

Universidad Tecnológica Centroamericana

Facultad de Ingeniería y Arquitectura

Proyecto de Graduación

Strategy Role Playing Game Engine

Previo a la obtención del Titulo

Ingeniero en Sistemas Computacionales

Presentado por:

20641045 Miguel Chicas

Asesor:

Ing. Iván de Jesús Déras

Campus San Pedro Sula

Junio, 2017

Autorización

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Hoja de Firmas

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Dedicatoria

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Resumen Ejecutivo

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Índice

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Glosario

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Introducción

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Planteamiento del Problema

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Antecedentes

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Definición del problema

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Preguntas de investigación

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Objetivos

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

### Objetivo General

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

### Objetivos Específicos

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Justificación

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Marco Teórico

## Video juego

### Juego

#### Definición

Para definir un video juego se debe empezar por definir que es un juego.

Según la Real Academia Española se define un juego como : “Ejercicio recreativo o de competición sometido a reglas, y en el cuales gana o se pierde” (“juego”, s/f).

La página del diccionario Merriam-Webster define juego como una competencia física o mental conducida en acorde ciertas reglas, en la cual los participantes se encuentran directamente en oposición unos a otros.(“Definition of GAME”, s/f).



Ilustración 1. Juego de Go.

Fuente: (Donarreiskoffer, 2004)

Un juego es un sistema formal, cerrado, que subjetivamente representa un subconjunto de la realidad. Por cerrado quiere decir que es autosuficiente, el modelo del mundo recreado por el juego es internamente completo. Las reglas de un juego propiamente diseñado cubren todas las posibilidades que se puedan encontrar en el juego. Estos son formales debido a que las reglas son explicitas y transparentes. Los juegos son sistemas en el sentido completo de la palabra, una colección de partes que interactúan de diversas maneras entre sí y son objetivamente irreales por qué no recrean las situaciones físicamente, sin embargo, las situaciones son reales para quien los juega (Crawford, Chris, 1984).

Se pueden comparar los juegos con otros tipos de entretenimiento de la siguiente manera. Un juego no es un acertijo, debido a que existen agentes que interactúan directamente sobre el jugador. Existen agentes sean estos otros jugadores u otros factores que conflictúan con el objetivo de otros jugadores. A diferencia de una historia, que es lineal en naturaleza, los juegos ofrecen ramificaciones de posibilidades. También, un juego es formal en cuanto a su reglamento, mientras que a un juguete se le puede dar el uso que se desee (Crawford, Chris, 1984).

### Video juego

La Real Academia Española define videojuego como : “Juego electrónico que se visualiza en una pantalla” (“videojuego”, s/f).

El diccionario de Oxford en línea define un video juego como un juego que se juega manipulando imágenes producidas por un programa de computadora en un monitor u otro tipo de visualización. (“video game - definition of video game in English | Oxford Dictionaries”, s/f)

El primer juego de computadora del que se tiene conocimiento fue creado para la tesis del en ese entonces candidato a Doctorado de la universidad de Cambridge Alexander S. Douglas. Douglas, fascinado por la Electronic Delay Storage Automatic Calculator (EDSAC) agrego a esta un juego de tres en línea al cual llamó Noughts and Crosses, complementando su tesis sobre la interacción computador-humano. […](Goldberg, 2011).

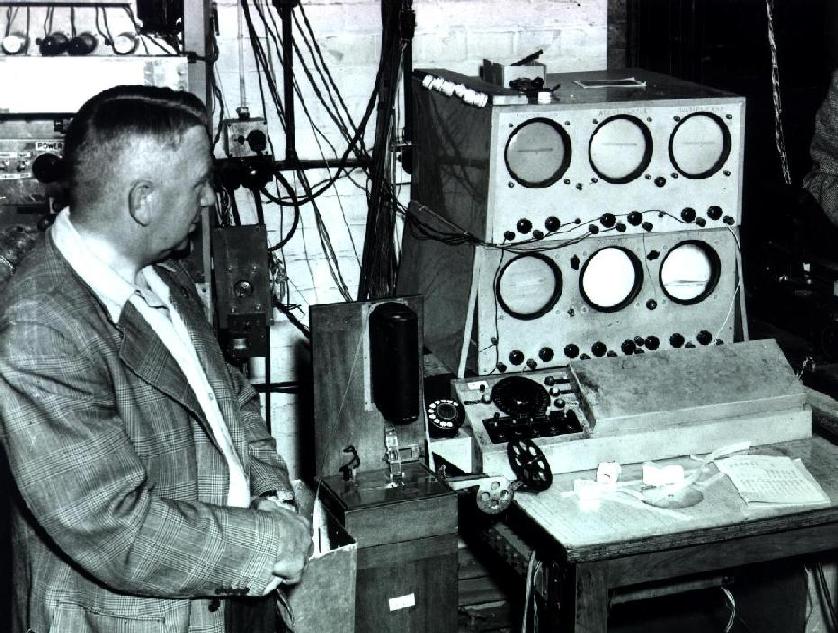


Ilustración 2. Panel de control de la EDSAC

Fuente: (Cambridge Computer Laboratory, s/f)

Otro intento temprano se le puede atribuir a William Higinbotham, quien trabajo en el Proyecto Manhattan, desarrollando los interruptores de tiempo que hicieron explotar en el momento correcto la bomba atómica. En 1958, luego de la guerra, se convirtió en el jefe de la división de instrumentación del Laboratorio Nacional de Brookhaven. Debido a que el laboratorio muestra ciertas exhibiciones al año, Higinbotham decide crear una exhibición interactiva, con la ayuda de Robert Dvorak, dan nuevo propósito a un osciloscopio y crean el juego tenis para dos. Este fue un éxito con los visitantes, apareciendo incluso el año siguiente, pero ni Dvorak ni Higinbotham, ni nadie más en ese momento pensó más de este incidente […]. (Donovan, 2010)

En febrero de 1962, “Spacewar!” fue creado por Steve Russell y sus compañeros del MIT. Este juego consistía de dos puntos verdes que flotaban en gravedad cero, disparándose en el espacio exterior. Esta experiencia inspiro a Nolan Bushnell, quien después de algunos momentos bajos en su vida decidió que la única manera de hacer dinero era ser emprendedor. En 1970 Bushnell empezó a construir su primera máquina de árcade. En junio de 1972 Bushnell fundó Atari inspirado en la idea de traer los juegos que había conocido hasta el momento a las masas (Donovan, 2010).

En 1966, Ralph Baer, un ingeniero de alto éxito, tuvo una epifanía mientras esperaba a un cliente. Baer, quien su trabajo hasta el momento involucraba radares y electrónicos anti aéreos y anti submarinos, imaginó una caja que pudiese conectarse a cualquiera de los millones de televisores estadounidenses y jugar todo tipo de juegos, la llamo inicialmente “Channel Let’s Play!” (¡Canal Juguemos!) (Donovan, 2010).

### Juego de Rol (RPG)

El término “juego de rol” (RPG) representa una variedad de juegos. Este término se ha expandido para incluir cualquier juego en el que el jugador controla un personaje en el mundo del juego y lo desarrolla a través del curso del juego. Los juegos de rol difieren de otras formas de entretenimiento debido a que sus jugadores crean sus propias experiencias a través de los personajes que representan (Tresca, 2011).

En un juego típico de Calabozos y Dragones, los jugadores parten en una aventura, donde el maestro del calabozo controla los eventos, ya sea por sí mismo o siguiendo una publicación previa. Una serie de aventuras interconectadas se le llama una campaña. El maestro del calabozo se encarga de determinar los sucesos dependiendo de las estadísticas de los personajes y de tablas de resultados. Los jugadores ruedan dados para determinar que sucede en dichos eventos. En la Ilustración 3. se puede observar un juego de calabozos y dragones llevándose a cabo.



Ilustración 3. Mapa de un juego de Calabozos y Dragones

Fuente: (Moroboshi, 2005)

Si bien es raro encontrar dos RPG que utilicen la misma fórmula, usando el concepto de familiaridad, se puede definir ciertas características que los juegos de rol poseen. La mayor parte de los RPG contiene algún tipo de combate, haciendo énfasis en la táctica a utilizar, y formas de dar soporte al combate. Tienen también lugares en donde se pueden adquirir servicios ya sea para mejorar los personajes o sus posesiones. Otras características comunes pueden ser acertijos y laberintos. Al final el único factor común que podes realmente encontrar a través de los RPGs es un sistema de estadísticas que gobierna lo que pueden hacer los personajes. […] Los RPGs necesitan el azar y un sistema de promoción formal que mejore las estadísticas de los personajes (Barton, 2008). En la Ilustración 4. se puede observar el manual del juego Final Fantasy, donde se describen varios aspectos del juego, incluyendo las estadísticas de los personajes, y vistas previas de los armamentos, artículos y la pantalla de batalla.



Ilustración 4. Página del manual del videojuego Final Fantasy para Nintendo Entertainment System

Fuente: (Nintendo, 1990)

### Juego de Estrategia

Un juego de estrategia es aquel en el que el rol del jugador es el de un general o un ser similar a un dios que controla la acción de ciertas unidades desde la distancia. El jugador se encuentra sobre la acción, la estrategia surge en contexto de guerra, y las tácticas desde un contexto de batalla (Barton, 2008). Un ejemplo por excelencia de este género es el Ajedrez. El ajedrez es un juego de dos jugadores donde cada jugador controla dos ejércitos de 16 piezas. Se juega en un tablero de 64 casillas. Los jugadores toman turnos haciendo movimientos, el juego termina cuando un jugador inmoviliza la pieza del rey del oponente (Emms, 2003). En la Ilustración 5. se puede apreciar las diferentes piezas de ajedrez existentes.



Ilustración 5. Piezas de Ajedrez, de izquierda a derecha: Rey, Torre, Reina, Peon, Caballero y Alfil

Fuente: (Light, s/f)

Los video juegos de estrategia nacen de sus primos cercanos los juegos de mesa. Comparados con otros géneros de juegos, sus reglas suelen ser simples y se prestan para el análisis de la efectividad de las mismas. Como regla general un juego de estrategia puro tiende a ser por turnos. El jugador tiende a considerar sus movimientos posibles tratando de aprovechar al máximo sus recursos. Uno de los mayores beneficios de computarizar los juegos de estrategia es la capacidad de un ordenador de poder manejar imparcialmente conjuntos de reglas complejas que aburrirían a un ser humano tratando de seguirlas (Rollings & Adams, 2003). Un ejemplo de video juego de estrategia es Wargroove que se puede apreciar en la Ilustración 6., en el cual de uno a cuatro jugadores controlan un ejército y su capitán (“Wargroove - FAQ”, s/f).



Ilustración 6. Un mapa de Wargroove.

Fuente: (“Wargroove - Announcing Wargroove!”, s/f)

### Juego de Rol Táctico (TRPG) / Juego de Rol de Estrategia (SRPG) / Juego de Rol de Simulación (SRPG)

En un juego de rol de simulación, las batallas son el énfasis del juego. Los personajes se desarrollan de manera muy similar a cualquier juego de rol. Las batallas se desarrollan en mapas que contienen diferentes tipos de obstáculos, como elevación, edificios, agua o cualquier cosa que pueda existir en la localidad. Se puede elegir que personajes tomaran parte en la batalla y que llevaran con ellos. Es importante decidir a quién llevar a la batalla y su posicionamiento. Para ganar una batalla, es necesario cumplir con la condición de gane, normalmente esto significa inhabilitar a todos los personajes enemigos o a uno en específico. La exploración y la interacción de los personajes suele ser secundaria (Hollinger & Ratkos, 1998).

Un ejemplo moderno de juego de rol de estrategia es Disgaea 2: Dark Hero Days descrito en la ilustración 7, el combate es el centro de enfoque de este juego. En el turno correspondiente, el jugador y el oponente toman turnos para mover unidades y ejecutar acciones con todas las unidades que posea, hasta un máximo de 10 por batalla. Ambos bandos luchan hasta que todos los personajes del bando contrario sean vencidos. El mapa de Disgaea 2: Dark Hero Days consiste de una cuadricula con diferentes obstáculos y elevaciones. Los movimientos de los personajes pueden ser bloqueados por otros personajes a menos que el personaje pueda volar o haciendo que un personaje arroje a otro. Los personajes pueden actuar con diferentes tipos de armamentos cada uno con rango y propiedades propias, además de varios tipos de magia y otras habilidades (Ross, Harbick, De Angelus, Meyers, & Warriner, s/f).



Ilustración 7. Disgaea 2: Dark Hero Days, se pueden apreciar los elementos de un juego de rol y de un juego de estrategia.

Fuente: (Nipon Ichi Software, 2006)

## Programación

Programar es tomar secuencias lógicas y ejecutarlas. La programación de la computadora permite ejecutar tareas de manera eficiente, rápida y exacta. La computadora no tiene la capacidad de analizar los problemas y buscar una solución, es deber del programador analizar el problema y desarrollar la serie de pasos lógicos para poder solucionarlo. Una vez la computadora tiene esta información es posible replicar la solución de manera rápida y concisa cuando sea necesario. A la serie de pasos lógicos que solucionan el problema se le llama algoritmos (Dale & Weems, 2007).

Una metáfora que describe de una manera comprensible el desarrollo de software es la construcción de software. Construir software implica varias etapas de planeación, preparación y ejecución que varían según lo que se construye. Por ejemplo, construir una casa para un perro no requiere mucho menos planeación y es mucho más permisiva con los errores, por otra parte, construir una casa es un proceso que lleva mucha planeación, y los errores son mucho más costos. Una complejidad mayor implica un mayor costo tanto en construcción como en desarrollo de software. Otra similitud es que no es deseable volver a hacer lo que ya este hecho, a menos que sea por una razón muy específica, por desempeño, por ejemplo. Ambos se benefician de una planeación apropiada y sufren cuando se necesita un cambio estructural drástico. Finalmente, mientras más grande el proyecto, mas es el impacto de los errores, y mayor la necesidad de ser cuidadosos (McConnell, 2004).

### Algoritmo

El termino algoritmo proviene de la traducción al latín del apellido de Mohammed al-Kowarizmi, quien utilizase reglas paso a paso para operar números decimales. Es así como definimos que un algoritmo es una serie de pasos necesarios para resolver un problema. Los algoritmos no dependen del lenguaje de programación y es un aspecto muy importante de la programación (Joyanes Aguilar, 2006).

Joyanes Aguilar, (2006) describe los pasos para la resolución, mostrado en la Ilustración 8, de problemas de la siguiente manera:

1. Diseño del algoritmo, el cual no debe ser ambiguo y debe conducir a la solución del problema
2. Expresar el algoritmo como un programa, es decir codificar el algoritmo en un lenguaje de programación.
3. Ejecución y validación, la computadora procede a ejecutar el algoritmo y se verifica la salida.

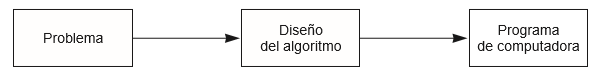


Ilustración 8. Resolución de problemas. (Joyanes Aguilar, 2006)

### Lenguaje de programación

En la computadora, los datos —cualquiera que sea su forma— se almacenan y emplean en códigos binarios, cadenas de “unos” y “ceros”. Las instrucciones y datos se almacenan en la memoria de la computadora por medio de estos códigos binarios. Si usted examinara los códigos binarios que representan instrucciones y datos en la memoria, no podría indicar la diferencia entre ellos; se distinguen sólo por la manera en que los usa la computadora. Esto hace posible que la computadora procese sus propias instrucciones como una forma de datos. (Dale & Weems, 2007, p.8)

Al principio solo era posible programar las computadoras en instrucciones integradas de la máquina, a esto se le llama lenguaje de máquina o código de máquina. EL código de máquina se introducía en código binario, esta era una tarea repetitiva y propensa a errores. Luego se introdujeron los lenguajes ensamblador para reducir los errores y facilitar la programación. Los lenguajes ensamblador coinciden uno a uno con las instrucciones de la máquina. Un paso más adelante se encuentran los lenguajes de alto nivel, los cuales se asemejan más a el lenguaje humano (Dale & Weems, 2007).

Un programa llamado compilador traduce los programas escritos en algunos lenguajes de alto nivel (C++, Pascal, FORTRAN, COBOL, Modula-2 y Ada, por ejemplo) en lenguaje de máquina. Si usted escribiera un programa en un lenguaje de alto nivel, puede ejecutarlo en cualquier computadora que tenga un compilador apropiado. Esto es posible porque la mayoría de los lenguajes de alto nivel están estandarizados, lo que significa que existe una descripción oficial del lenguaje. (Dale & Weems, 2007, p.8)

### C

C es un lenguaje de programación de alto nivel de propósito general, gracias a su eficacia y potencia es conocido como el lenguaje de programación de sistemas por excelencia. Es soportado por prácticamente todos los fabricantes de sistemas operativos, así como entidades educativas. C++, Java y C#, los tres lenguajes de programación más populares de la primera década del siglo XXI son hijos directos de C (Dale & Weems, 2007). La Ilustración 9 muestra un programa sencillo de C que muestra “Hello World” en la pantalla.

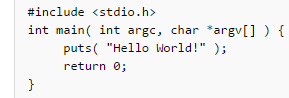


Ilustración 9. Programa “Hola Mundo” en C

Fuente: (“Hello world/Newbie - Rosetta Code”, s/f)

### C++

C++ es una extensión de C con características más potentes. Estrictamente hablando, es un superconjunto de C. Al igual que sucede con Java y C# que son superconjuntos de C++. El ANSI C estándar no sólo define el lenguaje C sino que también define una biblioteca de C estándar que las implementaciones de ANSI C deben soportar. C++ también utiliza esa biblioteca, además de su propia biblioteca estándar de clases. (Joyanes Aguilar, 2006, p.41)

Debido a esto casi todo programa escrito para C, es correcto en C++, pero no viceversa. Si bien esto es cierto, rara vez un programa de C++ se escribe similar a C. Esto se debe a que C++ provee características propias más potentes que C (Joyanes Aguilar, 2006).

Una función en C++ es una estructura del código que permite descomponer el programa en partes. Todo programa de C++ debe tener la función main, con esta se inicia el programa, main puede invocar otras funciones las cuales al terminar retornan el control a main (Dale & Weems, 2007).



Ilustración 10. Programa “Hola Mundo” de C++ utilizando la librería iostream

Fuente: (“Hello world/Newbie - Rosetta Code”, s/f)

Un programa de computadora opera sobre datos, en C++ estos datos deben ser de un tipo específico, algunos de estos tipos son tan comunes que ya se encuentran definidos previamente en el lenguaje, otros tipos se pueden definir por el usuario, estos son llamados tipos definidos por el usuario (Dale & Weems, 2007).

C++ fue diseñado para soportar la abstracción de datos, programación orientada a objetos y programación genérica, pero no está diseñado para forzar ningún tipo de estilo. Cada nombre y expresión tiene un tipo que determina que operaciones se pueden hacer con ese tipo de dato. C++ posee un conjunto de instrucciones para selección y para ciclos. C++ tiene soporte para apuntadores y para arreglos. C++ provee un mecanismo para poder agrupar datos y funciones relacionadas en nombres de espacio (namespaces) separados. También soporta la compilación separada, es decir que se pueden compilar partes del código por separado. C++ posee un conjunto de módulos de excepción, aunque se prefiere el manejo de errores a usar este. C++ soporta tipos concretos y abstractos y funciones virtuales y jerarquía de datos (Stroustrup, 1997). Los tipos fundamentales de C++ se pueden observar en la Tabla 1.

Tabla 1. Tipos fundamentales C++

|  |  |  |
| --- | --- | --- |
| Tipo | Ejemplos | Descripción |
| Booleano | bool | Un tipo booleano, este puede tener solamente dos valores, true (verdadero) y false (falso) |
| Carácter | char, wchar\_t, | Un tipo para describir un carácter. |
| Integrales | int, short, long | Un tipo para describir números enteros. |
| Punto Flotante | float, double, long doublé | Un tipo que representa un numero en formato de punto flotante. |
| Vacío | void | Un tipo que indica que una función no tiene valor de retorno. |

Fuente: (Stroustrup, 1997)

Los apuntadores son tipos que contienen la dirección de una variable, el objeto mas pequeño que se puede apartar memoria para es un char. Un apuntador se denota con un asterisco “\*”, el cual también se utiliza para dereferenciar apuntadores (Stroustrup, 1997). Se puede ver en la Ilustración 11 un ejemplo de apuntadores.

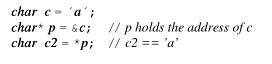


Ilustración 11. Apuntadores en C++

Fuente: (Stroustrup, 1997)

Para un tipo T, T[tamaño] representa un arreglo con una cantidad de elementos igual a tamaño, y se pueden acceder desde el índice 0 al índice tamaño – 1 (Stroustrup, 1997). Un ejemplo se puede ver en la Ilustración 12.

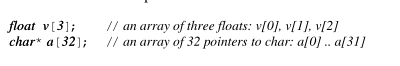


Ilustración 12. Arreglos en C++

Fuente: (Stroustrup, 1997)

#### C++11 y C++14

Hay muchas herramientas nuevas en C++11 y C++14. Las declaraciones “auto”, ciclos “for” basados en rango, expresiones lambda, referencias a los valores de derecha (rvalue), “nullptr” para indicar apuntadores nulos en lugar de 0, enumeraciones con alcance, mover en lugar de copiar. La característica más predominante de C++11 es probablemente las semánticas de movimiento. El fundamento de las semánticas de movimiento que distingue los valores de derecha (rvalue) de los valores de izquierda (lvalue). Los rvalue conceptualmente se refiere a objetos que pueden ser movidos mientras que en general los lvalue no pueden ser movidos (Meyers, 2015).

El concepto de “auto”, aunque simple, es más sutil de lo que se pueda pensar. Las variables “auto” obtienen su tipo del inicializador, esto quiere decir que siempre deben ser inicializadas. Debido a su naturaleza de deducción de tipos, esta puede representar tipos que solo el compilador conoce. En C++14 “auto” puede incluso ser utilizado como parámetro de expresiones lambda. Otra ventaja de “auto” es que debido a que utiliza el tipo preciso del objeto, se evitan errores cuando diferentes implementaciones de tipos que son parecidos no coinciden en sistemas diferentes (Meyers, 2015).

C++11 introduce inicialización uniforme, esta inicialización utiliza llaves para expresar la sintaxis de inicialización. Se puede utilizar para la inicialización de miembros de datos estáticos, para inicializar con un conjunto de datos un objeto y para inicializar objetos no copiables (Meyers, 2015).

En versiones anteriores de C++, se utilizaba “0” o el macro “NULL” para indicar un apuntador nulo. Los compiladores de C++ toman esto como un apuntado nulo solamente en última instancia, así que C++11 introduce el valor “nullptr”. El valor “nullptr” es de tipo “std::nullptr\_t”, el cual implícitamente se convierte a todo tipo de apuntador bruto, de esta manera se evita que se utilicen sobrecargas equivocadas a la hora de utilizar nulo como parámetro. Otra ventaja es que al momento de intentar deducir un tipo se evita la ambigüedad que proveería utilizar “0” o “NULL”(Meyers, 2015).

C++11 ofrece una alternativa a los “typedef”, se le llaman “aliases”. Para declarar un alias se utiliza la palabra reservada “using” seguido del nombre del alias, un símbolo de igual y el tipo al que se le desea dar el alias. Una ventaja del “alias” sobre el “typedef” es que el alias puede usarse con plantillas directamente sin necesidad de una solución alternativa (Meyers, 2015).

Como regla general, declarar un nombre dentro de llaves limita su visibilidad, este no es el caso con las enumeraciones normales de C++, debido a esto se creó la enumeración con alcance, comúnmente llamada clases de enumeración. Las enumeraciones con alcance contaminan menos el entorno de desarrollo y no se pueden convertir implícitamente a tipos integrales, lo cual hace que la intención del desarrollador se mantenga al solamente poder compararse con el mismo tipo. Un aspecto nuevo par ambos tipos de enumeraciones es la capacidad de declararlas previo a definirlas, en el caso de las enumeraciones sin alcance se debe de especificar que tipo interno se desea que sea (Meyers, 2015).

Las funciones borradas son otro aspecto de C++11, su propósito es evitar que se llamen funciones particulares. Se puede borrar una función simplemente igualándola a “delete”. En C++98 se colocaba la función como privada y se igualaba a 0, hacer esto generaba un error al querer utilizar la función al momento de hacer el enlazamiento, utilizando “delete” se puede detectar en tiempo de compilación. Otra ventaja de “delete” es que se puede utilizar para eliminar cualquier función, incluyendo sobrecargas especificas (Meyers, 2015).

C++11 agrega la palabra reservada “override” para poder indicar al compilador que la función que se está declarando es un reemplazo de la función de la base. Sin esto es posible crear funciones en una clase derivada muy parecidas a la original pero no exactamente iguales, en cuyo caso no actúa como se piensa que debería actuar la función (Meyers, 2015).

Los apuntadores en C++ son un concepto muy poderoso, y así como es de poderoso puede ser peligroso si no se tratan con cuidado. C++11 presenta la oportunidad de mantener el poder de los apuntadores en bruto con los apuntadores inteligentes. Los apuntadores inteligentes saben a quién pertenece el objeto apuntado y cuando y como destruirlo. Entre los apuntadores inteligentes existen los “auto\_ptr”, los “unique\_ptr”, los “shared\_ptr” y los “weak\_ptr” (Meyers, 2015).

Los “auto\_ptr” son una implementación previa a C++11 y se deben evitar, debido a que los otros apuntadores inteligentes hacen el trabajo de “auto\_ptr” mucho mejor. Cuando se necesita un apuntador inteligente, la primera opción es el “unique\_ptr”. El “unique\_ptr” tiene un tamaño por defecto igual al del apuntador en bruto para la mayor parte de las operaciones. El “unique\_ptr” tiene la peculiaridad de que no puede ser copiado, esto se debe a que al copiarlo el nuevo “unique\_ptr” apuntaría al mismo lugar que el original. Al momento de ser destruido un “unique\_ptr” no nulo destruye a lo que sea que este apuntando en ese momento. Se debe tener en cuenta que si bien por defecto un “unique\_ptr” es del mismo tamaño que un apuntador en bruto, el “unique\_ptr” crece cuando entran en juego destructores personalizados (Meyers, 2015).

La recolección de basura es una de las conveniencias de la programación moderna, pero suele ocurrir en momentos poco oportunos. C++11 introduce un concepto que permite que la colección de basura sea automática y a la vez predecible, los apuntadores compartidos o “shared\_ptr”. Como su nombre lo indica los “shared\_ptr” son apuntadores que no tienen posesión del objeto al que apuntan, pero entre todos los que apuntan al mismo objeto ayudan a asegurar su destrucción cuando ya no es necesario. A diferencia de los “unique\_ptr”, los “shared\_ptr” suelen tener un apuntador hacia el objeto y uno hacia el conteo de referencias de ese objeto, el cual es dinámicamente reservado. El costo de reservar memoria dinámicamente par aun “shared\_ptr” puede ser evitado utilizando “std::make\_shared”, pero no siempre se puede utilizar. Cabe destacar que es más eficiente mover un “shared\_ptr” que copiarlo(Meyers, 2015).

El cuarto tipo de apuntador inteligente es el “weak\_ptr”. Este tipo de apuntador inteligente puede quedar colgando, es decir que el apuntador apunte a un objeto invalido, esto se debe a que su uso es diferente al de los que se han mencionado con anterioridad. A diferencia de un “shared\_ptr”, el “weak\_ptr” no aumenta el conteo de referencias, debido a esto una vez que un objeto no contenga más referencias se elimina, dejando al “weak\_ptr” colgado, de esta manera el “weak\_ptr” no interfiere con el ciclo de vida del objeto, pero si se quiere utilizar el objeto se puede obtener un “shared\_ptr” a través de un “weak\_ptr”. Usos potenciales de este tipo de apuntador incluyen listas de observadores, prevención de ciclos de “shared\_ptr” y hacer caches (Meyers, 2015).

Las expresiones lambda son nuevas en C++11, estas proveen una forma muy conveniente de crear objetos de función. Una expresión lambda es esencialmente eso, una expresión. Una cerradura es el objeto creado por una lambda, y dependiendo del tipo de captura de la lambda, puede contener copias o referencias del ambiente en que se creó la lambda. En general se utilizan para crear argumentos para funciones que aceptan funciones como parámetros (Meyers, 2015).

## Herramientas de manutención de software

Software de construcción

Los programas de software se construyen a partir de sus archivos fuente, también conocido como código fuente. Una herramienta de construcción de software utiliza el código fuente y pasos predefinidos para ejecutar herramientas para crear el programa. A los archivos que se utilizan para tal tarea se les conoce como archivos de construcción (build files). Las herramientas de construcción deben poder construir el programa y conocer el orden en que se deben construir las partes y poder reconstruirlas independientemente de plataforma. Las herramientas de construcción generan ejecutables que pueden ser ejecutados en otras plataformas, a esto se le conoce como compilación cruzada (cross-compiling) (Doar, 2005).

Escribir código fuente es sencillo hasta que la cantidad de código empieza a crecer. Para mantener manejable el código se divide en partes, hacer esto crea dependencias y es en ese momento que pueden surgir problemas. Para reducir las dependencias se han creado diversos mecanismos, como ser archivos de encabezado, encapsulamiento e interfaces. El orden en que se construye el programa juega parte importante en la construcción satisfactoria de un programa, mantener este orden no es tarea fácil, pero las herramientas de construcción de software nos ayudan a hacer justamente esto, la tabla 1 ilustra que acciones se deben tomar al haber cambios. (Doar, 2005).

Tabla 2. Razones por las cuales un proyecto debe ser reconstruido

|  |  |
| --- | --- |
| Tipo de cambio | Acción que debe tomar la herramienta |
| Nuevos archivos agregados | El desarrollador debe agregar los archivos nuevos a los archivos de construcción y revisar si están correctos. |
| El contenido de los archivos cambió | Reconstruir el archivo; detectar si la compilación falo o fue un éxito. |
| Un archivo depende de un archivo que cambió. | Reconstruir los archivos afectados. |
| Un archivo ahora depende de un archivo que ya existe. | Reconstruir el árbol de dependencias. Verificar si el archivo puede ser encontrado correctamente. |
| Un archivo ahora depende de un nuevo archivo. | Reconstruir el árbol de dependencias. Verificar la existencia del archivo nuevo o si será creado como parte de la construcción. |
| Un archivo ya no depende de otro. | Reconstruir el árbol de dependencias. Verificar si el archivo antiguo causa algún problema o si debería ser eliminado. |
| Un archivo ahora depende de un archivo generado. | Asegurar que el árbol de dependencias cause la creación del archivo generado antes de que se necesite. |
| Un archivo ahora depende de un archivo generado cuyo archivo fuente ha sido cambiado. | Volver a generar los archivos generados necesarios. |

(Doar, 2005)

### Make

Make es la herramienta de construcción original, y la más popular con desarrolladores de C y C++. Las dependencias en make se especifican explícitamente en archivos llamados convencionalmente “Makefile” o “makefile” y son escritos en su propio lenguaje de make (Doar, 2005).

### CMake

Cmake es una familia de herramientas de fuente abierta y de plataforma cruzada diseñada para construir, probar y empaquetar software. Es utilizada para controlar el proceso de compilación del software utilizando una plataforma simple y archivos de configuración independientes del compilador para generar “makefiles” nativos y espacios de trabajo que pueden utilizar el compilador de la elección del usuario (“cmake.org”, consultado el 12/6/2017).

## Compilador

Compilar es el proceso en el que el código fuente de un programa se traduce a objetos, normalmente el objeto se encuentra en código de máquina. Un compilador es un programa que hace esto, el resultado de una compilación suele ser un programa ejecutable. (Joyanes Aguilar, 2006).

### GCC y G++

GCC por sus siglas en ingles quiere decir colección de compiladores de GNU (GNU Compiler Collection), es una distribución integrada de compiladores para varios lenguajes de programación importantes. Estos incluyen en este momento C, C++, Java, Fortran, Ada, Objective-C y Objective-C++. El compilador GNU para C++ es G++, el cual genera directamente código de máquina (“G++ and GCC - Using the GNU Compiler Collection (GCC)”, consultado el 12/6/2017).

## Entorno Integrado de Desarrollo

Un entorno de desarrollo integrado (IDE por sus siglas en inglés) está diseñado para facilitar el trabajo de desarrollo. Los IDEs contienen características como:

* Una consola para ingresar comandos.
* Editor de código fuente.
* Explorador de objetos.
* Interfaces para compilar y correr el software.
* Manejo de proyectos.
* Asistencia de depuramiento.
* Editores de objetos.
* Integración con la documentación (Verzani, 2011).

### QtCreator

QtCreator es un entorno de desarrollo integrado de plataforma cruzada para el desarrollo de aplicaciones de escritorio, empotradas o móviles. (“http://doc.qt.io/qtcreator/”, consultado el 12/6/2017). Se puede ver el espacio de trabajo de QtCreator en la Ilustración 9.

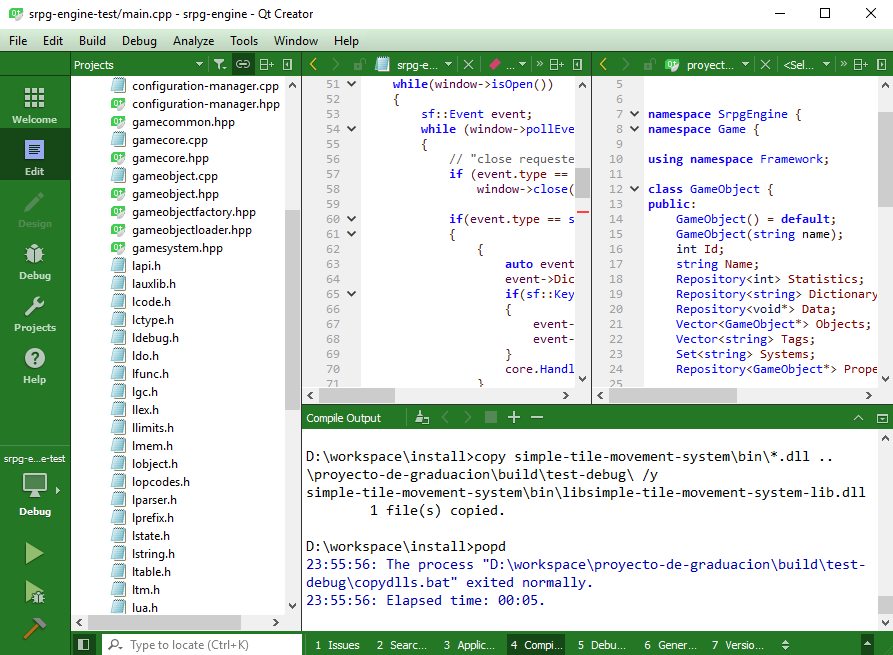


Ilustración 13. Ejemplo de un espacio de trabajo en QtCreator

Fuente: (Qt Corporation, 2017)

# Desarrollo del proyecto

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Variables de Investigación/Hipótesis

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Enfoque y Métodos

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Población y Muestra

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Técnicas e instrumentos aplicados

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Unidad de análisis y respuesta

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Materiales

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Fuentes de información

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Cronograma de actividades

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Resultados y Análisis

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Conclusiones

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Recomendaciones

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Aplicabilidad/Implementación

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Evolución del trabajo actual

Bibliografía

Barton, M. (2008). *Dungeons and Desktops: The History of Computer Role-Playing Games* (1a. ed.). A K Peters, Ltd.

Cambridge Computer Laboratory. (s/f). Computer Laboratory – The Relics Project: Archive photos. Recuperado el 13 de junio de 2017, a partir de http://www.cl.cam.ac.uk/relics/archive\_photos.html#Copyright\_Licencing

CMake. (s/f). Recuperado el 12 de junio de 2017, a partir de https://cmake.org/

Crawford, C. (1984). *The art of computer game design*. Osborne/McGraw-Hill.

Dale, N., & Weems, C. (2007). *Programación y resolución de problemas con C++* (Cuarta Edición). McGraw-Hill Interamericana.

Definition of GAME. (s/f). Recuperado el 10 de junio de 2017, a partir de https://www.merriam-webster.com/dictionary/game

Doar. (2005). *Practical Development Environments*. O’Reilly.

Donarreiskoffer. (2004). *English: Go board, at a Go-weekend, Hoge Rielen, Belgium*. Recuperado a partir de https://commons.wikimedia.org/wiki/File:Go\_board.jpg

Donovan, T. (2010). *Replay: The History of Video Games* (1a. ed.). Yellow Ant.

Emms, J. (2003). *Concise Chess, the compact guide for beginners.* (1a. ed.). Gloucester Publishers plc.

G++ and GCC - Using the GNU Compiler Collection (GCC). (s/f). Recuperado el 12 de junio de 2017, a partir de https://gcc.gnu.org/onlinedocs/gcc-4.0.2/gcc/G\_002b\_002b-and-GCC.html

Goldberg, H. (2011). *All your base are belong to us: how 50 years of videogames conquered pop culture* (1a. ed.). Three Rivers Press.

Hello world/Newbie - Rosetta Code. (s/f). Recuperado el 23 de junio de 2017, a partir de http://rosettacode.org/wiki/Hello\_world/Newbie#C

Hollinger, E., & Ratkos, J. (1998). *Final Fantasy Tactics: Prima’s official strategy guide*. Rocklin, California: Prima Publishing.

Joyanes Aguilar, L. (2006). *Programación en C++: algoritmos, estructuras de datos y objetos* (Segunda Edición). España: McGraw-Hill España.

juego. (s/f). Recuperado el 10 de junio de 2017, a partir de http://dle.rae.es/?id=MaS6XPk

Light, A. (s/f). *English: Chess pieces – left to right: king, rook, queen, pawn, knight and bishop.* Recuperado a partir de https://commons.wikimedia.org/wiki/File:ChessSet.jpg

McConnell, S. (2004). *Code Complete: A practical book of software construction* (2a. ed.). Redmond, Washington: Microsoft Press.

Meyers, S. (2015). *Effective Modern C++: 42 specific ways to improve your use of C++11 and C++14* (1a. ed.). 1005 Gravenstein Highway North, Sebastopol, CA 95472: O’Reilly Media Inc.

Moroboshi. (2005). *The picture show the map of a Dungeons and Dragons game, I (“Rocco Pier Luigi”, user:Moroboshi took this pictures the 13 July 2005*. Recuperado a partir de https://commons.wikimedia.org/wiki/File:Dungeons\_and\_Dragons\_game.jpg

Nintendo. (1990). *Final Fantasy Explorer’s Handbook*. Nintendo Of America Inc.

Nipon Ichi Software. (2006). *Disgaea 2: Dark Hero Days*.

Qt Corporation. (2017). *QtCreator Screen Capture*.

Qt Creator Manual. (s/f). Recuperado el 12 de junio de 2017, a partir de http://doc.qt.io/qtcreator/

Rollings, A., & Adams, E. (2003). *Andrew Rollings and Ernest Adams on Game Design*. New Riders Publishing.

Ross, I., Harbick, K., De Angelus, C., Meyers, N., & Warriner, C. (s/f). *Disgaea 2: Dark Hero Days. Official Strategy Guide*. Double Jump.

Stroustrup, B. (1997). *The C++ Programming Language* (3a. ed.). New Jersey: Addison-Wesley Professional.

Tresca, M. J. (2011). *Of Fantasy Role-Playing Games* (1a. ed.). McFarland & Company, Inc.

Verzani, J. (2011). *Getting Started with RStudio* (1a. ed.). United States of America: O’Reilly Media Inc.

video game - definition of video game in English | Oxford Dictionaries. (s/f). Recuperado el 10 de junio de 2017, a partir de https://en.oxforddictionaries.com/definition/video\_game

videojuego. (s/f). Recuperado el 10 de junio de 2017, a partir de http://dle.rae.es/?id=bmnbNU7

Wargroove - Announcing Wargroove! (s/f). Recuperado el 16 de junio de 2017, a partir de http://wargroove.com/announcing-wargroove/

Wargroove - FAQ. (s/f). Recuperado el 16 de junio de 2017, a partir de http://wargroove.com/faq/

Anexos