# Marco teórico.

En este apartado del proyecto se pretende describir los conceptos fundamentales con los que se va trabajar, así que será el punto de partida para todos los desarrollos teóricos y prácticos posteriores. También sentará las bases para lo que se realizará en los siguientes puntos.

El marco teórico se divide en tres apartados:

1. En la primera parte se van a presentar tres estudios relacionados con el proyecto así los conceptos que se van a manejar durante el desarrollo del trabajo.
2. La parte central del marco teórico se dedicará a definir unos criterios de selección en base a los cuales se van a puntuar los **frameworks** que se analizarán en el cuerpo del trabajo.
3. Con los criterios ya definidos, se va realizar una primera selección de **frameworks**. En esta preselección se van a reducir todo lo posible el número de programas a analizar en el cuerpo del trabajo y excluir aquellos que tengan incompatibilidades con la finalidad del proyecto o impidan la consecución de los objetivos.

## Estudios previos

### Dev Master.

Dev Master es una base de conocimiento web de motores de videojuegos, herramientas, librerías accesorias, compañías y profesionales dedicados al sector de la programación de videojuegos. En ella se encuentran más de 350 motores.

Cada sección dedicada a un motor gráfico cuenta con todos sus datos (sitio web, propietarios, licencia y fecha de lanzamiento entre otros),sus datos técnicos (por ejemplo su lenguaje de programación, motores de físicas o shaders que usa), los libros escritos sobre el mismo relacionados y un apartado para que los usuarios y profesionales puedan dejar sus comentarios y análisis sobre el programa.

La principal diferencia con el estudio del presente proyecto radica en los siguientes aspectos fundamentales:

1. Dev Master trata únicamente de motores gráficos y no también de frameworks.
2. No compara los motores entre si como se pretende hacer en este proyecto, sino que se limita a listar todas sus características y opiniones de los usuarios.

|  |  |
| --- | --- |
|  |  |
| Tipo | Proyecto Web |
| Año | 2017 |
| Autor/es | Colaborativo |
| Link | http://devmaster.net/ |

Tabla 1: Dev Master

### Comparison and evaluation of 3D mobile game engines.

Se trata de una tesis realizada por Akekarat Pattrasitidecha para la universidad de Gotemburgo y que trata de encontrar el mejor programa para desarrollar videojuegos en tres dimensiones para dispositivos móviles.

Al buscar y comprar entre motores 3D centra la mayoría del trabajo en los aspectos técnicos de los motores de juego como las físicas 3D, los efectos de partículas o el video en 3D dejando los aspectos como la usabilidad y la facilidad de uso en un segundo plano al contrario que en el este proyecto. Además se limita únicamente a motores de juego para dispositivos móviles y no abarca todos aquellos para escritorio y consolas como si se valora en este estudio.

|  |  |
| --- | --- |
|  |  |
| Tipo | Tesis |
| Año | 2014 |
| Autor/es | Akekarat Pattrasitidecha |
| Link | http://publications.lib.chalmers.se/records/fulltext/193979/193979.pdf |

Tabla 2: Comparison and evaluation of 3D mobile game engines

### Comparación de motores de videojuegos para la creación de juegos serios.

Proyecto de fin de grado presentado por dos estudiantes de la Universidad Tecnológica de Pereria (Colombia), que trata de buscar y definir unos criterios tanto teóricos como tecnológicos que tiene que tener un de videojuego para poder desarrollar un videojuego profesional.

También tiene un apartado dedicado a pequeño análisis y comparación de motores de juego aunque el objeto de estudio solo son tres motores: JMonkey, Unreal Engine y Unity 5.

Por limitarse a solo el estudio de tres motores y estar centrado en definir los requisitos de un motor para crear un videojuego profesional, cuando en el caso del presente proyecto se desea abarcar proyectos de todo tipo, se diferencia bastante del este estudio.

|  |  |
| --- | --- |
|  |  |
| Tipo | Tesis |
| Año | 2015 |
| Autor/es | Cristian Steven Chavarro Rico  Mauricio Valencia Arbelaez |
| Link | http://repositorio.utp.edu.co/dspace/bitstream/handle/11059/6016/00669CH512.pdf?sequence=1 |

Tabla 3: Comparación de motores de videojuegos para la creación de juegos serios

## Conceptos previos.

### Framework.

Un **framework** en esencia es un patrón o esqueleto para el desarrollo de una aplicación, juego o contenido multimedia. Una definición más académica o formal podría ser que un **framewok** es un conjunto estandarizado de conceptos, prácticas y criterios para desarrollar y crear aplicaciones.

Los **frameworks** suele incluir:

1. Soporte de programas.
2. Librerías.
3. Lenguaje de scripting.
4. Software accesorio para desarrollar partes especificas del programa.
5. Un lenguaje de programación o interpretado.

Su uso implica un coste de aprendizaje aunque a largo plazo facilita el desarrollo, mantenimiento de la aplicación y permiten evitar trabajar en los detalles de bajo nivel permitiendo concretar esfuerzo en las partes esenciales de programa.

Desde el punto de vista del desarrollo de software, un **framework** es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado.

### SDK (Kit de desarrollo).

Un **SDK o Kit de desarrollo** es un conjunto de herramientas y programas de desarrollo de software que permite al programador crear aplicaciones para un determinado paquete y estructura de software, plataforma de hardware, sistema operativo de computadora o máquina.

Un **SDK** no es más que una **API[[1]](#footnote-1)** (interfaz de aplicaciones) que permite el uso de uno o varios (aunque esta opción es menos habitual ya que suele trabajar en exclusividad con un lenguaje) lenguajes de programación.

Los **SDK** suelen llevar incluidas herramientas que ayudan al desarrollador a crear sus aplicaciones. Entre ellas destacan las siguientes:

1. Debug.
2. Soporte para le detección de errores de programación.
3. Códigos de ejemplo y documentación.
4. Entorno de programación **IDE[[2]](#footnote-2).**

### Game Engine (Motor de videojuegos).

El **Game Engine o Motor de videojuegos** proporciona un entorno de desarrollo integrado para la creación de juegos, no es más que un framework diseñado para la creación y desarrollo de juegos.

Un motor de juego suele incluir diferentes componentes que se corresponden con los elementos de un videojuego. Los elementos que suele incluir entre otros son los siguientes:

1. **Programa del juego principal.**
2. **Motor de render.**
3. **Motor de audio.**
4. **Motor de físicas.**
5. **Motor de IA.**
6. **Motor de Red.**

Antes de los Game Engine se escribían como entidades singulares y no reutilizables. El término motor de juego apareció a mediados de los 90 ligado al crecimiento de los juegos 3D especialmente shooters en primera persona, ya que permitía en juegos muy complejos técnicamente no comenzar desde cero en cada nuevo desarrollo.



Figura : Wof3d Pimer motor de juegos 3D

## 3.2 Criterios de selección

Para el presente proyecto una de las partes fundamentales es seleccionar un framework, motor gráfico o SDK, sobre el que realizar todos los objetivos que se pretenden alcanzar. Para seleccionar el programa se deben definir previamente unos criterios que permitan que esta elección sea lo más correcta posible ya que de ella depende en gran medida el éxito del proyecto.

Estos criterios se han pensado concretamente para cumplir los objetivos y las finalidades del proyecto y no servirían por lo tanto para otros propósitos que no sean los propios del presente proyecto.

Posteriormente a la determinación de estos criterios se le otorgará a cada uno de ellos una puntuación máxima y mínima que se podrá otorgar en la fase de análisis a cada uno de los motores, sdk o frameworks candidatos a ser seleccionados. La suma de estas puntuaciones determinará cuál de estos programas se va usar para realizar el proyecto.

Se va usar una escala de 100 para definir la puntuación ya que da más facilidades a la hora de visualizar las mismas y comparar los distintos programas entre si. No todos los criterios tienen la misma puntuación ya que no todos tienen la misma importancia, ni los mismos elementos.

En la siguiente tabla se encuentran definidas las distintas puntuaciones que pueden tener los elementos de cada criterio y su valoración en cuanto a su importancia y denominación.

|  |  |  |
| --- | --- | --- |
| Puntuación | Importancia | Explicación |
| 0-1 | Mínima | Elemento sin mucha importancia pero que hay tener en cuenta y valorarlo. |
| 0-2 | Poca | Apartado con cierta importancia pero que se desea que no influya demasiado en el resultado final |
| 0-4 | Normal | Los criterios en este apartado tienen una importancia standard, puntuación por defecto |
| 0-6 | Alta | Elementos bastante importantes y que deben contar mucho a la hora de seleccionar un programa |
| 0-8 | Clave | Elementos clave y fundamentales. |
| >10 | Especial | Se detallará en cada caso si hay alguna puntuación de este tipo. |

Tabla 4: Clasificación de puntuaciones

### Criterio 1: Requisitos e instalación.

El primer criterio que hay tener en cuenta a la hora de analizar los diferentes motores es el requisito de funcionamiento e instalación.

En cuanto a los requisitos de instalación, dado el carácter generalista del presente proyecto, se valorará con mayor puntuación a aquellos programas En cuanto a los requisitos de instalación, dado el carácter generalista del presente proyecto, se valorará con mayor puntuación a aquellos programas que necesiten menos recursos del sistema para funcionar y que además funcionen en más sistemas operativos.

Es importante que el proceso de instalación sea tenido en cuenta, para comprobar la sencillez del mismo, la posible necesidad de instalación de otros programas o librerías accesorias y una buena guía al usuario durante todo el proceso. En este apartado será también será considera el tiempo que dura el proceso.

A continuación se muestran los apartados que compone este criterio y su importancia.

|  |  |  |
| --- | --- | --- |
| CRITERIO | IMPORTANCIA | PUNTUACIÓN MAX-MIN |
| Requisitos de instalación. | Normal | 0-4 |
| Instalación. | Poca | 0-2 |

Tabla 5: Criterio: Instalación y requisitos

#### Medición de la puntuación: Instalación.

El apartado instalación se medirá con la realización de encuestas en las que se solicitará a cada participante que mida el tiempo que tarda en todo el proceso de descarga e instalación del programa en su equipo y posteriormente se le hará las siguientes preguntas:

1. **Tiempo total de instalación.**
2. **Valoración de la complejidad del proceso:** de instalación en una escala de 10 donde 10 es sencillo y cero es complicado.
3. **Valoración del proceso de instalación:** Donde se va tratar de medir si los pasos en la instalación están bien explicados, donde 1 es que es poco o nada explicados y 10 que el programa guía al usuario de forma correcta y explica todos los pasos que va a realizar.
4. **Campo adicional: Librerías adicionales.** Donde el usuario deberá indicar si el programa le ha solicitado la instalación e alguna librería o programa adicional para funcionar.
5. **Campo adicional: Fallo de instalación.** Si el programa falla durante la instalación el usuario deberá indicarlo en este campo e indicar el error aparecido para posteriormente intentar determinar que ha podido ocurrir y si esto puede restar puntuación al candidato.

### Criterio 2: Documentación, tutoriales y comunidad.

Como el presente proyecto pretende ser de alcance mayoritario, sin necesidad de conocimientos previos de ingeniería o programación, es fundamental que el programa elegido cuente con una buena documentación, tutoriales y una gran comunidad de usuarios.

En este apartado se incluyen toda la documentación oficial proporcionada por los propietarios del software y su soporte oficial, los tutoriales Tanto los proporcionados de forma oficial como los realizados por usuarios en plataformas de video online, como You Tube o Vimeo,. en blog especializados y las comunidades o foros de desarrolladores que trabajan con el programa.

El pilar fundamental de este criterio es el análisis de los tutoriales oficiales. Estos tutoriales son la primera aproximación de los usuarios al programa y por lo tanto deben contar con todos los elementos para que se puedan llevar a cabo de manera correcta, rápida y entendible. Se necesita un programa con unos tutoriales que aporten conocimiento e inciten a seguir aprendiendo y desarrollando habilidades, ya que se pretende que el presente proyecto sea un punto de partida a un aprendizaje en profundidad del programa y por lo tanto es necesario contar con los recursos necesarios.

Se otorgará puntuación extra a aquellos programas que tengan documentación, foros y tutoriales en castellano.

Los subapartados que componente este criterio son los listados en la siguiente tabla.

|  |  |  |
| --- | --- | --- |
| CRITERIO | IMPORTANCIA | PUNTUACIÓN MAX-MIN |
| Documentación, foros y tutoriales oficiales | Normal | 0-4 |
| Análisis de los tutoriales oficiales | Clave | 0-8 |
| Comunidades y foros de desarrolladores. | Normal | 0-4 |

Tabla 6: Criterio: Documentación, tutoriales y comunidad

#### Medición de la puntuación: Documentación, tutoriales y comunidad.

Al tratarse de un apartado clave en la elección del framework , hay que tener muy claro cómo se va medir la puntuación del mismo.

Respecto al subapartado **documentación, foros y tutoriales oficiales** se va otorgar la puntuación en función de los recursos aportados por los propios desarrolladores del programa o por terceros pero que sean considerados oficiales por los propietarios del framework. Se tendrán en cuenta todos los elementos puestos a disposiciones a disposición de los usuarios, independientemente de su naturaleza o formato aunque se presentará especial atención al manual oficial, los tutoriales y si cuentan o no con un foro oficial.

En subcriterio anterior se medía la cantidad de recursos que se ponen a disposición de los usuarios de forma oficial, el subcriterio siguiente, **análisis de los tutoriales oficiales,** se trata de comprobar la calidad de estos recursos. Para medir este apartado se realizará uno o varios tutoriales oficiales y se analizará para comprobar si tienen o no errores, si su implementación es correcta, clara e entendible o si consiguen el objetivo de formar al usuario en el uso del programa e incitarla a continuar usando el programa.

Por último y para completar a los dos subapartados anteriores, nos encontramos con el subcriterio **comunidades y foros de desarrolladores** **no oficiales**. Para concretar la puntuación en este subapartado, se va comprobar la presencia de comunidades activas en Reddit[[3]](#footnote-3) , ya que es foro con mayor cantidad de usuarios a nivel mundial, y en Taringa, el equivalente en idioma español a R[[4]](#footnote-4)eddit.

### Criterio 3: Funcionamiento y dificultad de uso.

El funcionamiento general del programa trata de detectar fallos técnicos tanto en la fase de instalación como posteriormente durante el uso del programa tales como errores que impiden la ejecución, el manejo o la realización de las distintas acciones y la estabilidad del programa.

Se puede decir que es una forma de medir si el programa se adecua a los estándares que un usuario puede comprender y usar de una manera rápida y efectiva consiguiendo sus propósitos funcionales.

La curva de aprendizaje es un apartado importante para decidir el programa va ser objeto de las acciones de este estudio ya que pretende ser un proyecto que en algunas partes del mismo, como puede ser las guías de aprendizaje puedan ser realizados posteriormente por usuarios de todos los niveles.

Por esta razón se busca un programa en el que la curva de aprendizaje no sea demasiado alta y pueda ser asumida por un usuario medio. Respecto a este criterio no se trata de la curva para aprender a dominar todos y cada uno de los aspectos del programa, sino de poder empezar a trabajar con el programa y crear pequeños proyectos, es decir, el foco está en el comienzo de la curva de aprendizaje.

Se busca por lo tanto un programa que sea muy sencillo en los primeros momentos y de adaptación muy rápida por parte de los usuarios sin importar tanto la dificultad de dominar todos los aspectos del mismo.

Los criterios que componentes este apartado de análisis son los mostrados en la tabla siguiente.

|  |  |  |
| --- | --- | --- |
| CRITERIO | IMPORTANCIA | PUNTUACIÓN MAX-MIN |
| Funcionamiento General | NORMAL | 0-4 |
| Primeros Pasos: Creación de un pequeño proyecto. | NORMAL | 0-4 |
| Funciones básicas | OTROS\* | 0-12 |
| Lógica de programación necesaria | NORMAL | 0-4 |

Tabla 7: Criterio: Funcionamiento general y curva de aprendizaje

#### Medición de la puntuación: Funcionamiento y dificultad de uso.

Dado que este criterio tiene un carácter puramente práctico se han de establecer las siguientes reglas o patrones para que la comparación entre los distintos programas sea lo más ecuánime posible, justa y adecuada para el propósito general del programa.

Detectar la curva de aprendizaje de un programa es una tarea bastante complicada de por sí, ya que entran parámetros subjetivos de la persona que está intentando identificar dicha curva de dificultad. Además en el caso que ocupa el presente trabajo presenta otra dificultad añadida, al tratarse de frameworks que tiene un funcionamiento en muchos apartados similar no se va apreciar la misma dificultad en el primer programa analizado que en los siguientes viendo ésta continuamente disminuida. Para mejorar ésta criterio que es posiblemente uno de los más importantes dentro de la comparación se van a realizar las siguientes acciones:

1. Para cada programa se va realizar uno o varios tutoriales a ser posible oficial que consistan en la creación de un pequeño juego hasta completar aproximadamente 4 horas de trabajo con ellos. Documentando las dificultades que se van encontrando (en el caso que se trate de tutoriales oficiales los fallos o malas explicaciones en estos ya que afectan a otros criterios del estudio). Con lo que el apartado de “Creación de un pequeño proyecto” estará cubierto.
2. Una vez completado este tiempo se estará en disposición de indicar una puntuación para los primeros pasos donde cero es la máxima dificultad y cuatro es muy sencillo.
3. Con los conocimientos adquiridos en las horas dedicadas a la realización de un pequeño proyecto vía tutoriales se podrán definir como se hacen las funciones básicas de un juego y su dificultad, las funcionalidades básicas que se tendrán en cuenta son las siguientes: Movimiento de personaje por el escenario, mecánica básica de disparo, salto, colisión con otro objeto y desaparición de objeto colisionado.
4. La puntuación del apartado “Creación de un pequeño proyecto” se asignará en función del nivel de complejidad conseguido el tiempo dedicado a la realización de los tutoriales oficiales, también influirá la cantidad de contenido creado o si han podido realizar diferentes tipos de juegos.
5. A las puntuaciones de este apartado se les va aplicar un factor corrector incremental que reducirá las puntuaciones en función de la posición en que se han realizado. **FACTOR multiplicador de 1 que se irá reduciendo en 0.1 en cada programa.** Estos frameworks pese a ser distintos tienen bastantes aspectos en común en su funcionamiento por lo que la percepción de dificultad se reducirá conforme se vayan analizando programas , por lo que es fundamental contar con el factor de corrección para equilibrar las puntuaciones y que los programas analizados en los últimos lugares no se vean beneficiados por el conocimiento adquirido en el uso este tipo de herramientas.

### Criterio 4: Editor Gráfico/Visual.

El contar con un editor visual para poder posicionar los objetos, personajes en el mundo y con el que poder crear relaciones entre ellos sin necesidad de *escribir código* y disponiendo solo de una edición gráfica es un punto esencial para seleccionar un framework u otro.

Se valorará no solo que tenga un editor gráfico sino que éste sea accesible, sencillo e intuitivo para los usuarios. Se tendrá en cuenta también el funcionamiento general, es decir, que no tenga bugs ni errores que impidan su funcionamiento o que dificulten el proceso de creación y desarrollo de los proyectos.

La usabilidad[[5]](#footnote-5) en el funcionamiento de la interfaz y de sus elementos también será un aspecto a tener en cuenta.

La tabla siguiente muestra la puntuación del presenta apartado.

|  |  |  |
| --- | --- | --- |
| CRITERIO | IMPORTANCIA | PUNTUACIÓN MAX-MIN |
| Editor/Gráfico visual | Clave | 0-8 |

Tabla 8: Criterio: Editor gráfico/visual

#### Medición de la puntuación: Editor Gráfico/Visual.

Este criterio tiene en la mayoría de sus aspectos un carácter subjetivo, por lo que la mejor forma de valorarlo es a través de realización de encuestas, una por cada programa, más una extra para comparación.

La muestra que van a realizar el estudio la compondrán personas con conocimientos en distintos campos tanto de la ingeniería informática y multimedia, como del diseño y profesionales dedicados al sector tecnológico.

La composición profesional de la muestra es la que muestra la siguiente tabla:

|  |  |
| --- | --- |
| Profesión/formación | Número de individuos. |
| Ingenieros Multimedia (Rama videojuegos) |  |
| Ingenieros Multimedia (Rama web) |  |
| Ingenieros Informáticos |  |
| Profesionales del sector tecnológico. |  |
| Arquitectos. |  |

Tabla 9: Participantes en la encuesta

El proceso que se va a seguir para la valoración de estos criterios por la muestra escogida será el siguiente y tendrá las siguientes características:

1. El estudio tendrá en cuenta solo las encuestas de los usuarios que completen las 5 encuestas que componen el estudio, esto se hace para mantener una coherencia y cohesión en los datos, los mismos usuarios valoran distintas interfaces.
2. El proceso tendrá una duración aproximada de un mes enviándose a los participantes una encuesta cada semana.
3. Cada encuesta tendrá un encabezado para que el usuario se identifique con un nombre, apodo o de cualquier forma que una vez completado todo el estudio se pueda recolectar los resultados tanto del conjunto como de cada encuestado individualmente, además es requisito para cumplir el punto número 1.
4. En la primera de las encuestas se integrará un apartado para que el usuario indique las características del equipo en el que va a realizar las pruebas y que va ser el mismo para todas las pruebas.
5. Los apartados que los encuestados deben valorar para determinar la puntuación de cada programa tienen una puntuación de entre 1 y 5 siendo cinco el máximo.
6. Para elapartado **editor visual intuitivo** se pedirá a los usuarios que valoren:   
   -A priori parece un programa sencillo de utilizar.  
   -Los iconos representan bien los elementos.
7. En el caso de valorar si el editor es visualmente atractivo, con un aspecto cuidado y actual los apartados a puntuar son los siguientes:  
   -La interfaz es atractiva.  
   -Tiene un aspecto cuidado y moderno.  
   -Puntúa el diseño del programa.
8. El buen funcionamiento del programa se medirá con los siguientes apartados:  
   -Rapidez al iniciarse.  
   -Consumo de recursos del programa.  
   -El programa se muestra rápido.

Existen dos apartados no obligatorios y no cuantificables, donde el usuario puede añadir información adicional que posteriormente si son completados habrá que valorar, son los siguientes apartados:  
-Un apartado destinado a que el usuario indique si el programa no ha funcionado bien los errores que ha tenido. Si se produce algún caso de este tipo influirá negativamente en la valoración del funcionamiento general.  
-Un campo para que el usuario valore con palabras la interfaz, que después habrá que interpretar.

### Criterio 5: Escalabilidad.

El presente proyecto en su apartado de guía de uso pretende ser una iniciación al uso de un framework de videojuegos tanto para usuarios expertos en programación como para usuarios medios.

Pero los conocimientos adquiridos durante la realización del mismo deben servir tanto para realizar pequeños proyectos como los que aquí se proponen como para futuros juegos más grandes y ambiciosos. De esta forma el programa seleccionado debe tener una gran escalabilidad que permita afrontar problemas mayores y seguir progresando en cuanto a la complejidad de los proyectos creados.

Los dos apartados que componente este criterio son los mostrados en la siguiente tabla y su puntuación es la siguiente.

|  |  |  |
| --- | --- | --- |
| CRITERIO | IMPORTANCIA | PUNTUACIÓN MAX-MIN |
| Posibilidad de crear funciones y scripts propios y/o lenguaje de programación propio. | Clave | 0-8 |
| Add-ons y plug-ins | Alta | 0-6 |

Tabla 10: Criterio: Escalabilidad

### Criterio 6: Resultados profesionales.

Los resultados que otros desarrolladores y/o creadores han conseguido con los diversos frameworks también es un criterio importante a la hora de seleccionar aquél sobre el que se va realizar las acciones del presente proyecto.

Elementos como la implantación de dichos motores en el mercado de juegos profesionales y los acabados gráficos y/o estéticos que se pueden conseguir con ellos son esenciales para que el motor sea adecuado para este estudio, ya que como se ha comentado pretende fomentar posteriormente un uso más profesional de dicha herramienta.

Un aspecto muy importante en este apartado es la diferenciación de los juegos profesionales o la no repetición de patrones visuales. Esto hace referencia a que los proyectos puedan ser tan diferentes tanto visualmente como funcionalmente de forma que no se parezcan entre ellos y se note la procedencia de los mismos. Es fundamental que el programa evite en la medida de lo posible que el usuario pueda detectar con un vistazo rápido con que herramienta ha sido desarrollado el juego. Se habla aquí de la primera sensación y de carácter usuario ya que un desarrollador experto puede detectar el motor gráfico.

En la tabla mostrada a continuación se encuentran los subapartados de este criterio, su importancia y su puntuación.

|  |  |  |
| --- | --- | --- |
| CRITERIO | IMPORTANCIA | PUNTUACIÓN MAX-MIN |
| Principales juegos desarrollados | Clave | 0-8 |
| Diferenciación | Normal | 0-4 |

Tabla 11: Criterio: Resultados profesionales

### Criterio 7: Estado actual.

A la hora de seleccionar un framework, se debe que tener en cuenta el estado actual del mismo. Se trata de encontrar un framework que permita un uso prolongado en el tiempo y para ello debe estar activo, es decir, no abandonado por sus desarrolladores. Se valorará también la frecuencia de actualizaciones ya que permite detectar que los bugs y problemas que surgen en todo programa con su uso se van solucionando, además que estas nuevas versiones permiten incluir nuevas funcionalidades.

Los elementos que componen el estado actual son los siguiente:

|  |  |  |
| --- | --- | --- |
| CRITERIO | IMPORTANCIA | PUNTUACIÓN MAX-MIN |
| Estado actual | Normal | 0-2 |
| Frecuencia de actualizaciones. | Mínima | 0-1 |
| Fecha desde la última actualización. | Mínima | 0-1 |

Tabla 12: Criterio: Estado actual

### Criterio 8: Sistemas operativos soportados y requerimientos.

El carácter mayoritario y generalista ya comentado del presente trabajo, aconseja tener en cuenta para la selección de la herramienta de desarrollo los requisitos y requerimientos del sistema para hacer funcionar los juegos. Estos deben ser adecuados para llegar al mayor número posible de usuarios teniendo en cuenta tanto el hardware como el software y sistema operativo.

En la puntuación de este criterio tendrán una gran influencia positiva aquellos programas aquellos programas que posibiliten la exportación de sus juegos a un mayor número de dispositivos posibles. En este apartado no se valora si esta exportación a los diversos dispositivos y sistemas operativos es una función propia del programa base y viene incluida en su licencia principal o necesita módulos especiales de pago o gratuitos para llevarla a cabo, ya que esto ya será tenido en cuenta en el criterio de licencia.

La siguiente tabla muestra la importancia y puntuación de este criterio.

|  |  |  |
| --- | --- | --- |
| CRITERIO | IMPORTANCIA | PUNTUACIÓN MAX-MIN |
| Posibilidades de exportación: Sistemas y plataformas | Clave | 0-8 |

Tabla 13: Criterio: Sistemas operativos soportados

# Criterio 9: Licencia.

El precio del software necesario y el tipo de licencia de cada programa a analizar también es un punto importante a la hora de seleccionar el mismo.

Al pretender ser un proyecto abierto y generalista tendrán mejor puntuación aquellas herramientas con un precio menor o aquellas que sus versiones gratuitas cuenten con unas menores limitaciones de uso.

Pese a ello, este criterio tiene menos importancia que otros ya que no se debe desvirtuar el objeto del estudio, ni que por seleccionar un software de menor precio o licencia con menos limitaciones no se llegue a la consecución de determinados objetivos o no se completen de manera satisfactoria.

|  |  |  |
| --- | --- | --- |
| CRITERIO | IMPORTANCIA | PUNTUACIÓN MAX-MIN |
| Precio de Licencias | Normal | 0-4 |
| Modo de prueba o free | Poca | 0-2 |
| Precio módulos de exportación | Poca | 0-2 |

Tabla 14: Criterio: Licencias

### Puntuación total.

|  |  |
| --- | --- |
| CRITERIO | PUNTUACIÓN MIN-MAX |
| Requisitos e instalación | 0-6 |
| Documentación, tutoriales y comunidad | 0-16 |
| Funcionamiento y dificultad de uso | 0-24 |
| Editor Gráfico/Visual | 0-8 |
| Escalabilidad | 0-14 |
| Resultados profesionales | 0-12 |
| Estado actual | 0-4 |
| Sistemas operativos soportados y requerimientos. | 0-8 |
| Licencias | 0-8 |
| TOTAL | 0-100 |

Ilustración : Puntuación Total

## 3.2 Primera selección de framework.

Antes de proceder a un análisis de los posibles candidatos, se ha realizado una primera lista de posibles motores, que pueden cumplir los criterios referidos en el punto anterior y ser el medio con el que se van a cumplir los objetivos del presente trabajo.

Esta primera lista parte de la búsqueda en general de motores gráficos y frameworks para desarrollar juegos 2D, sin entrar a valorarlos con los criterios anteriormente citados.

De ellos se descartaran automáticamente aquellos que posean características incompatibles con los objetivos planteados, aquellos que se vea sin necesidad de un estudio completo que no cumplen los requisitos buscados o aquellos que por otros motivos justificados no pueden ser una opción.

Los siguientes programas mostrado en la tabla son los que más se ajustan a priori al objeto de estudio del proyecto y por lo tanto se consideran a priori candidatos a ser el seleccionado para el mismo.

|  |  |
| --- | --- |
| Click Team Fusion 2.5 | Torque 2D |
| RPG-Maker | Gdevelop |
| Unreal 4 2D | Game Maker |
| Cocos2D | GODOT |
| Construct 2 | Marmalade |
| Eclipse Origins | Unity 2D |
| Adventure Game Studio | Stencyl |
| Angel 2D |  |

Tabla 15: Primera selección de motores

### Motores descartados y justificación

De la tabla mostrada anteriormente se han descartado una serie de programa a continuación se explican las razones de la exclusión de cada uno de ellos.

#### Resultado de imagen de torque 2d logoTorque 2D

Figura 2: Torque 2D

Tabla 16: Logo Torque 2D

Torque 2D o Torque Game Builder es la adaptación a 2D del motor gráfico Torque 3D creado por la compañía GarageGames para su juego Tribes 2 en 2001. Esta versión 2D del motor fue lanzada en 2013 bajo licencia MIT[[6]](#footnote-6) lo que le confiere un punto a favor al tratarse de software libre.

Pese a ser a contar con una licencia libre Torque 2D tiene la posibilidad de comprar una interfaz de desarrollo Torque Studio.

Después de una primera aproximación se ha podido comprobar que la instalación y configuración es demasiado compleja para el usuario medio, la creación de proyectos es excesivamente tediosa y es prácticamente indispensable contar con conocimientos de programación para poder crear contenido por lo que se ha descartado.

#### Unreal 4 2D

Unreal 4 es la evolución del motor desarrollado en 1998 para el juego del mismo nombre y que bajo él y sus constantes evoluciones se han desarrollado juegos tan populares como Deus Ex, Bioshock, Batman: Arkham Asylum o Mass Effect.

Tabla 17: Unreal 4

Figura 3: Unreal Engine

Como pasara en su día que evolucionó desde un motor exclusivo de juegos en primera persona o shooters a abarcar también juegos tercera persona ahora en sus últimas versiones intenta también dar soporte a juegos 2D.

Aunque su potencia es muy grande y se pueden realizar grandes proyectos con él toda su interfaz está pensada para elementos tres dimensiones con lo que en algunos aspectos resulta bastante complejo el manejo de las dos dimensiones, además es un motor bastante exigente respecto a los requerimientos del sistema tanto para trabajar con el cómo en los proyectos resultantes.

#### Cocos2D.

Cocos2D es un framework para diseñar y crear juegos 2D inicialmente basado en Phyton aunque diferentes versiones lo han adaptado a diferentes lenguajes de programación como por ejemplo la versión X o la versión Js que dan soporte a C++ y Javascript respectivamente.

Figura 4: Cocos2D

Cocos2D se descarta en esta primera fase ya que aunque posee una creación rápida de proyectos, un manejo fácil y rápido de Sprites es imprescindible programar ya sea en Phyton con la versión original o en los lenguajes propios de otras versiones.

Ilustración 2: Cocos 2D

Cuenta además con la desventaja de no contar con una interfaz visual de manejo, creación y posicionamiento de los elementos, su funcionamiento es similar a motores como SFML, en los cuales tienes que llamar desde código a las funciones propias del mismo para realizar diversas acciones, pero añadiéndole su propio entorno de programación.

#### GODOT Engine.

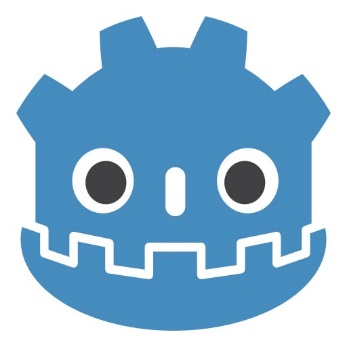
A GODOT Engine es un motor de videojuegos multiplataforma y de código abierto. Su funcionamiento basado en nodos y escenas lo diferencia en gran medida de los motores y framworks analizados en el presente proyecto.

Figura 5: GODOT Engine

Uno de los objetivos de este proyecto es que usuarios no programadores se acerquen al desarrollo de videojuegos. Puedan crear sus primeros proyectos, para posteriormente con la base adquirida poder aumentar su conocimiento, esto incluye poder utilizar cualquier otro motor o framework que no sea el seleccionado, por lo que uno que trabaje de forma tan distinta y diferencia del resto no es un buen candidato.

A parte de su funcionamiento diferente nos encontramos también en un caso como el de Cocos 2D, donde necesitamos nociones de programación para poder realizar proyectos en GODOT.

#### Marmalade.

Marmalade es un SDK y motor gráfico basado en C++. Es uno de los SDK´s de desarrollado más utilizados por los grandes estudios para crear juegos 2D y 3D , juegos como Lara Croft and the Guardian of Light, Plants vs Zombies , Cut the Rope o el modo Zombies 2D de Call of Duty: World at War. Ha sido premiado en los Games Industry Awards 2015 en la categoría Engine and Middleware, Tools and Tech.

Figura 6: Marmalade

Pese al gran potencial de Marmalade y las cotas de calidad que pueden alcanzar sus juegos, su entorno basado en el lenguaje C++ que se debe conocer y dominar para crear contenido y su alta curva de aprendizaje hacen que sea descartado.

#### Eclipse Game Engine.

Eclipse Origins es un motor gráfico y una interfaz o suite para la creación de MMORPG [[7]](#footnote-7) 2D. Su uso es sencillo y su interfaz visual hace bastante sencillas las tareas de creación de escenarios y colocación de Sprites.

Su limitación a un solo género, la escasa documentación y comunidades activas unido a que es motor que prácticamente ha quedado en desuso hacen que no sea un buen candidato.

#### Unity 2D

Unity es uno de los motores de videojuegos más utilizados hoy en día tanto por grandes equipos como por pequeños estudios independientes. Su gran escalabilidad, gran interfaz visual, carácter multiplataforma y el potente sistema de scripts lo convierten un motor apto para cualquier nivel usuarios.

Figura 7: Unity

Pero el problema principal que hace que sea descartado es que Unity está pensado principalmente para desarrollo de juegos 3D y aunque sus últimas versiones incorporan la posibilidad de desarrollar juegos 2D todavía no tiene toda la potencia que sí tiene para juegos 3D. Aunque no se duda de que en futuras versiones sigan mejorando en ese sentido.

Al igual que con Unreal 2D, al ser un motor 3D adaptado a 2D, es pesado y consume excesivos recursos.

#### Stencyl

Stencyl es un programa de creación de videojuegos 2D para computadoras y dispositivos móviles. Es un software gratuito con la opción de comprar módulos de pago.

Stencyl funciona dividiendo cada ciclo de ejecución en tres pasos: creació , dibujado y actualización de datos. Para las físicas hace uso del motor Box2D.

El principal inconveniente de Stencyl es que usa Flash para la exportación de sus juegos y las vulnerabilidades de Flash han hecho que la mayoría de navegadores hayan dejado de dar soporte a dicha tecnología. Aunque es conveniente seguir usando las aplicaciones en Flash incluyendo Flash Player en el sitio Web es conveniente tratar de usar otros programas que hagan uso de otras tecnologías.

#### Tabla resumen de programa descartados

La siguiente tabla muestra los motores descartados y las razones fundamentales para cada uno de ellos, algunos aparecen en varios motivos.

|  |  |
| --- | --- |
| MOTIVO | CANDIDATO EXCLUIDO |
| Complejidad | Torque2D, Unreal 2D, Marmalade. |
| Alto consumo de recursos | Unreal 2D, Unity2D |
| Necesidad de conocimientos en programación | Cocos2D, GODOT Engine. Marmalade |
| Diferencia excesiva con el resto | GODOT Engine. |
| Limitaciones | Eclipse Game Engine. |
| Escasa documentación | Marmalade. |
| Motivos de Seguridad | Stencyl |
| Programa en desuso | Eclipse Game Engine, Stencyl |

Tabla 18: Exclusiones de Frameworks

### Motores preseleccionados.

Después de los programas excluidos han quedado candidatos posibles para el objeto de nuestro proyecto. Es a estos frameworks a los que se les hará el estudio completo basado en los criterios expuestos.

#### Gdevelop

Gdevelop es un software para la creación de juegos 2D y 3D, si necesidad de utilizar ningún lenguaje de programación. Gdeveloper es un programa de código abierto y totalmente gratuito.

Figura 8: Gdevelop

El programa está escrito en C++ y trabaja fundamentalmente con la librería gráfica SFML , añadiéndole por encima una interfaz basada en WxWidgets[[8]](#footnote-8) y utiliza también TinyXMl para la gestión de mapas y Tiles.

Gdevelop trabaja en base a varios editores que abarcan todo el proceso de creación de un videojuego. Los editores con los que cuenta Gdevelop son los siguientes:

**Gestor del proyecto:** Es el editor principal del programa en el cual se indican los datos básicos del juego como pueden ser nombre o autor o pantallas de cargas. A través de este editor se accede al resto de editores. Funciona como entorno principal de trabajo.

**Editor de escenas:** Permite organizar cada escena, pantalla o nivel del juego de manera individual. En él se realiza por ejemplo el posicionamiento de objetos o enemigos. Cuenta con un depurador para comprobar el correcto funcionamiento de la escena y un *player* que posibilita probar la escena.

**Editor de objetos en la escena:** Posibilita la creación de los distintos tipos de objetos que pueden encontrarse en una escena. Permite crear distintos tipos de objetos como podrían ser Sprites, Textos u objetos 3D. En este editor se añaden todos los efectos tanto de físicas como de partículas o comportamientos predefinidos.

**Editor de las capas de la escena:**  Gdevelop puede trabajar con diferentes capas dentro de una escena, lo que ayuda a organizar y mejorar el trabajo pudiendo por ejemplo tener una capa para enemigos, otra para objetos de fondo, otros para objetos con los que poder colisionar, ….

**Editor de eventos:** Gestionan los eventos de la escena y permiten realizar las funciones propias del juegos. Este editor compone toda la capa lógica del juego y en la que se va definir toda la jugabilidad del mismo.

**Banco de imágenes:** Gestiona todas las imágenes, videos y gráficos que se van a utilizar en un proyecto.

#### Clickteam Fusion.

Click Team Fusion, apartir de ahora (CTF) es un framework para la creación de videojuegos especializada en juegos 2D lanzado en el año 2013 que actualiza la herramienta de creación y programación de juegos Multimedia Fusion que fue uno de los primeros programas que permitían crear juegos de manera visual aunque de una manera limitada.

Pero con la actualización de Multimedia Fusion a Click Team Fusion se añadieron toda una serie de herramientas que hicieron del programa una potente herramienta para crear videojuegos de manera rápida y visual. Algunas de estas características fueron por ejemplo la integración del motor de físicas Box2D que le daba a las mismas un acabado profesional, un nuevo sistema de objetos que hacía más sencilla la tarea de creación de los mismos y una nueva interfaz que llevaba la programación gráfica a un nuevo nivel no visto en las versiones anteriores.

Lo más destacable de CTF es su sistema de gestión de eventos, que constituye el núcleo central de todo el programa, en la Figura 9 se observa un ejemplo de eventos de CTF.

|  |  |
| --- | --- |
| GDEVELOP | |
| Fecha de lanzamiento | 2014 |
| Programado en | C++ |
| Lenguaje de programación | Javascript |
| Licencia | Libre |

Tabla 19: Gdevelop Datos

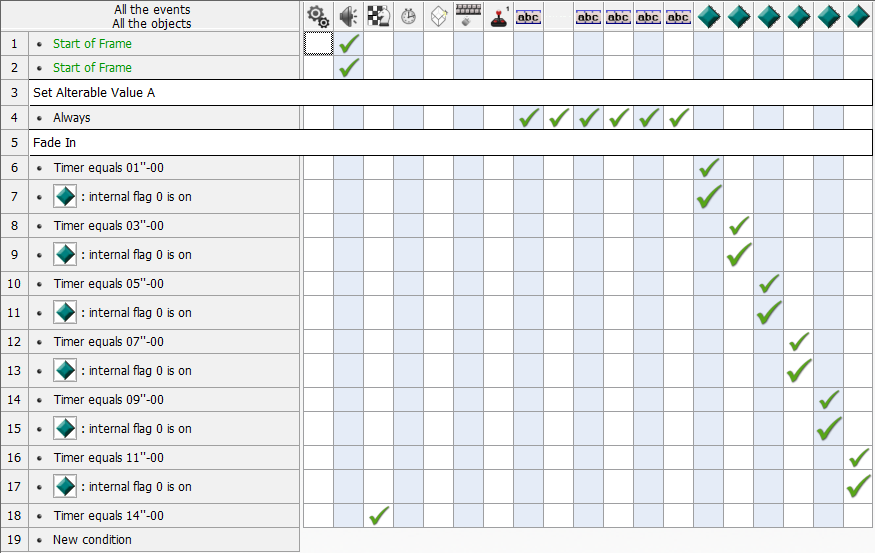


Figura : Sistema de eventos Click Team Fusion

|  |  |
| --- | --- |
| CLICK TEAM FUSION | |
| Fecha de lanzamiento | 2012 |
| Programado en | C, C++ |
| Lenguaje de programación | C, C++, Propio |
| Licencia | Software Propietario |

Tabla 20: Click Team Fusion Datos

#### Construct 2

La primera versión de Construct de código abierto fue creada por un grupo de estudiantes,cuya ultima build fue lanzada en 2012 dando por finalizado en ese momento el ciclo de vida del programa.

Construct 2 fue una evolución comercial del primer Construct ,creada por el mismo grupo de desarrolladores ya constituidos en la empresa Scirra, y sacada al mercado en Febrero de 2011.

Figura 10: Construct 2

Pensando como un motor de creación de videojuegos para usuarios sin conocimiento de programación, está basado basando en dos aspectos claves: el drag and drop de elementos y en la otorgación de patrones de comportamiento de manera visual.

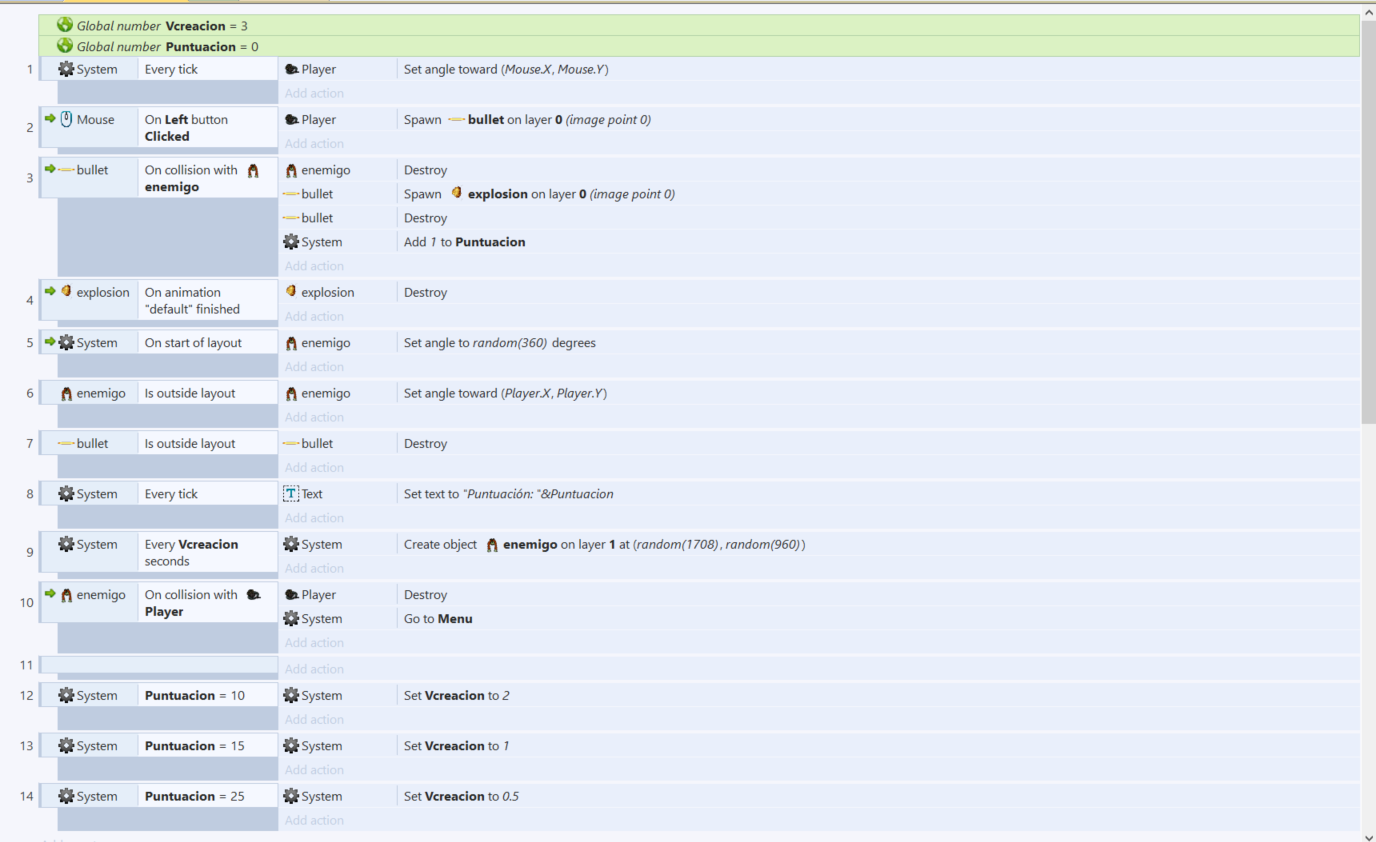
Su funcionamiento principal es a través de las hojas de comportamiento y/o evento, en las que a partir del cumplimiento de una serie de condiciones el programa ejecuta unas sentencias previamente definidas por el usuario. Además, a estas condiciones se les pueden añadir subcondiciones, o agregaciones para hacer todo lo complejo que uno quiera el comportamiento de los objetos del juego. En la Figura 11 se puede ver un ejemplo de una hoja de comportamiento de Construct 2.

Figura 11: Hoja de Eventos Construct 2

|  |  |
| --- | --- |
| CONSTRUCT 2 | |
| Fecha de lanzamiento | 2011 |
| Programado en | C++, Javascript |
| Lenguaje de programación | Javascript |
| Licencia | Software Propietario |

Tabla 21: Construct 2 Datos

#### Game Maker

Fundamentalmente divide su funcionamiento en dos grandes apartados, por un lado en el sistema de eventos y comportamientos, que se compone de hojas de eventos en la que cada hoja tiene una serie de eventos y sus sentencias condicionales y una vez que se cumpla se realizará el comportamiento asignado. Y por otro lado en evitar la selección de instancias mediante la selección de los objetos por medio del menú contextual que especifica el creador en las hojas de comportamiento.

Game Maker Studio, (a partir de ahora GM) es un software y kit de desarrollo para la creación de videojuegos creado en **Delphi.** Nació en el ámbito académico a finales de los noventa en Países Bajos. El profesor de la universidad de Utrecht, Mark Overmars, empezó a desarrollar una herramienta para ayudar a los estudiantes en la animación de personajes tanto de juegos como de otros ámbitos del dibujo, pero el proyecto fue creciendo y se convirtió en un software para que los estudiantes pudieran crear videojuegos en los primeros momentos de su carrera con pocos conocimientos de programación. En 1999 GM dejo de ser únicamente un programa para los alumnos de Overmars y fue lanzado al mercado.

Figura 12: Game Maker

La principal característica de Game Maker es que se intenta adaptar a todo tipo de usuarios, ya que permite realizar la implementación de las funcionalidades de dos formas completamente distintas: una desde su interfaz visual haciendo uso del Drag and Drop sobre objetos y otra mediante su lenguaje de programación GML.

En 2012 Yoyo Games (compañía propietaria de GM) lanzó en paralelo a la versión estándar de Game Maker otra versión llamada Game Maker: Studio, que incorporaba cambios profundos el programa tanto a nivel visual como a bajo nivel e incluía nuevas posibilidades de exportación a plataformas y sistemas operativos. En 2015 la versión Game Maker fue retirada por Yoyo Games y la rama Game Maker Studio se convirtió en la única versión del programa.

|  |  |
| --- | --- |
| GAME MAKER | |
| Fecha de lanzamiento | Noviembre 1999 |
| Programado en | Delphi |
| Lenguaje de programación | GML (Propio) |
| Licencia | Software Propietario |

Tabla 22: Game Maker Datos

1. conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orienta-da a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. [↑](#footnote-ref-1)
2. Aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software, como pueden ser editor de código fuente, herramientas de construcción automática o compilador. [↑](#footnote-ref-2)
3. Reddit es un sitio web de marcadores sociales y agregador de noticias donde los usuarios pueden dejar enlaces a contenidos web. Otros usuarios pueden votar a favor o en contra de los enlaces, haciendo que aparezcan más o menos destacados. Se trata de un mapa de discusión, como parte de un DDS global distribuido. [↑](#footnote-ref-3)
4. Taringa! es una red social de origen argentino, de uso masivo en países de habla hispana. [↑](#footnote-ref-4)
5. La Usabilidad es la medida de la calidad de la experiencia que tiene un usuario cuando interactúa con un producto o sistema [↑](#footnote-ref-5)
6. Esta licencia es una Licencia de software libre permisiva lo que significa que impone muy pocas limitaciones en la reutilización y por tanto posee una excelente Compatibilidad de licencia [↑](#footnote-ref-6)
7. son videojuegos de rol que permiten a miles de jugadores introducirse en un mundo virtual de forma simultánea a través de internet e interactuar entre ellos. [↑](#footnote-ref-7)
8. bibliotecas multiplataforma y libres, para el desarrollo de interfaces gráficas programadas en lenguaje C++ [↑](#footnote-ref-8)