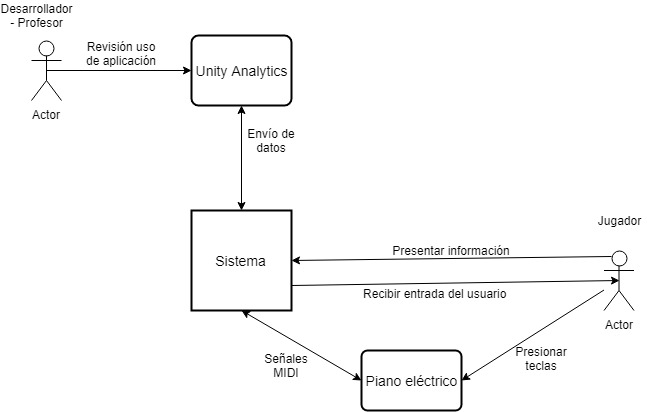


Figura 4. Contexto del sistema

## Vista Funcional

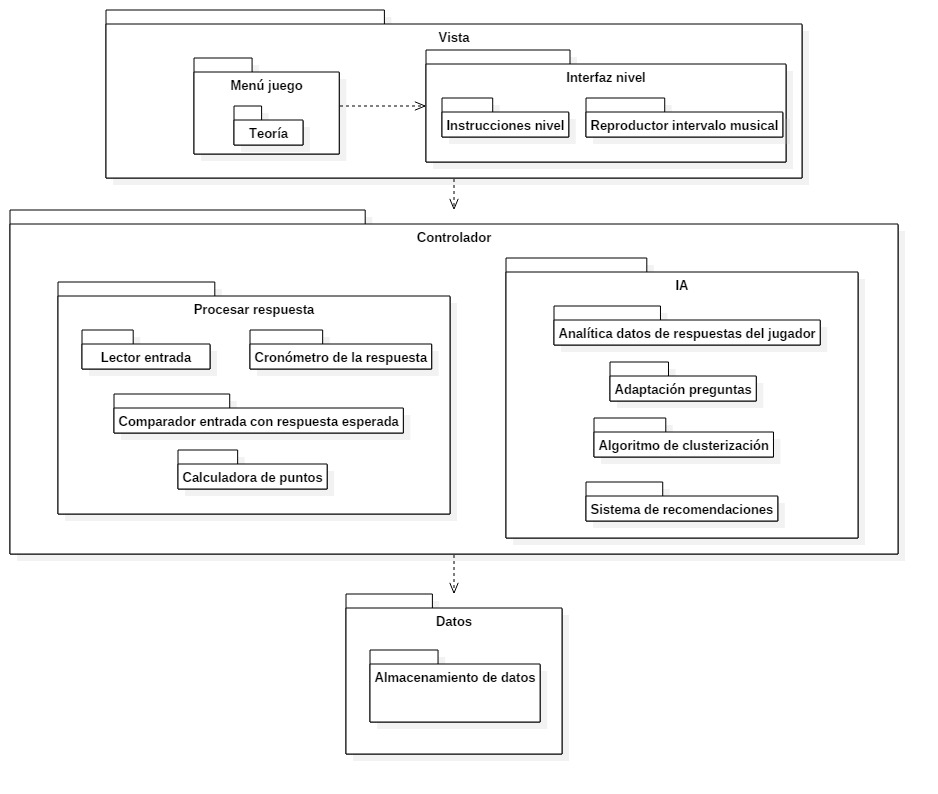


Figura 5. Diagrama funcional

|  |  |
| --- | --- |
| **Nombre del Sistema, Subsistema, Entidad o Componente** | Vista |
| Descripción | Se encarga de la presentación, facilitando la comunicación usuario-sistema. Entre sus componentes principales se encuentra el menú del juego, el cual lleva a cada uno de los niveles y presenta un poco de teoría sobre el entrenamiento auditivo. Dentro de la interfaz de cada nivel se tiene acceso a las instrucciones del mismo, y se presenta un componente que se encarga de la reproducción de los intervalos. |
| Interacción | La vista recibe información del controlador de forma que puede presentar al usuario cambios importantes. Por ejemplo, el controlador le informa a la vista que teclas debe marcar como correctas o incorrectas con el fin de que el usuario note que su respuesta ya fue registrada y si esta estuvo correcta o no. |

|  |  |
| --- | --- |
| **Nombre del Sistema, Subsistema, Entidad o Componente** | Controlador |
| Descripción | Se encarga del procesamiento de la información y de los algoritmos necesarios para el funcionamiento del sistema. En este punto sobresalen dos componentes que se encargan del procesamiento de la respuesta y la inteligencia artificial del sistema. El primero incluye funciones como la medición del tiempo de respuesta, la verificación de si la respuesta es correcta o no, el cálculo de los puntos, y la recepción de la información de entrada. El segundo, se encarga de hacer análisis de los datos, de adaptar las preguntas al usuario, y del algoritmo de clusterización que facilitará la función del sistema de recomendaciones. |
| Interacción | El controlador envía la información que requiere persistencia al componente de datos. |

|  |  |
| --- | --- |
| **Nombre del Sistema, Subsistema, Entidad o Componente** | Datos |
| Descripción | Se encarga del almacenamiento de la información que necesita persistencia. |
| Interacción | Envía la información al controlador, especialmente al componente encargado de la inteligencia artificial. |

## Vista de Informaciónn / Modelo de Datos

### Definición de archivos

Se van a almacenar archivos en el sistema, en los cuales Unity almacena y recupera datos. Para ello, se usa un formato tipo JSON con almacenamiento key-value.

{

{"Usuario", <String: usuario>},

{ "Nivel", <String: nombre del nivel> },

{ "Respuesta correcta", <boolean: usuario respondió correctamente?> },

{ "Intervalo preguntado", <String: nombre del intervalo preguntado> },

{ "Intervalo respondido", <String: nombre del intervalo respondido>},

{ "Primer nota", <String: nombre de la primer nota del intervalo> },

{ "Segunda nota (preguntada)", <String: nombre de la segunda nota del interval, la esperada> },

{ "Segunda nota (respondida)", <String: nombre de la segunda nota del interval, la respondida>},

{ "Tiempo de respuesta", <foat: tiempo que tardó el usuario en responder>}

}

El esquema anterior es para el segundo nivel (3D), para el primer nivel (2D), se usa un esquema similar, pero sin el campo "Segunda nota (respondida)".

### Datos en Unity Analytics

Los datos enviados a enviar a Unity Analytics tienen un formato tipo diccionario, de parejas key-value, donde la llave es de tipo string y el value de tipo objeto, similar a un documento JSON. Para el nivel 2 se maneja un esquema igual al usado en almacenamiento, sólo que sin el campo "Usuario".

## Vista de Despliegue

### Modelo de Plataforma de Ejecución

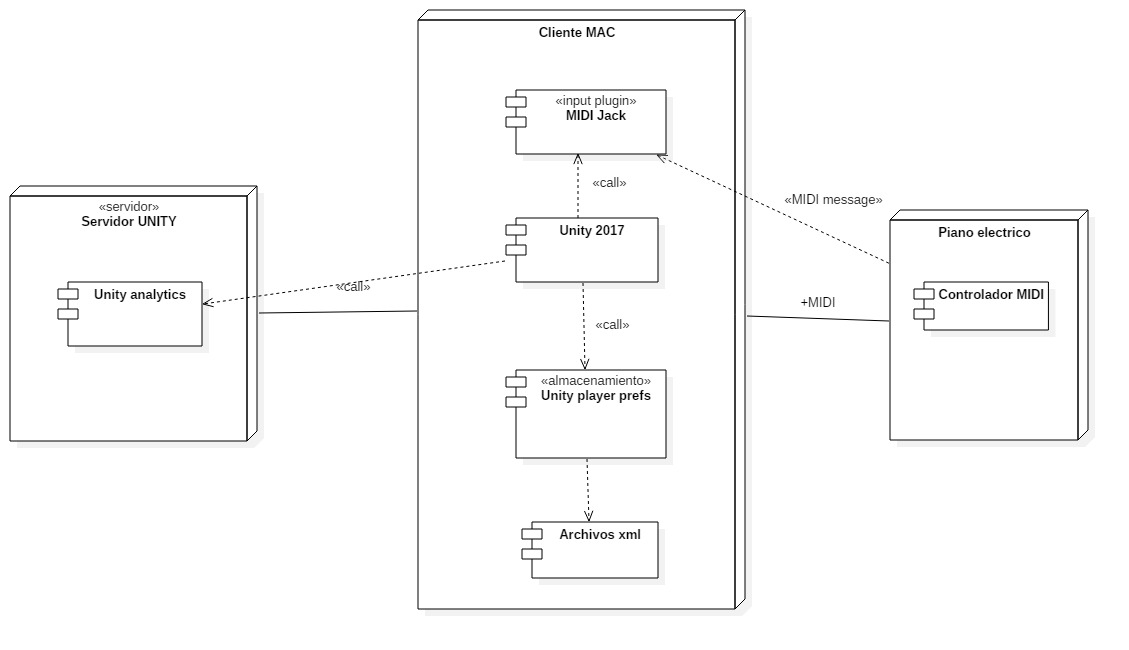


Figura 6. Diagrama de despliegue.

### Dependencias de Software

Como se puede observar en el diagrama se tienen tres nodos. En el computador se especifica como sistema operativo Mac OS (cliente Mac) debido a que la sala destinada para el entrenamiento auditivo de la Universidad Eafit dispone únicamente de computadores con este sistema operativo. El juego será desarrollado y desplegado por medio de Unity 2017, el cual es escogido por ser gratuito y multiplataforma, además, frente a su mayor competidor Unreal, presenta una menor curva de aprendizaje, mayor soporte de la comunidad y requisitos de ejecución más bajos (no todo el equipo de desarrollo podría instalar Unreal en sus máquinas personales). Teniendo en cuenta que se usa Unity, para el almacenamiento y la analítica de datos se escogen sistemas integrados a la plataforma, lo que disminuye el riesgo al usar software de terceros, además estos paquetes son gratuitos. El almacenamiento en Unity se realiza por medio de Player Prefs el cual permite almacenar la información en archivos xml dentro del sistema operativo. La anlítica, se hace sobre un servidor provisto por Unity. Además, para poder leer señales MIDI, protocolo estándar para la comunicación entre instrumentos musicales y el sistema, se integrará el paquete para Unity llamado MIDI Jack.

A continuación, se profundiza un poco más en los componentes a utilizar.

|  |  |
| --- | --- |
| **Nombre del Componente** | MIDI Jack |
| **Justificación de compra o reuso** | Se necesita un componente que facilite la comunicación entre un instrumento electrónico y el computador, es decir, facilita la interoperabilidad del sistema. MIDI Jack permite hacer esto y su uso no tiene ningún costo. |
| **Recomendaciones para su adquisición** | Se debe utilizar una versión de Unity superior o igual a Unity 5.  Se puede usar sólo para sistemas operativos Windows o Mac OS X. |
| **Sitio / Origen del componente** | <https://github.com/keijiro/MidiJack> |

|  |  |
| --- | --- |
| **Nombre del Componente: Unity 2017** | Unity 2017 |
| **Justificación de compra o reuso** | Para el desarrollo rápido y efectivo de desarrollo de videojuegos se requiere un motor, entre los más reconocidos se encuentra Unity, cuya versión actual es Unity 2017. Es gratuito y permite compilar para Mac OS. |
| **Recomendaciones para su adquisición** | Todos los integrantes del equipo de desarrollo deben poder ejecutar el programa en sus computadores personales, para ello dichos equipos deben cumplir los requisitos de sistemas presentados en el siguiente link:  <https://unity3d.com/es/unity/system-requirements> |
| **Sitio / Origen del componente** | Creado por Unity Technologies  <https://unity3d.com/es> |

|  |  |
| --- | --- |
| **Nombre del Componente** | Unity Player Prefs |
| **Justificación de compra o reuso** | Paquete gratuito integrado a Unity que facilita el almacenamiento de información generada en el juego. Se escoge por presentar una integración mejor con el motor de juegos que bases de datos de terceros como MySQL o SQLite |
| **Recomendaciones para su adquisición** | Se debe tener en cuenta que el almacenamiento se realiza en archivos xml dentro del sistema operativo, para Mac OS en la carpeta ~/Library/Preferences. |
| **Sitio / Origen del componente** | Se encuentra integrado a Unity. |

|  |  |
| --- | --- |
| **Nombre del Componente** | Unity Analytics |
| **Justificación de compra o reuso** | Se necesita un mecanismo para facilitar la analítica de la información obtenida en el juego, para luego aplicar los algoritmos de inteligencia artificial. Este paquete es gratuito y se encuentra integrado a Unity, por lo que hay menos riesgos de integración. |
| **Recomendaciones para su adquisición** | Se debe activar su uso en Unity para poder utilizarlo.  Está soportado para Mac. Recibe datos del sistema, más la actualización de la analítica no necesariamente se observa en tiempo real, puede tomar horas en actualizar. |
| **Sitio / Origen del componente** | Se encuentra integrado a Unity. Más información en:  <https://unity3d.com/es/solutions/analytics> |

|  |  |
| --- | --- |
| **Nombre del Componente** | Controlador MIDI |
| **Justificación de compra o reuso** | Los instrumentos musicales necesitan de dicho controlador para comunicarse con un ordenador. |
| **Recomendaciones para su adquisición** | Los instrumentos a usar lo deben tener integrado, de lo contrario se debe cambiar de instrumento. |
| **Sitio / Origen del componente** | Los productores de instrumentos lo integran a dicho instrumento. |

## Vista de Desarrollo

### Module Structure

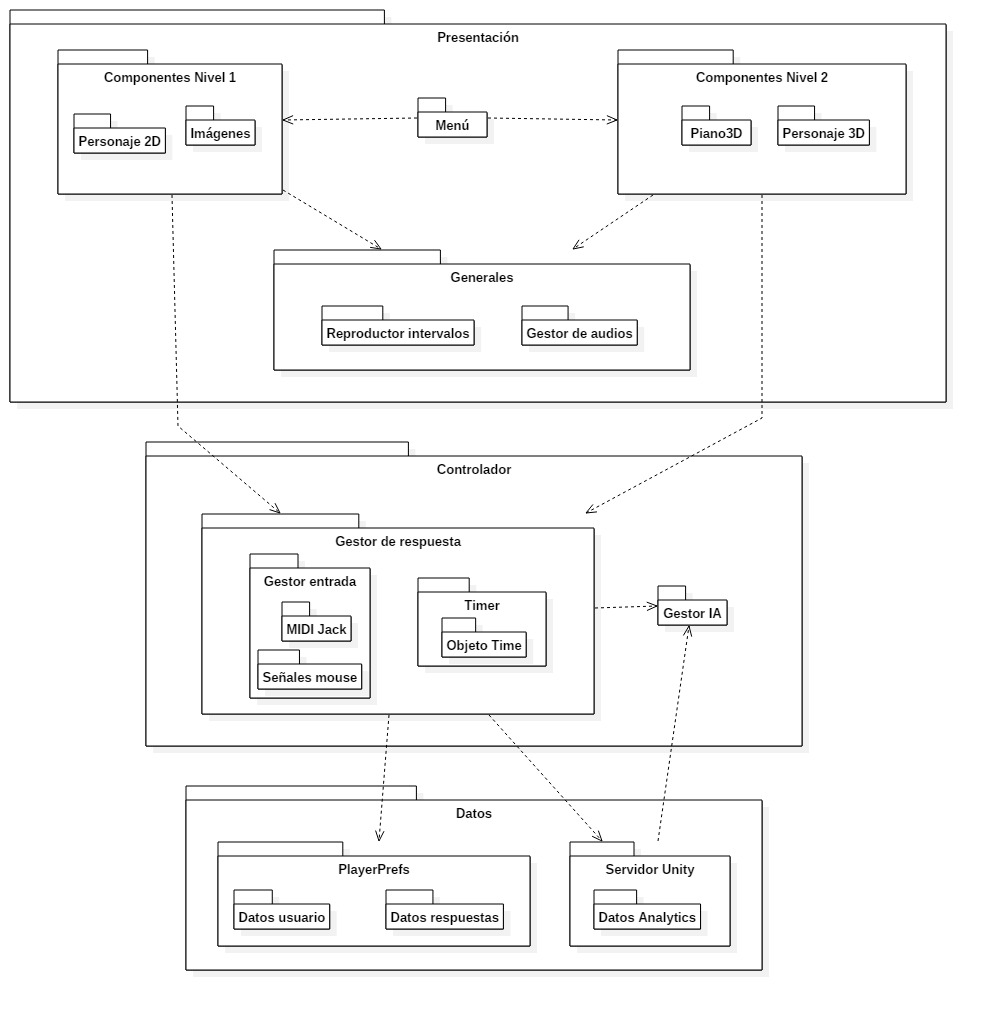


Figura 7. Diagrama de la estructura de módulos

# Atributos de Calidad del Sistema

## Rendimiento

|  |  |
| --- | --- |
| **Requirement** | **How Met** |
| **El tiempo de procesamiento de la respuesta dada por el usuario debe ser menor a 1 segundo** | Para logralo se hace uso del patrón Observer, haciendo más eficiente la comunicación entre scripts dentro de Unity |
| **Los fotogramas por segundo (FPS) debe presentar un valor igual o mayor a 30 fps** | Se deben mantener niveles óptimos de rendimiento para que el juego ejecute de forma fluida para el usuario. |
| **La recomendación que se le va a realizar al usuario debe procesarse en menos de 3 segundos.** | El algoritmo de clusterización debe almacenar la información para hacer más rápido el proceso. El algoritmo de predicción debe ser optimizado para ser lo más óptimo posible. |

## Seguridad*.*

|  |  |
| --- | --- |
| **Requirement** | **How Met** |
| **Todos los usuario deben registrarse la primera vez que acceden al juego** | Se debe realizar almacenamiento de esta información en el equipo. |
| **Los archivos almacenados en el equipo, deben estar cifrados** | Los archivos almacenados por medio de Player Prefs deben ser cifrados para evitar modificación por el usuario. |

## Otros Atributos de Calidad

### Usabilidad

El juego debe tener alta usabilidad, para ello se tendrá en cuenta:

* Efectividad del aprendizaje: Los usuarios que usen la aplicación deben presentar en promedio una menor curva de aprendizaje de los tipos de intervalos que aquellos que no usan la aplicación.
* Calidad del audio de las notas: Los audios usados para la reproducción de notas e intervalos, deben ser aprobados por el profesor Rodrigo Henao del departamento de música de la universidad.
* Menú inicial: El sistema debe de proveer de un menú lo suficientemente intuitivo de forma que el usuario, con sólo una leída, sea capaz de guiarse por la aplicación.
* Instrucciones: Cada nivel debe tener las instrucciones correspondientes, de forma que el usuario tenga los conocimientos necesarios para jugar sin necesidad de realizar un proceso iterativo de ensayo y error para aprender la dinámica. Con sólo leer las instrucciones una vez, el usuario debe estar en capacidad de jugar sin problemas.
* Interfaz amigable: El usuario debe ser capaz de reconocer todos los elementos visuales presentados y entender su función con sólo verlos o después de leer las instrucciones una sola vez.
* Retroalimentación: El usuario debe poder reconocer si realiza una acción correcta o no, sólo con la visualización de las acciones presentadas por el sistema.
* Dinamismo: El usuario debe poder usar el sistema por aproximadamente una hora y media, sin distraerse por un tiempo total de 15 minutos.