UNIVERSIDAD TÉCNICA ESTATAL DE QUEVEDO

FACULTAD DE CIENCIAS DE LA INGENIERÍA

CARRERA INGENIERÍA EN SISTEMAS

Proyecto de Investigación previo a la obtención del título de Ingeniero en Sistemas

Título del Proyecto de Investigación:

"DISEÑO E IMPLEMENTACIÓN DE UN SOFTWARE EDUCATIVO PARA MEJORAR EL APRENDIZAJE EN LA ASIGNATURA PROGRAMACIÓN MEDIANTE INTERACCIÓN GESTUAL"

Autor:

Mychael Otto Castro Espinoza

Director de Proyecto de Investigación:

Ing. Orlando Erazo Moreta, PhD

Quevedo – Los Ríos – Ecuador

2017

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

Yo, **Mychael Otto Castro Espinoza,** declaro que el trabajo aquí descrito es de mí autoría, que no ha sido previamente presentado para ningún grado o calificación profesional y que he consultado las referencias bibliográficas que se incluyen en este documento.

La Universidad Técnica Estatal de Quevedo puede hacer uso de los derechos correspondientes a este trabajo, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normatividad institucional vigente.

f. _____

Mychael Otto Castro Espinoza

C.I. 1206395863

CERTIFICACIÓN DE CULMINACIÓN DEL PROYECTO DE INVESTIGACIÓN

El suscrito, **Dr. Orlando Erazo Moreta,** docente de la Universidad Técnica Estatal de Quevedo, certifica que el estudiante **Sr. Mychael Otto Castro Espinoza,** realizó el Proyecto de Investigación de Grado Titulado **"DISEÑO E IMPLEMENTACIÓN DE UN SOFTWARE EDUCATIVO PARA MEJORAR EL APRENDIZAJE EN LA ASIGNATURA PROGRAMACIÓN MEDIANTE INTERACCIÓN GESTUAL"**, previo a la obtención del título de **INGENIERO EN SISTEMAS,** bajo mi dirección, habiendo cumplido con las disposiciones reglamentarias establecidas para el efecto.

……..

Ing. Orlando Erazo Moreta, PhD

DIRECTOR DE PROYECTO DE INVESTIGACIÓN

CERTIFICACIÓN DEL REPORTE DE LA HERRAMIENTA DE PREVENCIÓN DE COINCIDENCIA Y/O PLAGIO ACADÉMICO

Yo, Dr. Orlando Erazo Moreta, en calidad de director del Proyecto de Investigación **"DISEÑO E IMPLEMENTACIÓNDE UN SOFTWARE EDUCATIVO PARA MEJORAR EL APRENDIZAJE EN LA ASIGNATURA PROGRAMACIÓN MEDIANTE INTERACCIÓN GESTUAL"** me permito manifestar a usted y por su intermedio al Consejo Académico lo siguiente:

El estudiante Mychael Otto Castro Espinoza, egresado de la Facultad Ciencia de la Ingeniería, carrera Ingeniería en Sistemas, ha cumplido con las correcciones pertinentes, e ingresado su Proyecto de Investigación al sistema **URKUND**. Tengo a bien de certificar la siguiente información sobre el informe del sistema anti plagio con un porcentaje de 0%.

Atentamente,

_____

Ing. Orlando Erazo Moreta, PhD

DIRECTOR DEL PROYECTO DE INVESTIGACIÓN

CERTIFICACIÓN DE REDACCIÓN TÉCNICA DEL PROYECTO DE INVESTIGACIÓN

El suscrito, **Eliceo Ramírez Chávez, MSc** docente de la Universidad Técnica Estatal de Quevedo, certifica que el estudiante **Sr. Mychael Otto Castro Espinoza,** realizaron el Proyecto de Investigación de Grado Titulado **"DISEÑO E IMPLEMENTACIÓN DE UN SOFTWARE EDUCATIVO PARA MEJORAR EL APRENDIZAJE EN LA ASIGNATURA PROGRAMACIÓN MEDIANTE INTERACCIÓN GESTUAL"**, previo a la obtención del título de **INGENIERO EN SISTEMAS,** bajo mi dirección, habiendo cumplido con las disposiciones reglamentarias establecidas para el efecto.

……..

Ab. Eliceo Ramírez Chávez, MSc

RESPONSABLE DE REDACCIÓN TÉCNICA

UNIVERSIDAD TÉCNICA ESTATAL DE QUEVEDO

FACULTAD DE CIENCIAS DE LA INGENIERÍA

CARRERA DE INGENIERÍA EN SISTEMAS

PROYECTO DE INVESTIGACIÓN

Título:

"DISEÑO E IMPLEMENTACIÓN DE UN SOFTWARE EDUCATIVO PARA MEJORAR EL APRENDIZAJE EN LA ASIGNATURA PROGRAMACIÓN MEDIANTE INTERACCIÓN GESTUAL"

Presentado a la Comisión Académica como requisito previo a la obtención del título de Ingeniero en Sistemas.

Aprobado por:

_____

PRESIDENTE DEL TRIBUNAL

ING. BYRON OVIEDO VAYAS, PhD

_____	_____
MIEMBRO DEL TRIBUNAL	**MIEMBRO DEL TRIBUNAL**
ING. EMILIO ZHUMA MERA, MSc	ING. JANETH MORA SECAIDA, MSc

QUEVEDO – LOS RÍOS – ECUADOR

2017

AGRADECIMIENTO

Mi gratitud de todo corazón está dirigida a mis padres OTTO CASTRO y MARCIA ESPINOZA pilares fundamentales de mi familia, que han estado conmigo en las buenas y en las malas brindándome su apoyo, el cual ha sido esencial para poder alcanzar esta meta.

A mi abuelo JORGE CASTRO, que siempre lo llevaré en mi corazón y en mi mente, por sus pequeños pero grandes gestos de amor que me han servido de mucho.

A mi enamorada, a quien amo, por estar junto a mí en todo momento desde que inicié este viaje de aventuras y retos.

A todos mis amigos y profesores, de los cuales aprendí mucho en especial del Ing. Ariosto Vicuña y el Ing. Orlando Erazo. Sabias personas parte fundamental del éxito alcanzado.

Gracias totales,
Mychael Castro

DEDICATORIA

Este trabajo está dedicado a mi familia, mi abuelo (que en paz descanse), y a todas las personas que en algún momento formaron parte de este logro.

Mi padre dijo: un lapicero pesa menos que una pala. Gracias totales papa.

Gracias totales.

RESUMEN EJECUTIVO Y PALABRAS CLAVES

Las Tecnologías de la Información y la Comunicación (TICs) se han utilizado con fines educativos en varios campos, contribuyendo a mejorar las metodologías de enseñanza y aprendizaje. Uno de estos campos es la programación, en la que el aprendizaje de la lógica de la programación no es fácil porque las personas deben comprender y aplicar conceptos de resúmenes para resolver problemas. También se han empleado las TICs para este propósito, pero existen nuevas tecnologías que podrían brindar buenas oportunidades, como el uso de interfaces de usuario naturales (NUIs).

Este proyecto propone el uso de NUIs basadas en gestos manuales sin contacto para ayudar al aprendizaje de la programación. La propuesta incluye la gamificación de problemas para aprender a programar mientras se juega un juego. Por lo tanto, las personas deben realizar gestos para ejecutar sentencias de programación (por ejemplo, si, mientras, etc.) en escenarios específicos previamente preparados.

Para analizar la propuesta, se llevó a cabo un grupo de enfoque para extraer las características/requisitos que deberían incluirse en el prototipo del software. Luego, se utilizó tecnologías como UNITY 3D y MS Kinect para desarrollar el prototipo. Finalmente, se llevó a cabo un estudio de usuarios para evaluar la propuesta, en la cual las personas participantes usaron la aplicación y proporcionaron sus opiniones respondiendo un cuestionario. Los participaron estuvieron de acuerdo con la propuesta. Por lo tanto, el uso de aplicaciones basadas en gestos podría contribuir a la enseñanza y el aprendizaje de la programación.

Palabras Claves: Lógica de programación, Iteración Gestual, Interacción Humano-Computador, Kinect, Unity 3D

ABSTRACT AND KEYWORDS

Information and Communication Technologies (ICTs) have been used with education purposes in several fields, contributing to help improving the teaching-learning methodologies. One of this fields is programming, in which the learning of the logic of programming is not easy because people should understand and apply abstracts concepts to solve problems. ICTs have also been employed for this purpose, but there are new technologies that could provide good opportunities, such us the use of Natural User Interfaces (NUIs).

This project proposes the use of NUIs based on touchless hand gestures to help the learning of programming. Our proposal includes the gamification of problems in order to learn programming while playing a game. Thus, people should perform gestures to execute programming sentences (e.g., if, while, etc.) in specific scenarios previously prepared.

To analyze our proposal, we started carrying out a focus group to extract characteristics/requirements that should be included in the software prototype. Then, we utilized technologies such as UNITY 3D and MS Kinect to develop the prototype. Finally, a user study was conducted to evaluate the proposal, in which the people used the application and provided their opinions answering a questionnaire. The people who participated agreed with the proposal. Therefore, the use of applications based on gestures could contribute to the teaching and learning of programming.

Key Words: Programming Logic, Gestural Iteration, Human-Computer Interaction, Kinect, Unity 3D

TABLA DE CONTENIDO

CAPÍTULO I 2

CONTEXTUALIZACIÓN DE LA INVESTIGACIÓN 2

1.1. Problema de investigación 3

1.1.1. ..... Planteamiento del problema 3

1.1.3. ..... Formulación del problema 5

1.1.4. ..... Sistematización del problema 5

1.2. Objetivos 6

1.2.1. ..... Objetivo General 6

1.2.2. ..... Objetivos Específicos 6

1.3. Justificación 7

CAPÍTULO II 8

FUNDAMENTACIÓN TEÓRICA DE LA INVESTIGACIÓN 8

2.1. Marco Contextual 9

2.2. Marco Conceptual 9

2.2.1. ..... Programación 9

2.2.2. ..... Algoritmo 9

2.2.3. ..... Fases en la resolución de problemas 10

2.2.4. ..... Interacción Persona-Ordenador 11

2.2.4.1. ..... Interacción Basada en Movimiento 12

2.2.4.2. ..... Interfaces naturales de usuario 12

2.2.4.3. ..... Kinect 12

2.2.4.4. ..... Funcionamiento 13

2.2.4.5. ..... Reconocimiento de voz 14

2.2.4.6. ..... Reconocimiento de esqueleto 14

2.2.5. ..... Juegos serios en el aprendizaje 15

2.2.6. ..... Gamificación 15

2.2.7. ..... Unity 3D 15

2.3. Marco Referencial 17

2.3.1. ..... Diseño de interfaces basada en gestos de manuales para facilitar la participación de los estudiantes desde su asiento 17

2.3.2. ..... Interfaces de usuario basadas en gestos manuales sin contacto para la sala de clases 18

2.3.3. ..... Gamificación del Proceso de Aprendizaje 19

CAPÍTULO III 21
METODOLOGÍA DE LA INVESTIGACIÓN 21
3.1. Localización 22
3.2. Tipos de investigación 22
3.2.1. Investigación Descriptiva 22
3.2.2. Investigación Exploratoria 22
3.3. Métodos y técnicas a usar en la investigación 22
3.4. Recursos y materiales 23
CAPÍTULO IV 25
METODOLOGÍA DE DESARROLLO 25
4.1. Desarrollo del software 26
4.1.1. Metodología en cascada 26
4.1.2. Análisis de requerimientos 26
4.1.3. Diseño del sistema 27
4.1.4. Fase de codificación 33
4.1.5. Pruebas 34
CAPÍTULO V 35
RESULTADOS 35
5.1.1 Planteamiento de las características que debe tener la interfaz de usuario para facilitar el aprendizaje de la lógica de programación 36
5.1.2 Identificación de las tecnologías adecuadas para el desarrollo del prototipo del software .. 41
5.1.3 Resultados del desarrollo del software 43
5.1.4 Estudio de usuarios para la evaluación de la utilidad y aceptabilidad de la propuesta 47
CAPÍTULO VI 50
CONCLUSIONES Y RECOMENDACIONES 50
6.1 Conclusiones 51
6.2 Recomendaciones 52
CAPÍTULO VII 53

BIBLIOGRAFÍA 53

CAPÍTULO VIII 55

ANEXOS 55

7.1 Diccionario de datos 56

7.2 Modelo físico de la base de datos 58

7.3 Gestos usados dentro del software 58

7.4 Protocolo para la realización del grupo de enfoque de requerimientos del sistema 66

7.4.1 Resultados del cuestionario demográfico del grupo de enfoque para la adquisición de requerimientos del software 69

ÍNDICE DE FIGURAS

Figura 1. Fundamentos de programación 11
Figura 2. Sensor Kinect 13
Figura 3. Funcionamiento detector 3D Kinect 13
Figura 4. Reconocimiento del esqueleto mediante el dispositivo Kinect 14
Figura 5 Porcentaje de juegos realizados con Unity 16
Figura 6 Espacio de gestos del usuario 17
Figura 7 Gestos propuestos 18
Figura 8 Formas de representación de los usuarios 18
Figura 9 Feedback visual al seleccionar un botón mediante gestos “Hover” (izquierda) y “Push” (derecha) 19
Figura 10 Ejemplo del mapa de PLMan 20
Figura 11 Ejemplo de código IA, escrito en Prolog, para superar un mapa de PLMan 20
Figura 12 Patrón de arquitectura de Software 27
Figura 13 Casos de uso del sistema 28
Figura 14 Arquitectura final del prototipo del software 33
Figura 15 Pantalla de inicio para seleccionar un jugador 43
Figura 16 Escena para mostrar información mediante un personaje instructor 43
Figura 17 Escenas de información de variables, sentencias y operadores 44
Figura 18 Apartado para muestra de instrucciones sobre las tareas 44
Figura 19 Gestos disponibles dentro de una escena 44
Figura 20 Algoritmo construido por el usuario 45
Figura 21 Vista aérea de la mini-ciudad 45
Figura 22 Recorrido del autobús dentro de la escena 46
Figura 23 Tareas dentro de la escena 46
Figura 24 Personajes mostrando interacción (esperando) dentro de la escena 46
Figura 25 Diagrama de Base de datos 58
Figura 26 Ejemplos de participantes del estudio de usuarios (programadores nivel avanzado grupo KFC Quito-Ecuador) 76

ÍNDICE DE TABLAS

Tabla 1. Requerimientos del hardware 23
Tabla 2. Requerimientos del software 23
Tabla 3. Presupuesto del proyecto de investigación 24
Tabla 4. Caso de uso: Iniciar partida 29
Tabla 5. Caso de uso: Seleccionar partida 30
Tabla 6. Caso de uso: Mostrar información (sentencia) 31
Tabla 7. Caso de uso: Resolver problema 32
Tabla 8. Resultados del Grupo de enfoque 36
Tabla 9. Comparación entre motores para el desarrollo de videojuegos 41
Tabla 10. Comparación de sensores para el reconocimiento de gestos 42
Tabla 11. Resultados preguntas cerradas 47
Tabla 12. Diccionario de datos- tabla Usuario 56
Tabla 13. Diccionario de datos- tabla Partida 56
Tabla 14. Diccionario de datos- tabla Escena 56
Tabla 15. Diccionario de datos- tabla DetallePartida 57
Tabla 16. Diccionario de datos- tabla Modulo 57
Tabla 17. Diccionario de datos- tabla Estado 57
Tabla 18. Tabla de gestos 58
Tabla 19. Storyboard de presentación inicial 60
Tabla 20. Storyboard para la escena de operadores básicos 61
Tabla 21. Storyboard para las escenas de la estructura de control While 62
Tabla 22. Storyboard para la escena que muestra tareas de la estructura de control For 63
Tabla 23. Storyboard para la escena que muestra tareas de la estructura de control DO WHILE 64
Tabla 24. Preguntas objetivas para la obtención de requerimientos de los participantes del grupo de enfoque 67
Tabla 25. Encuesta demográfica para los participantes del grupo de enfoque 68
Tabla 26. Encuesta demográfica realizada a los participantes del estudio de usuarios 74

CÓDIGO DUBLIN

Título:	Diseño e implementación de un software educativo para mejorar el aprendizaje en la asignatura programación mediante interacción gestual.			
Autor:	Castro Espinoza Mychael Otto			
Palabras clave:	Lógica de programación	Iteración Gestual	Interacción Humano-Computador	Kinect
Fecha de publicación:				
Editorial:				
Resumen:	**Resumen.** Las Tecnologías de la Información y la Comunicación (TICs) se han utilizado con fines educativos en varios campos. Uno de estos campos es la programación, en la que el aprendizaje de la lógica de la programación no es fácil porque las personas deben comprender y aplicar conceptos de resúmenes para resolver problemas. Este proyecto propone el uso de interfaces de usuario natural (NUIs) basadas en gestos manuales sin contacto para ayudar al aprendizaje de la programación. Para analizar la propuesta, se llevó a cabo un grupo de enfoque para extraer las características/requisitos. Finalmente, se llevó a cabo un estudio de usuarios para evaluar la propuesta. **Abstract.** Information and Communication Technologies (ICTs) have been used with education purposes in several fields. One of this fields is programming, in which the learning of the logic of programming is not easy because people should understand and apply abstracts concepts to solve problems. This project proposes the use of NUIs based on touchless hand gestures to help the learning of programming. To analyze our proposal, we started carrying out a focus group to extract characteristics/requirements. Finally, a user study was conducted to evaluate the proposal.			
Descripción:				

INTRODUCCIÓN

La programación es fundamental dentro del pensum académico de las universidades que la imparten en carreras como Ciencias de la Computación, Tecnologías de Información y Comunicación (TICs) y afines. Los avances en las técnicas usadas dentro de esta disciplina han ido evolucionando a través del tiempo con el objetivo de abarcar nuevos requerimientos del mercado. Esto es importante para que las personas que estudian ésta disciplina estén en un constante aprendizaje.

El aprendizaje de la programación y sus fundamentos implican tener actitudes como orden, concentración y constancia, además de tener pasión por resolver problemas mediante una computadora. Dentro del proceso de aprendizaje se presentan ciertas dificultades para algunas personas debido a factores tales como: la falta de concentración, motivación, apatía frente al tema o incluso no contar con las herramientas necesarias disponibles. Aspectos como estos influyen en el descuido de conceptos y aspectos importantes, los cuales conllevan al no entendimiento de temas posteriores ocasionado que la enseñanza y/o aprendizaje aumente en su grado de dificultad.

Actualmente existen algunas iniciativas para el aprendizaje de programación. Un ejemplo de esto es la conocida “The Hour Of Code” (“la hora del código”) que está orientada a los estudiantes de escuelas en Estados Unidos y en América latina dentro de la educación básica y media [1]. Además, existen herramientas informáticas gratuitas para el aprendizaje como Scratch [2], que es un lenguaje de programación gráfico creado para ayudar especialmente a los niños a desarrollar el pensamiento computacional al permitirle imaginar, reflexionar y compartir mediante el juego desde su computador y/o dispositivo móvil, dichas iniciativas tecnológicas han aportado a la mejora de lógica de programación y pensamiento computacional en las personas.

Tomando en cuenta las iniciativas de usar la tecnología dentro del ámbito académico y el juego como factores fundamentales para la enseñanza-aprendizaje, indistintamente a los métodos tradicionales, se propone el uso de software educativo que ayude al aprendizaje del pensamiento lógico de programación empleando un estilo de interacción diferente al tradicional. Los usuarios podrán interactuar con el software mediante una interfaz de usuario basada en gestos manuales y sin contacto físico con dispositivo alguno. Para ello, la utilidad de la propuesta es comprobada mediante un estudio de usuarios en el que las personas participantes llevan a cabo ciertas tareas de programación mediante gestos y dan su opinión de la utilidad/aceptación de la herramienta propuesta.

CAPÍTULO I

CONTEXTUALIZACIÓN DE LA INVESTIGACIÓN

1.1. Problema de investigación

1.1.1. Planteamiento del problema

En la actualidad los métodos para la enseñanza-aprendizaje de la lógica de programación han evolucionado con un cambio en el modelo educativo, que van desde un modelo centrado en la "enseñanza del profesor" al "aprendizaje del alumno" [3]. Aspectos como estos conllevan a la formación de los profesores en nuevas metodologías. Sin embargo, poco se ha invertido en la formación de nuevos contenidos acorde a las nuevas metodologías. Por lo tanto, estos modelos de enseñanza-aprendizaje pueden ser poco interactivos dentro de las aulas de clase, teniendo como consecuencia que los estudiantes no estén involucrados activamente en el proceso de aprendizaje y/o enseñanza.

Hay que tener en cuenta que difícilmente se puede aprender a programar en su "totalidad" dentro de poco tiempo. El "arte" de programar requiere de orden, esfuerzo, dedicación, constancia, entre otras capacidades, que no se consiguen desarrollar en plazos cortos. Además, en ocasiones las personas se centran en el entorno de desarrollo (IDE) y/o en lenguaje de programación para aprender por "comodidad" y/o facilidad, olvidándose de aspectos importantes como entender correctamente el problema, diseñar un proceso para solucionarlo (algoritmo), escribir el código (programa) en el lenguaje correcto y finalizar con pruebas y depuración, y finalmente la documentación [4].

Tomando en consideración los aspectos a seguir y cumplir dentro de este proceso de enseñanza-aprendizaje, desarrollar la lógica de programación parece ser algo difícil para las personas, las cuales tienen que entender en poco tiempo una variedad de conceptos y técnicas que en algunos casos tienden a ser complejos.

1.1.2. Diagnóstico

Uno de los factores influyentes en el poco interés de las personas inmersas en el proceso de enseñanza-aprendizaje de programación suele ser el modelo de estudio, que en ciertos casos resulta ser poco interactivo, debido al bajo uso de la tecnología disponible que ayuda dentro de este proceso como imágenes, videos, juegos, etc. Este limitado aprovechamiento de la tecnología dentro de los modelos de estudio causa que las personas no estén con el grado de concentración adecuada para comprender la complejidad que trae consigo la programación y su lógica, trayendo como consecuencias desertar en el tema, desgaste de tiempo y en el caso de estudiantes la pérdida de un ciclo de estudio. Por consiguiente, es importante la comprensión de los conceptos esenciales inmersos en ésta materia.

En el caso que las personas o estudiantes no tengan claro los conceptos fundamentales esenciales en la enseñanza-aprendizaje de la programación, su comprensión resulta ser una tarea complicada, sea por confusión de conceptos y/o práctica o carencia de los mismos. Hacer uso de una de las herramientas más importantes de todos los tiempos como lo es el Internet puede resultar útil, sin embargo, se usa a menudo para buscar soluciones resueltas por otras personas obteniendo como consecuencias profesionales buscadores de códigos y no capaces de escribirlos por la falta de lógica en programación.

En el caso de las metodologías de enseñanza-aprendizaje modernas, la mayoría logra su objetivo principal, que es el de captar la atención de las personas, usando recursos y técnicas como la Gamificación (aplicar mecánicas de juegos en contextos no “jugables”) o aprender jugando, los cuales son llamativos, pero por corto tiempo.

Hay que tomar en consideración que con el aumento de la tecnología las expectativas para que los nuevos productos (juegos) logren mantener la atención de las personas que son cada vez más altas.

1.1.2.1. Pronóstico

La elección inadecuada de un modelo poco interactivo en la enseñanza-aprendizaje para mejorar la lógica de programación, generará que las personas no tengan claro los conceptos básicos esenciales de la materia, aumentando la dificultad para la resolución de problemas mediante el computador.

1.1.3. Formulación del problema

¿Cómo apoyar el aprendizaje de la lógica de programación usando el paradigma de la interacción sin contacto?

1.1.4. Sistematización del problema

¿Qué características debería tener la interfaz de usuario del software para que aporte al aprendizaje de la lógica de programación en las personas?

¿Qué tecnologías de interacción sin contacto basada en gestos con las manos son adecuadas de utilizar en el desarrollo del prototipo del software propuesto?

¿De qué modo se puede determinar la utilidad y aceptabilidad de la propuesta?

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar un software para apoyar el aprendizaje de la lógica de programación de las personas usando el paradigma de la interacción sin contacto.

1.2.2. Objetivos Específicos

- Determinar las características que debería tener la interfaz de usuario del software para que aporte al aprendizaje de la lógica de programación.

- Identificar la tecnología de interacción sin contacto basada en gestos con las manos adecuada para el desarrollo del prototipo del software.

- Ejecutar un estudio de usuarios para evaluar la utilidad y aceptabilidad de la propuesta.

1.3. Justificación

Hoy en día, con la facilidad de acceso a internet y su alta cantidad de información disponible, las personas encuentran más fácil, entretenido y divertido aprender algo utilizando videos, sonidos, imágenes, e incluso utilizando un juego, en lugar de escuchar la materia en un aula de clases de la manera tradicional [5]. A medida que estas formas de aprendizaje han aumentado, se ha concebido el uso de las Tecnologías de Información y Comunicación (TICs) en la educación. Ellas se han ido adecuando a los procesos de enseñanza-aprendizaje y han dado paso a nuevas metodologías de enseñanza.

Las nuevas metodologías ayudan a que el estudiante este motivado durante el proceso de aprendizaje [6]. Uno de los métodos actuales consiste en la aplicación de técnicas de Gamificación [7] o ludificación durante el proceso de enseñanza. Este método consiste en utilizar mecanismos, técnicas y paradigmas aplicados en los juegos, a contextos no lúdicos. Como resultado se obtienen los denominados juegos serios ("Serious games" [7]) que son aplicados dentro del proceso de aprendizaje para aumentar la motivación del estudiante, trasmitir un mensaje o cambiar un comportamiento mediante una experiencia lúdica.

El uso de los juegos, combinado con Interfaces de Usuario Natural (NUIs), ofrecen una mejora en la interacción del estudiante durante el aprendizaje, aumentando la participación activa dentro y fuera de las aulas de clases [8]. Debido a la complejidad que tienen ciertos temas que contiene la programación y los beneficios que ofrecen los juegos para el aprendizaje, se aprovechará ésta oportunidad para desarrollar un software de tipo juego serio que ayude a los estudiantes a aprender, practicar y desarrollar la lógica de programación. Este software, haciendo uso de gestos manuales, permite al estudiante interactuar con él como si estos gestos fueran una extensión del juego.

CAPÍTULO II

FUNDAMENTACIÓN TEÓRICA DE LA INVESTIGACIÓN

2.1. Marco Contextual

En la actualidad, las instituciones educativas del Ecuador y de otros países en el mundo deben tener en consideración que con el aumento de la tecnología el modelo de enseñanza-aprendizaje para la lógica de programación y sus expectativas para mantener la atención de las personas son cada vez más altas. Por ello, este proyecto busca analizar el uso de software basado en interacción sin contacto, que complementándose con el uso de técnicas de gamificación, coadyuve al proceso de enseñanza-aprendizaje de la lógica de programación.

2.2. Marco Conceptual

2.2.1. Programación

La programación de computadoras es el arte de diseñar, codificar depurar, mantener el código fuente de programas de computadora. Estos códigos son escritos en un lenguaje de programación que luego son traducidos a un lenguaje máquina que podrá ser interpretado por el computador.

Para la mayoría de las necesidades existen leguajes de programación, “desde aplicaciones para manejar directamente un circuito electrónico, sistemas de nómina, hasta software para registrarse y realizar compras por Internet” [9].

La programación se encuentra inclusive en dispositivos móviles, empleando la programación al jugar, o buscar un contacto en una lista telefónica [9].

2.2.2. Algoritmo

Son un conjunto finito de ideas sistemáticamente ordenadas de forma lógica para dar solución a un problema determinado. Para el diseño de estos se necesita de creatividad y conocimientos del arte de programar o programación [9].

Un algoritmo no depende de un lenguaje de programación, son independientes del mismo y son la base de cualquier solución escrita en algún lenguaje de programación, es tanta su importancia, que un algoritmo se considera más importante que un lenguaje de programación o un computador ya que estos son tan solo un medio para ser expresado y la segunda un procesador para ejecutarlo [10].

2.2.3. Fases en la resolución de problemas

Durante el transcurso de resolución de problemas encontramos la escritura y ejecución de un programa, de manera tal que dentro de este proceso los programadores pueden considerar una serie de pasos a seguir para el diseño de las soluciones a un problema mediante un programa [10].

Las fases de resolución de un problema con computadora son:

• Análisis del problema.
• Diseño del algoritmo.
• Codificación.
• Compilación y ejecución.
• Verificación.
• Depuración.
• Mantenimiento.
• Documentación.

Las características más sobresalientes de la resolución de problemas son:

• **Análisis.** Es la etapa inicial para la resolución de problemas mediante la computadora, dentro de esta fase de debe analizar el problema teniendo claramente definidos los requisitos del cliente.

• **Diseño.** Posterior al análisis, se procede a la elaboración paso a paso de un algoritmo que se encargue resolver el problema.

• **Codificación (implementación).** El código fuente se escribe en un lenguaje de programación de alto nivel, posterior a ello ser compilado y pueda ser interpretado por el computador.

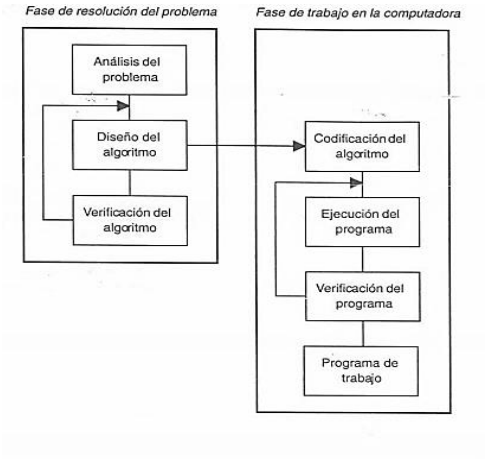
• **Ejecución, verificación y depuración.** Al realizar la ejecución del programa se toman en cuenta una variedad de datos de entrada comúnmente denominados datos de prueba. La depuración consiste en verificar si existen errores (“bugs”) dentro del programa y corregir/borrar dichos errores.

• **Mantenimiento.** Consiste en la actualización y modificación del programa en base a las necesidades del cliente/oportunidades de mejora, cada vez que sea necesario.

• **Documentación.** Una buena práctica para facilitar el mantenimiento de un programa es la documentación del mismo. Dentro de esta fase se incluye la información de las fases anteriores en especial análisis, diseño y la codificación.

Dentro de las dos primeras fases conducen al diseño detallado del algoritmo en base al análisis del problema. Dentro de la tercera fase se escribe (codificación) el algoritmo en algún lenguaje de programación plasmando las ideas y análisis obtenido en las fases anteriores. Las fases de compilación y ejecución cumplen la tarea de traducir y ejecutar el código escrito a un lenguaje de bajo nivel el cual entiende el computador. Finalmente en las fases de verificación y depuración se buscan y corrigen los errores de las etapas anteriores y se los corrige o eliminan [10]. La figura 1 presenta el flujo respectivo a la resolución de problemas y sus respectivas fases.

Figura 1. Fundamentos de programación



Fuente: Fundamentos de programación
Elaborado por: Luis Joyanes, Luis Rodríguez, Matilde Fernández

2.2.4. Interacción Persona-Ordenador

Es un área de investigación enfocada al estudio de las modalidades de interacción entre personas y computadoras, para el análisis y diseño de interfaces que proporcionen mejor usabilidad e interactividad entre ambos (persona y computador) con el objetivo que la comunicación entre si sea eficiente, minimizando errores, aumentando la satisfacción de las personas, y facilitar el uso relativo al dispositivo.

La interacción Persona-Ordenador, popular por sus siglas en ingles HCI (Human-Computer Interaction) estudia la comunicación entre el humano y las máquinas y en qué medidas éstas están desarrolladas para una interacción exitosa con los seres humanos.

Por máquina se considera cualquier tecnología, desde el típico ordenador de escritorio hasta un sistema de ordenadores de gran escala, sistema embebido [11].

2.2.4.1. Interacción Basada en Movimiento

Estos dispositivos brindan un campo de visión el cual permite la detección de las posiciones y el movimiento gracias a la continua interacción con los usuarios dentro del mencionado campo de visión. Esta interacción mediante los dispositivos usados en ella ofrece la posibilidad de analizar cada situación para actuar en consecuencia, donde su característica más importante es que permite controlar sistemas mediante movimientos "naturales" como pueden ser los gestos con las manos o posturas del cuerpo [11].

2.2.4.2. Interfaces naturales de usuario

La forma de interacción entre humanos y computadoras, ha tenido cambios importantes desde los comienzos de la computación; que van desde la utilización de tarjetas perforadas hasta la actualidad en la que se busca la interacción sin dispositivos aparentes para realizar la comunicación. "A este tipo de interacción, se le ha dado el nombre de: Interfaces Naturales de Usuario (NUIs por sus siglas en inglés)" [8].

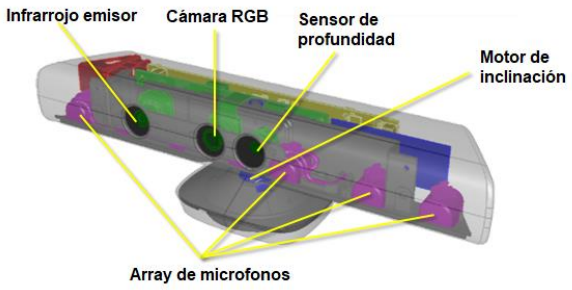
Un primer acercamiento a este tipo de interfaces sería: aquellas que permiten a los usuarios interactuar con un sistema, aplicación, etc. Sin utilizar sistemas de mando o dispositivos de entrada comunes en las interfaces graficas de usuario (GUI) tales como ratón, teclado alfanumérico, lápiz óptico, joystick, etc. Para construir una experiencia de uso, que le sea natural a nuestro usuario [8].

2.2.4.3. Kinect

Inicialmente Kinect fue considerado como un controlador de juego para permitir a los usuarios controlar e interactuar con la Xbox 360 sin contacto físico con un control de videojuegos tradicional (figura 2), mediante una interfaz natural de usuario que es capaz de reconocer gestos, lanzada a la venta el 4 de noviembre del 2010 resultado de 20 años de desarrollo por parte de Microsoft Research [11].

Los componentes que integran Kinect entre los cuales están su sensor de profundidad, cámara RGB, array de micrófonos y sensor de infrarrojos, es capaz de detectar y capturar el esqueleto humano, reconocerlo y posicionarlo en el plano [11]

Figura 2. Sensor Kinect

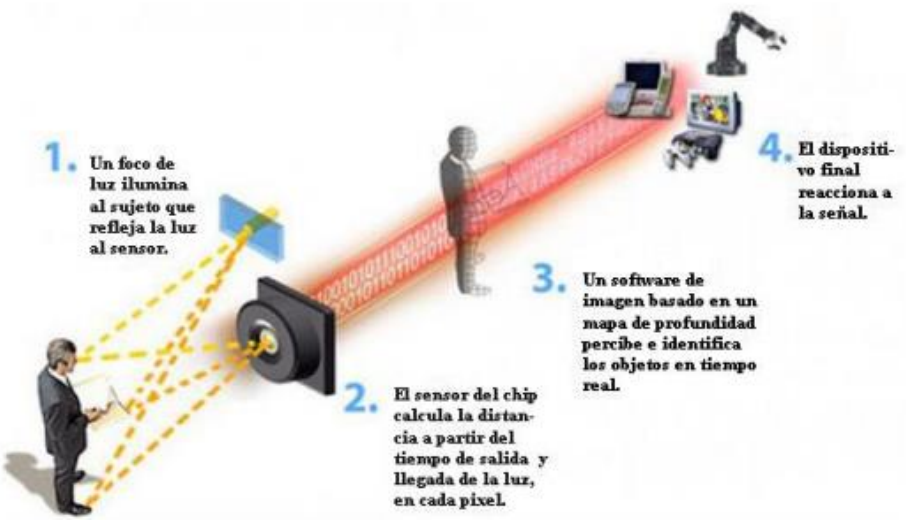


Fuente: Conocimiento propio
Elaborado por: Castro Mychael

2.2.4.4. Funcionamiento

La compañía PrimeSense que detrás del desarrollo del dispositivo Microsoft Kinect explicó el modo de funcionamiento del novedoso dispositivo. Está basada en el método "time-of-flight" de luz infrarroja para medir la profundidad y seguir los movimientos (figura 3) en la cual hace uso de un sistema de doble cámara con una cámara de baja resolución (QVGA 320x240) encargada de medir la profundidad y la otra VGA (640×480) que captura el movimiento y los colores [11].

Figura 3. Funcionamiento detector 3D Kinect



Fuente: Fundamentos de programación
Elaborado por: José Antonio Fernández Valls

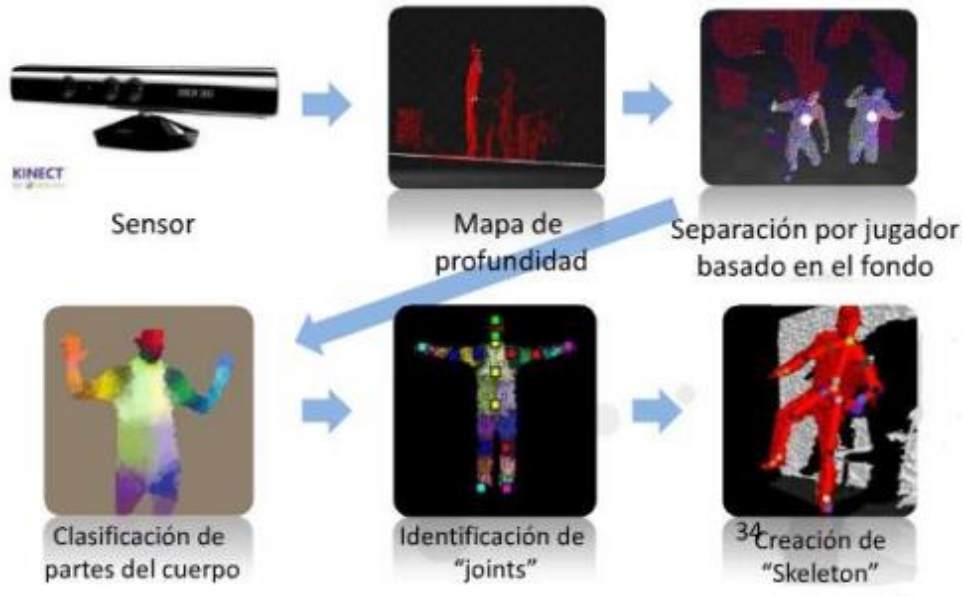
2.2.4.5. Reconocimiento de voz

Para el reconocimiento de voz, Kinect consta con un micrófono estéreo. Esta tecnología de audio es diferente al de los micrófonos convencionales integrados en las webcams de las computadoras portátiles o de escritorio, no poseen una filtrado o cancelación de ruido, funciona con una captura a todas las curvas resultantes de las diferentes intersecciones entre un cono y un plano (cónica) de la acústica en la sala. centrándose en la captura de sonidos como el habla de una o varias personas en una sala a pocos metros de distancia y no de sonidos cercanos como en un teléfono móvil o webcam de computadora [11].

2.2.4.6. Reconocimiento de esqueleto

El reconocimiento del esqueleto ("Skeletal Tracking") es una de las funcionalidades importantes que tiene Kinect (figura 4). Este reconocimiento se lo realiza mediante un software el cual está incluido en el Kinect donde detecta una serie de puntos (20 en total) del esqueleto de los usuarios los cuales representan sus articulaciones. "Estos permiten trabajar las interacciones de los usuarios por parte de los sistemas que utilicen el propio Kinect. El proceso seguido para conseguir los puntos es el resultado de un estudio realizado por investigadores de Microsoft en Cambridge" [11].

Figura 4. Reconocimiento del esqueleto mediante el dispositivo Kinect



Fuente: Nuevas Técnicas de Interacción Basada en Movimiento Aplicadas a Procesos de Rehabilitación
Elaborado por: José Antonio Fernández Valls

2.2.5. Juegos serios en el aprendizaje

Los juegos serios son tácticas estratégicas que tienen como objetivo principal facilitar el aprendizaje y han sido utilizados en todos los niveles en los contextos educativos dejando como segundo plano lo divertido de la actividad. Son reconocidos como un medio significativo para desarrollar el pensamiento estratégico, trabajo de equipo, tolerancia al riesgo, y autoconfianza, entre otros impactos positivos al emplear técnicas lúdicas en aspectos y situaciones no jugables como operaciones militares, empresariales, educativas o de marketing. “Su arribo en el contexto educativo ha sido mediante una transición del juego motriz al juego digital, es decir, del espacio físico al espacio digital” [12].

2.2.6. Gamificación

La gamificación consiste en aplicar las mecánicas empleadas dentro de los juegos en la educación/ámbito laboral con el fin aportar en la mejora de las tareas realizadas dentro de dichas áreas. Las técnicas usadas pueden ser reglas, competencias o premios con el objetivo de “mejorar/aumentar” la motivación y participación activa de las personas involucradas. Existen variadas dinámicas usadas en los juegos que se pueden ejecutar siguiendo esta ideología, como “niveles, puntos, desafíos, espacios virtuales, regalos, status, logros, recompensas, competición, superación, etc.” [13].

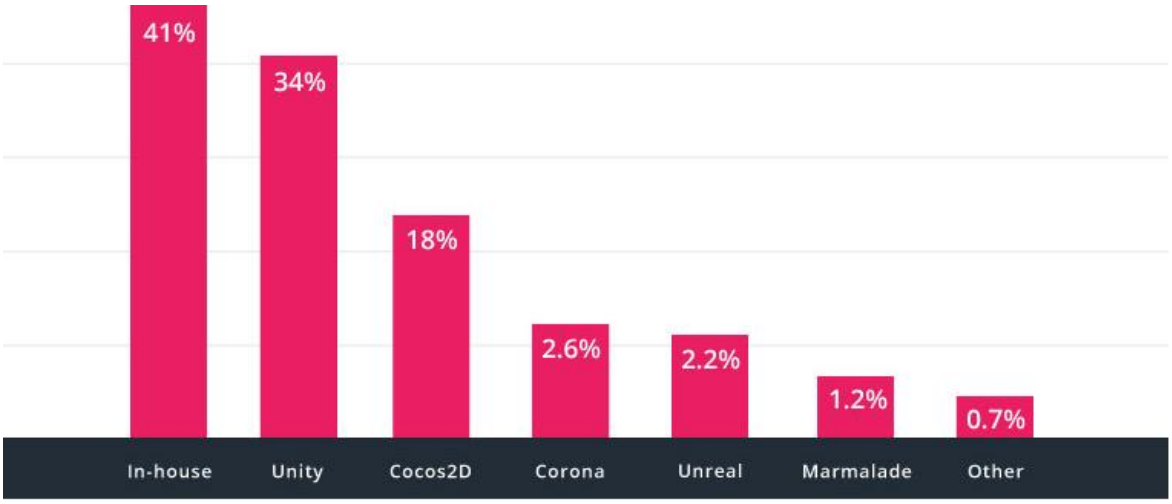
2.2.7. Unity 3D

Unity 3D es una herramienta usada para el desarrollo y creación de videojuegos muy popular en la actualidad, permite la creación de videojuegos en 2D, 3D, VR (realidad virtual) y RA (realidad aumentada), está disponible para plataformas como Microsoft Windows, OS X, y Linux. Esta herramienta disponible al público en diferentes versiones gratuita y profesional disponible en su página oficial.

Unity incluye la herramienta de desarrollo MonoDevelop la cual permite la escritura de scripts en JavaScript, C# y Boo (un dialecto del lenguaje Python), además, también posee una tienda proveedora de recursos (gratuitos y de pago) que permiten extender la herramienta mediante los plugins hallados en la tienda.

Hoy en día el 34% de los 1000 principales juegos han sido creados con Unity (figura 5) y esta aliada con grandes empresas, entre las que destacan Facebook, Google, Microsoft, Samsung, Sony, entre otros. Además, atiende a millones de desarrolladores registrados incluyendo fabricantes importantes, estudios Indie, estudiantes y aficionados de todas partes del mundo.

Figura 5 Porcentaje de juegos realizados con Unity



Fuente: Sitio web oficial de Unity
Elaborado por: Unity

2.3. Marco Referencial

2.3.1. Diseño de interfaces basada en gestos de manuales para facilitar la participación de los estudiantes desde su asiento

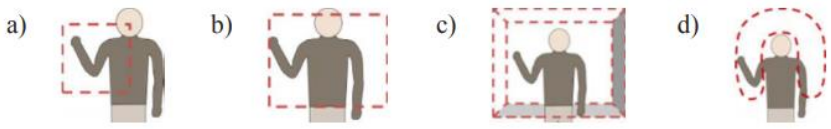
El citado trabajo explora la viabilidad del uso de gestos con las manos sin contacto (THG) durante las clases con el objetivo de incluir activamente la participación de los estudiantes y por ende aumentar su interactividad. Los autores del trabajo definen una serie de aspectos de diseño que deben ser considerados en el desarrollo de este tipo de aplicaciones, basados en una revisión de la literatura y su propia experiencia. Evalúan también la idoneidad de las guías obtenidas mediante un estudio de usuarios, en el cual confirmaron que este estilo interacción es posible y apropiado para ser utilizado en el aula de la manera propuesta [14].

Los aspectos de diseño definidos son los siguientes: el estilo de interacción, el espacio gesto, el conjunto de gestos, y la representación de los estudiantes [14].

El estilo de interacción se refiere al tipo de gestos a realizar en caso de usar toda la mano o los dedos.

Acerca del espacio gestual, los autores definen el alcance sobre la distancia mínima y máxima recomendada para captar las acciones de los usuarios. La figura 6 muestra varios espacios gestuales como el espacio gestual del usuario, el espacio propuesto por McNeill's, todo el espacio accesible con las manos, y el espacio alrededor del usuario.

Figura 6 Espacio de gestos del usuario



Fuente: Diseño de interfaces basadas en gestos manuales para facilitar la participación de los estudiantes desde su asiento
Elaborado por: O. Erazo, N. Baloian, A. Pino, F. Ochoa

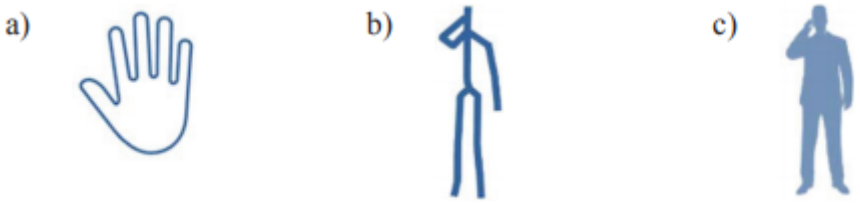
En cuanto al conjunto de gestos, este aspecto está relacionado con los gestos utilizados por los usuarios para la interacción con la aplicación. Los autores indican que no hay un conjunto de gestos definidos para ser utilizados como estándar, pero proporcionan un conjunto de

gestos típicos a ser utilizados para aplicaciones basadas en THG (figura 7) y formas de representar al usuario (figura 8).

Figura 7 Gestos propuestos

Fuente: Diseño de interfaces basadas en gestos manuales para facilitar la participación de los estudiantes desde su asiento
Elaborado por: O. Erazo, N. Baloian, A. Pino, F. Ochoa

Figura 8 Formas de representación de los usuarios



Fuente: Diseño de interfaces basadas en gestos manuales para facilitar la participación de los estudiantes desde su asiento
Elaborado por: O. Erazo, N. Baloian, A. Pino, F. Ochoa

2.3.2. Interfaces de usuario basadas en gestos manuales sin contacto para la sala de clases

Este trabajo consistió en una revisión bibliográfica sobre el uso, diseño y evaluación de interfaces basadas en gestos realizados con la mano sin contacto. Los autores proporcionan pautas para el desarrollo de aplicaciones que puedan ser usadas en las aulas de clases, indicando que son lugares donde la interacción sin contacto puede resultar de gran interés para los estudiantes y de utilidad para mejorar el proceso de enseñanza-aprendizaje apoyándolo con el uso de tecnología [15]. Además, se mencionan dispositivos usables para el seguimiento de la mano y el reconocimiento de gestos, entre los cuales se encuentra la

detección de gestos mediante cámaras. Los autores indican también que la visión por computador es la preferida en la actualidad para el desarrollo de aplicaciones basadas en gestos y sin contacto, usando dispositivos tales como MS-Kinect desarrollado por Microsoft, que de hecho es uno de los preferidos [15].

Por otro lado, el trabajo en mención muestra pautas a considerase para el desarrollo de interfaces basadas en gesto sin contacto indicando que estas son diferentes a las Interfaces de usuario graficas (GUI) dando a conocer principios de diseño para éste tipo de interfaces sin contacto, entre las cuales destacan:

Affordance. Usado para referirse a los atributos de un objeto que permita a los usuarios mediante intuición saber cómo utilizarlos.

Visibilidad. Señala que aquellas funciones que se espera que sean las más utilizadas estén claramente visible.

Retroalimentación (Feedback). Información enviada de vuelta por la aplicación posterior de alguna acción realizada por el usuario (figura 9), para informar que esta o ha finalizado una determinada acción.

Figura 9 Feedback visual al seleccionar un botón mediante gestos "Hover" (izquierda) y "Push" (derecha)

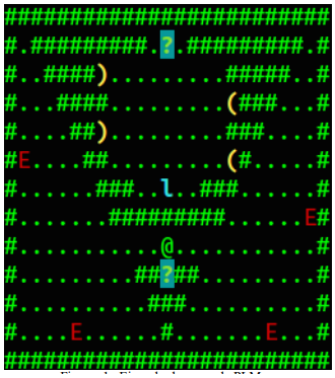
Fuente: Interfaces de usuario basadas en gestos manuales sin contacto para la sala de clases
Elaborado por: O. Erazo, R. Pico.

2.3.3. Gamificación del Proceso de Aprendizaje

Teniendo en cuenta que la Gamificación es la aplicación de estrategias de juegos en contextos no jugables aplicada con la finalidad de que las personas adopten ciertos comportamientos, este artículo muestra la experiencia de los autores en el uso de gamificación para la búsqueda de una docencia más efectiva de la materia de lógica en las titulaciones de grado de ingeniería informática en la Universidad de Alicante mediante su propuesta denominada PLMan [16]. La propuesta PLMan se trata de un videojuego de aprendizaje personalizado, automatizado y gamificado cuyo objetivo es introducir a los

estudiantes en la materia de Lógica aprendiendo el lenguaje de programación Prolog a lo largo de las clases de práctica para superar cada nivel del juego como se puede visualizar en la figura 10, en donde el símbolo "@" es PLMan y debe de comerse todos los cocos (".") evitando las paredes y enemigos ("#", "E") [16]. Para superar cada nivel hay que crear un programa en Prolog (figura 11) con las acciones que debe realizar el personaje principal para cumplir la tarea (comerse los cocos).

Figura 10 Ejemplo del mapa de PLMan



Fuente: Gamificación del Proceso de Aprendizaje
Elaborado por: Faraón Llorens, Francisco Gallego, Carlos Villagrá

Figura 11 Ejemplo de código IA, escrito en Prolog, para superar un mapa de PLMan

solucion.pl

```
:- use_module('pl-man-game/main').

regla :- see(normal, down, '.'), doAction(move(down)).
regla :- see(normal, up, '.'), doAction(move(up)).
regla :- see(normal, right, '.'), doAction(move(right)).
regla :- see(normal, left, '.'), doAction(move(left)).
regla :- see(normal, down, ' '), doAction(move(down)).
regla :- see(normal, right, ' '), doAction(move(right)).
regla :- see(normal, left, '>'), doAction(move(left))
regla :- see(normal, down, '>'), doAction(move(down)).
regla :- see(normal, up, 'l'), doAction(move(up)).
regla :- see(normal, right, '<'), doAction(move(right)).
regla :- see(normal, left, '<'), doAction(move(left)).
regla :- see(normal, left, ' '), doAction(move(left)).
regla :- see(normal, down, '#'), doAction(move(down)).
regla :- see(normal, up, 'l'), doAction(get(up)).
regla :- see(normal, right, 'E'), doAction(move(none)).
regla :- see(normal, down, 'E'), doAction(move(down)).
regla :- see(normal, up, 'E'), doAction(move(up)).
regla :- see(normal, right, 'E'), doAction(move(right)).
regla :- see(normal, left, '.'), doAction(move(left)).
regla :- see(normal, left, ' '), doAction(move(left)).
regla :- see(normal, left, '#'), doAction(use(left)).
regla :- see(normal, down, '('), doAction(move(down)).
```

Fuente: Gamificación del Proceso de Aprendizaje
Elaborado por: Faraón Llorens, Francisco Gallego, Carlos Villagrá

Según lo expuesto en este trabajo, los autores indican que con este sistema han logrado superar los problemas de las innovaciones como exceso de trabajo del profesor, correcciones lentas, feedback desfasado y sobrecarga evaluativa. Los resultados obtenidos en sus estudiantes muestran que con esta propuesta el docente ha conseguido motivarles y divertirles.

CAPÍTULO III

METODOLOGÍA DE LA INVESTIGACIÓN

3.1. Localización

En general, el presente trabajo se realizó en la Universidad Técnica Estatal de Quevedo ubicada en el Cantón Quevedo provincia de los Ríos kilómetro 1 ½ vía a Santo Domingo.

3.2. Tipos de investigación

3.2.1. Investigación Descriptiva

El presente proyecto se realizó bajo la investigación descriptiva, debido a que, para el desarrollo del mismo, en primera instancia se llevó a cabo un análisis exhaustivo de la información pertinente, entre las consideradas: artículos científicos, revistas, libros, trabajos realizados y grupos de enfoque para obtener características, y posteriormente fijar los alcances, protocolo, limitaciones y resultados que se pueden llegar a presentar. Partiendo de esta determinación de factores influyentes establecer qué tipo de tecnología podría ofrecer mejores resultados para el desarrollo del proyecto.

3.2.2. Investigación Exploratoria

Dentro de las diferentes características que fueron analizadas como el marco referencial para el desarrollo de este trabajo, ninguna utilizaba una interfaz de usuario natural (NUI) la cual mediante la integración de técnicas lúdicas y usando las actuales tecnologías de modelado y creación de juegos se puede obtener un software atractivo dentro del proceso de enseñanza-aprendizaje para de esta manera obtener una herramienta que coadyuve a dicho proceso.

3.3.Métodos y técnicas a usar en la investigación

3.3.1. Método bibliográfico

Se utilizó para constatar que el desarrollo de una aplicación educativa la cual haga uso del paradigma de interacción sin contacto propondría un método distinto al autoaprendizaje y el desarrollo de la lógica de programación ante la metodología que comúnmente estamos acostumbrados en la cual recae la mayoría de responsabilidad al docente.

3.3.2. Método inductivo

El software se fue evaluado por un grupo de usuarios con conocimientos sólidos en la programación de computadoras mediante un estudio de usuarios dentro del cual se asignaron una serie de tareas de manera tal que los participantes posteriormente de resolverlas dieron a conocer la aceptación del software e indicar si ésta es una herramienta apropiada para el aprendizaje de la lógica de programación dentro del cual se obtuvieron comentarios favorables para la misma.

3.3.3. Método deductivo

La ejecución de las tareas que contiene el software en sus distintos niveles de dificultad propuestos hacen que la persona se involucre en el proceso de enseñanza-aprendizaje de una manera diferente a la común enseñanza del profesor al alumno, permitiendo que se incluyan activamente en el proceso de manera tal que mejoran su lógica de programación.

3.4. Recursos y materiales

3.4.1. Recursos de hardware

Tabla 1. Requerimientos del hardware

Cantidad	Equipo	Características
1	Computador portátil	✓ Intel® Core™ i7 2.39GHz ✓ Memoria RAM 8 Gb. ✓ 1 TB Disco Duro
1	Impresora	✓ Ricoh Aficio M4540
1	Memoria USB	✓ Kingston DataTraveler® 100 G3 ✓ Capacidad 16Gb
1	Ms-Kinect	✓ Kinect para Xbox 360 V1

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

3.4.2. Recursos Software

Tabla 2. Requerimientos del software

Tipo	Descripción
Software Utilitario	MS Office 2013 • Word • Excel • Power Point
Adobe Photoshop CC	Ediciones de imágenes
Adobe Illustrator CC	Ediciones
Microsoft Visual Studio	Entorno de programación v2015
Microsoft SQL Server	Gestor de base de datos
DBMS SQLite	Sistema de base de datos relacional embebida

Fuente: Conocimiento propio.
Elaborado por: Castro Mychael (2017)

3.4.3. Materiales de oficina

- 3 Resmas papel bond tamaño A4

- Internet
- Flash Memory 8 Gbg
- 10 bolígrafos

3.4.4. Presupuesto

Tabla 3. Presupuesto del proyecto de investigación

Cantidad	Descripción	Valor Unitario	Valor Total
1	Kinect para Xbox 360	$57,95	$57,95
		TOTAL	$57,95

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

CAPÍTULO IV

METODOLOGÍA DE DESARROLLO

4.1. Desarrollo del software

4.1.1. Metodología en cascada

Para el desarrollo del prototipo se utilizó la metodología de desarrollo en cascada. Esta metodología se adaptó fácilmente a la secuencia lineal requerida para la creación del prototipo del software. La metodología consta de las siguientes fases:

1) Análisis de requerimientos. Determinar las necesidades del usuario o problemática a resolver, cuál será su alcance y requisitos.

2) Diseño del software. Determinar las funcionalidades que va a tener el software mediante el uso de diagramas de flujo, casos de usos, Storyboards, previo al desarrollo.

3) Fase de codificación. Es la etapa de codificación propiamente dicha, es donde se escriben los algoritmos y flujos a seguir en un lenguaje de programación que cumpla con lo establecido y objetivos del software.

4) Fase de pruebas. Los flujos y algoritmos ya programados son comprobados con un grupo de usuarios para demostrar su funcionalidad y aceptación.

A continuación, se presentan los detalles concernientes a cada una de las fases.

4.1.2. Análisis de requerimientos

El prototipo del software que se pretende desarrollar permitirá interactuar con los usuarios de manera “natural”, haciendo uso del paradigma de interacción sin contacto basado en gestos con las manos. Para ello se utilizará el sensor MS-Kinect y el motor de videojuegos Unity 3D usado para el modelado de la interfaz y personajes.

El software ayudará a los usuarios en la mejora de su lógica de programación. Para ello, el software dispone de breves explicaciones acerca del uso de los operadores y estructuras de control usados en programación. Además, se usa un ambiente cotidiano (como lo es el recorrido de un autobús en una ciudad) dentro del cual las sentencias de programación son representadas por objetos dentro de la escena con comportamiento similar, para que de esta manera las asociaciones del funcionamiento de las mismas resulten más fáciles de interpretar y usar.

4.1.3. Diseño del sistema

4.1.3.1. Patrón de arquitectura del software

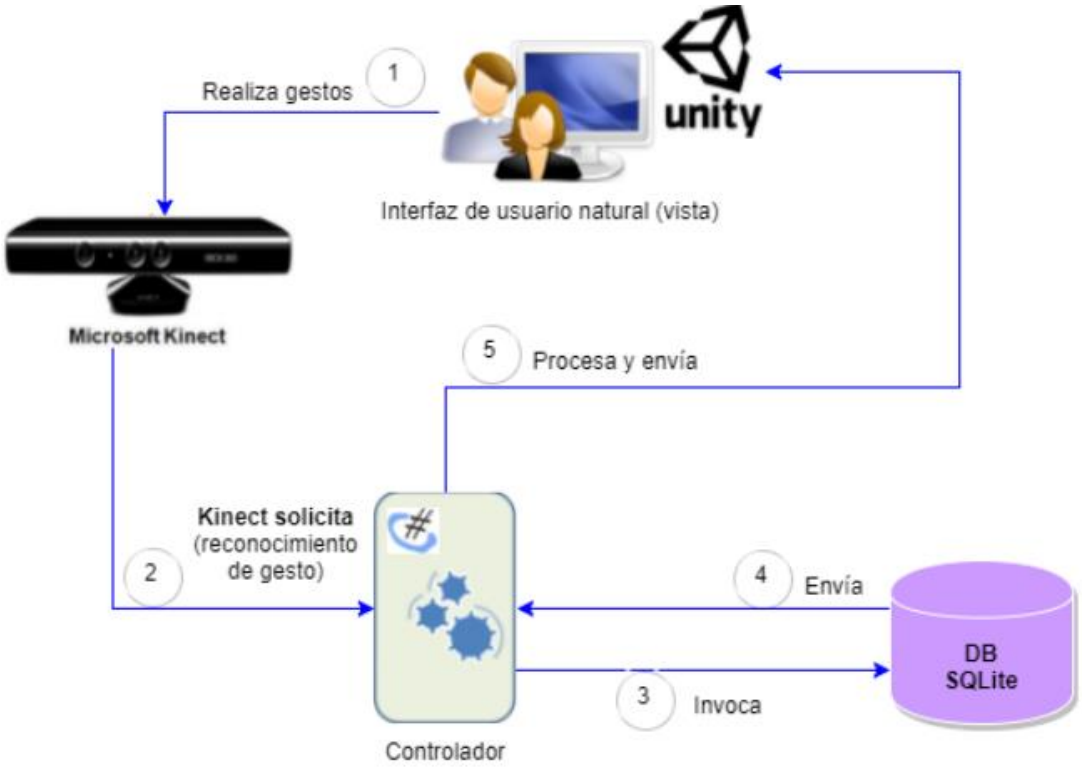
Para el desarrollo del software se utilizó la arquitectura MVC (Modelo Vista Controlador, figura 12). Esta técnica de desarrollo consiste en dividir el software en 3 componentes esenciales:

1) Modelo: Dentro de esta fase se incluyen las entidades encargadas del funcionamiento de lectura y escritura de datos hacia los archivos de configuración del juego y/o base de datos.

2) Vistas: Aquí se incluyen las escenas o pantallas de interacción y detección de movimiento visibles al usuario.

3) Controlador: Son entidades que proporcionan la comunicación entre la parte visual y el modelo, resolviendo las peticiones solicitadas desde la vista al modelo, para en él ser procesadas y retornar los resultados solicitados.

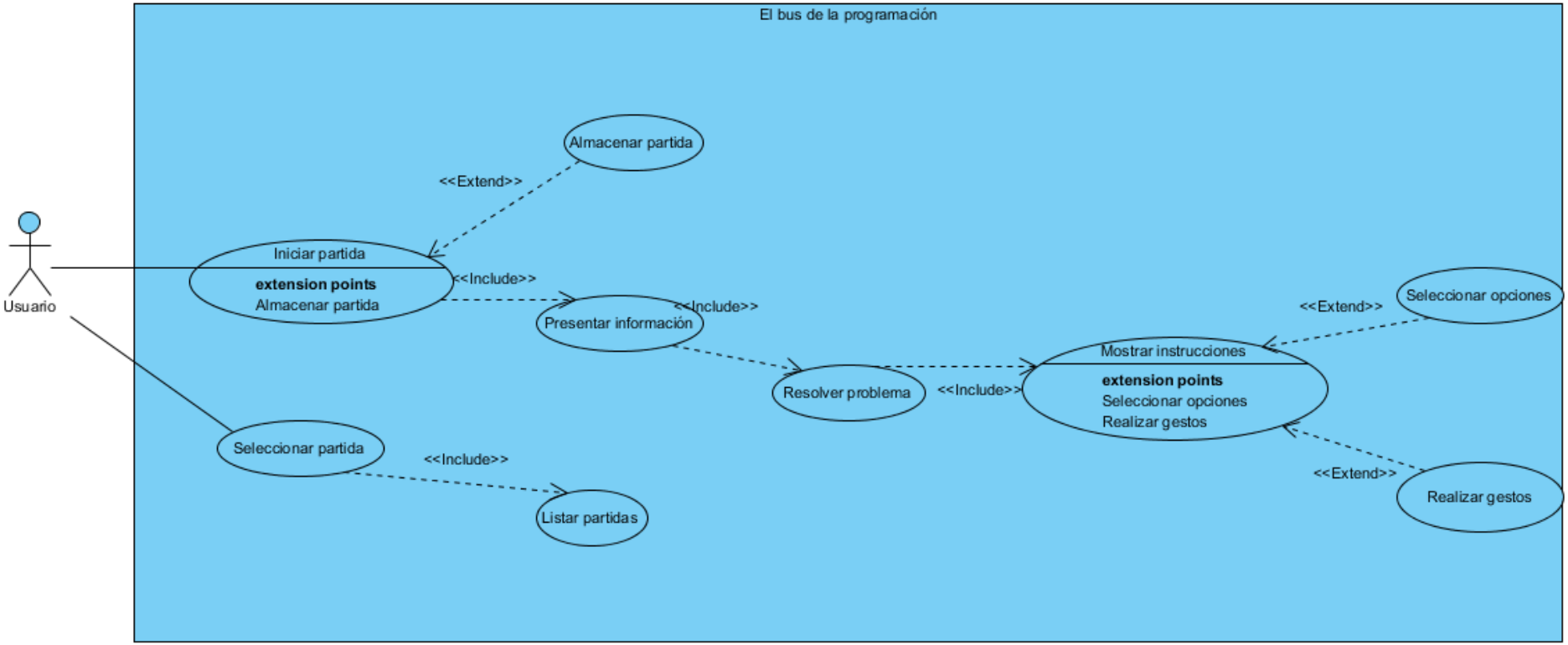
Figura 12 Patrón de arquitectura de Software



Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

4.1.3.2. Diagrama de casos de uso del sistema

Figura 13 Casos de uso del sistema



Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

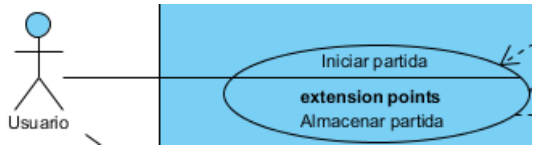
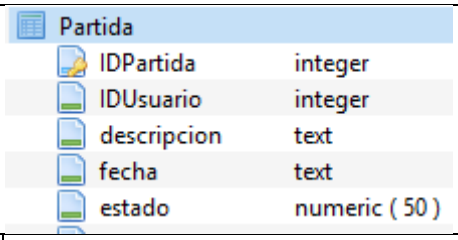
4.1.3.3. Casos de uso extendido

Tabla 4. Caso de uso: Iniciar partida

<table>
<tr><th colspan="3">CASO DE USO: Iniciar partida</th></tr>
<tr><td colspan="2">Tipo: REAL</td><td>Prioridad: ALTA</td></tr>
<tr><td colspan="2">Actores: Jugador</td><td>Paquete:</td></tr>
<tr><td colspan="3">Propósito: Iniciar una nueva partida.</td></tr>
<tr><td colspan="3">Resumen: Permite que el jugador inicie una partida (juego)</td></tr>
<tr><th colspan="3">FLUJO NORMAL</th></tr>
<tr><th colspan="2">ACTOR</th><th>RESPUESTA DEL SISTEMA</th></tr>
<tr><td colspan="2">1. Hace el gesto Touch en Iniciar Juego.

3. Visualiza información del funcionamiento de las sentencias a usarse con ejemplos y hace el gesto Touch para continuar.

5. Termina el caso de uso.</td><td>2. Crea una nueva partida, y muestra la vista de inicio del sistema, con la información inicial para la comprensión de las primeras etapas (tareas).

4. Presenta ordenadamente las escenas de información al jugador.</td></tr>
<tr><th colspan="3">FLUJO ALTERNO</th></tr>
<tr><td colspan="2"></td><td></td></tr>
<tr><th colspan="2">CASO DE USO RELACIONADOS</th><th>ESCENA</th></tr>
<tr><td colspan="2"> 
</td><td> 
</td></tr>
<tr><th colspan="3">BASE DE DATOS</th></tr>
<tr><td colspan="3"> 
</td></tr>
<tr><td>GESTOS</td><td colspan="2">Touch (clic), seguimiento de mano (cursor)</td></tr>
<tr><td>OBJETOS</td><td colspan="2">Partida</td></tr>
<tr><td>MÉTODOS</td><td colspan="2">nuevaPartida(GameObject Player);</td></tr>
<tr><td>VALIDACIONES</td><td>ValidarPartidaEnProceso ();
KinectStatus ();</td><td>Tabla: Partida
Campos: IDPartida, IDUSuario, Descripcion, FechaRegistro, Estado.
Formato: entero, texto;</td></tr>
</table>

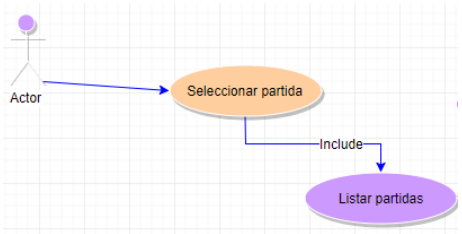
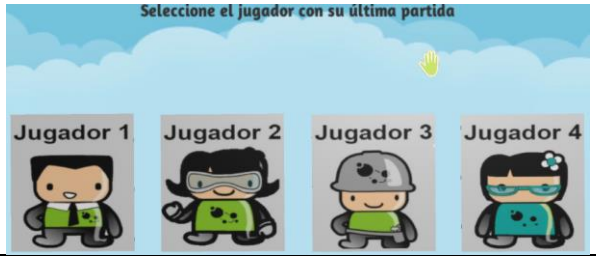
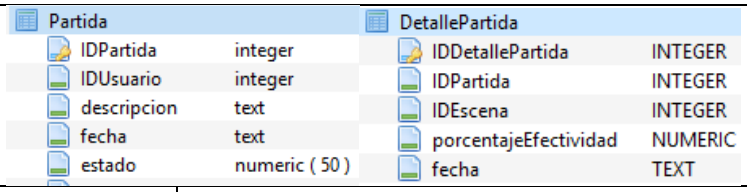
Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 5. Caso de uso: Seleccionar partida

<table>
<tr><th colspan="3">CASO DE USO: Seleccionar partida</th></tr>
<tr><td colspan="2">Tipo: REAL</td><td>Prioridad: Media</td></tr>
<tr><td colspan="2">Actores: Jugador</td><td>Paquete:</td></tr>
<tr><td colspan="3">Propósito: Seleccionar una partida guardada.</td></tr>
<tr><td colspan="3">Resumen: Permite que el jugador seleccione una partida previamente guardada, para iniciar el juego desde un punto específico en el cual se haya quedado.</td></tr>
<tr><th colspan="3">FLUJO NORMAL</th></tr>
<tr><th colspan="2">ACTOR</th><th>RESPUESTA DEL SISTEMA</th></tr>
<tr><td colspan="2">1. Hace el gesto Touch en Cargar partida.

3. Visualiza una lista de partidas y selecciona la que desea cargar mediante el Gesto Touch (el cual representa el clic) en el software.

5. Termina el caso de uso.</td><td>2. Muestra una lista de las partidas guardadas.

4. Realiza una petición mediante el controlador al modelo el cual obtiene los datos especifico del punto de la partida en el que el usuario desea volver a jugar.</td></tr>
<tr><th colspan="3">FLUJO ALTERNO</th></tr>
<tr><td colspan="2">1. El usuario no visualiza ninguna partida en la lista.</td><td>2. El sistema muestra un mensaje indicando que no se han registrado partidas en la base de datos.</td></tr>
<tr><th colspan="2">CASO DE USO RELACIONADOS</th><th>ESCENA</th></tr>
<tr><td colspan="2"> 
</td><td> 
</td></tr>
<tr><th colspan="3">BASE DE DATOS</th></tr>
<tr><td colspan="3"> 
</td></tr>
<tr><td>GESTOS</td><td colspan="2">Touch (clic), Seguimiento de mano (cursor)</td></tr>
<tr><td>OBJETOS</td><td colspan="2">Partida</td></tr>
<tr><td>MÉTODOS</td><td colspan="2">ListarPartidas();
CargarPartida() ;</td></tr>
<tr><td>VALIDACIONES</td><td>KinectStatus ();</td><td>Tabla: DetallePartida
Campos: IDDetallePartida, Escena,
Formato: entero, texto;
Tabla: Partida
Campos: IDUSuario.
Formato: entero.</td></tr>
</table>

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

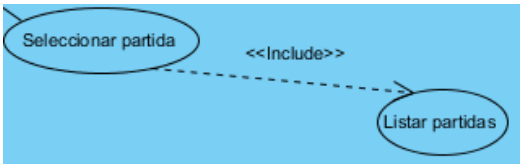
Tabla 6. Caso de uso: Mostrar información (sentencia)

<table>
<tr><th colspan="3">CASO DE USO: Mostrar información (sentencia)</th></tr>
<tr><td colspan="2">Tipo: REAL</td><td>Prioridad: Alta</td></tr>
<tr><td colspan="2">Actores: Jugador</td><td>Paquete:</td></tr>
<tr><td colspan="3">Propósito: Presentar información referente al uso de sentencias y operadores.</td></tr>
<tr><td colspan="3">Resumen: Muestra información acerca del modo y lógica de uso de los operadores y estructuras de control, con el objetivo de que el jugador tenga una idea concisa acerca de cómo usarlos y en que situaciones hacer uso de los mismos.</td></tr>
<tr><th colspan="3">FLUJO NORMAL</th></tr>
<tr><th colspan="2">ACTOR</th><th>RESPUESTA DEL SISTEMA</th></tr>
<tr><td colspan="2">2. lee escenas de información acerca del uso de los operadores y/o sentencias.

4. Repite el paso 2.

5. Termina el caso de uso.</td><td>1. Muestra escena de información acerca del uso y lógica de los operadores.

3. Muestra la siguiente escena en caso de existir más.

4. Finalizan las escenas de información.</td></tr>
<tr><th colspan="3">FLUJO ALTERNO</th></tr>
<tr><td colspan="2">1. El usuario no visualiza ninguna partida en la lista.</td><td>2. El sistema muestra un mensaje indicando que no se han registrado partidas en la base de datos.</td></tr>
<tr><th colspan="2">CASO DE USO RELACIONADOS</th><th>ESCENA</th></tr>
<tr><td colspan="2"> 
</td><td> 
</td></tr>
<tr><th colspan="3">BASE DE DATOS</th></tr>
<tr><td colspan="3"> 
</td></tr>
<tr><td>GESTOS</td><td colspan="2">Touch (clic), Seguimiento de mano (cursor)</td></tr>
<tr><td>OBJETOS</td><td colspan="2">Partida</td></tr>
<tr><td>MÉTODOS</td><td colspan="2">ListarPartidas();
CargarPartida() ;</td></tr>
<tr><td>VALIDACIONES</td><td>KiectStatus ();</td><td>Tabla: DetallePartida
Campos: IDDetallePartida, Escena,
Formato: entero, texto;
Tabla: Partida
Campos: IDUSuario.
Formato: entero.</td></tr>
</table>

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 7. Caso de uso: Resolver problema

<table>
<tr><th colspan="2">CASO DE USO: Resolver problema</th></tr>
<tr><td>Tipo: REAL</td><td>Prioridad: ALTA</td></tr>
<tr><td>Actores: Jugador</td><td>Paquete:</td></tr>
<tr><td colspan="2">Propósito: Presentar las preguntas lógicas necesarias para que el usuario resuelva una tarea.</td></tr>
<tr><td colspan="2">Resumen: Presentar ordenada y sistemáticamente un flujo de preguntas necesarias para que el jugador tome una decisión y luego realizar una elección sobre ellas.</td></tr>
<tr><th colspan="2">FLUJO NORMAL</th></tr>
<tr><th>ACTOR</th><th>RESPUESTA DEL SISTEMA</th></tr>
<tr><td>1. Visualiza la escena, y espera la tarea propuesta por el sistema.

3. Observa el problema planteado por el sistema.

5. Realiza el gesto correspondiente para seleccionar o representar una sentencia, indicada por el juego.

7. Repite el paso 5 hasta culminar las preguntas.

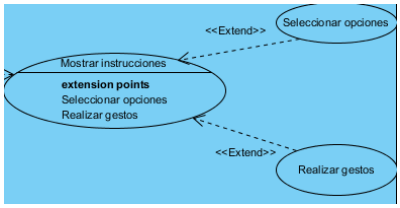
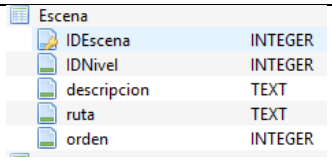
9. Fin del caso de uso.</td><td>2. Genera una tarea relacionada con una acción cotidiana dentro de la escena.

4. Realiza una pregunta al usuario para que seleccione según su pensamiento qué opción y gesto usar.

6. Muestra la siguiente pregunta para llegar a la solución del problema planteado.

8. El sistema muestra la simulación del algoritmo construido por el jugador, poniéndolo en práctica dentro de la escena.</td></tr>
<tr><th colspan="2">FLUJO ALTERNO</th></tr>
<tr><td>5. El usuario no realiza el gesto de selección correcto.

8. Repite el paso 5.</td><td>6. El sistema muestra un mensaje indicando que se ha equivocado.

7. Vuelve a mostrar la pregunta hecha para que el usuario la interprete nuevamente.</td></tr>
<tr><th>CASO DE USO RELACIONADOS</th><th>ESCENA</th></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><th colspan="2">BASE DE DATOS</th></tr>
<tr><td colspan="2"> 
</td></tr>
<tr><td>GESTOS</td><td>Touch (clic), Seguimiento de mano (cursor), condicional, iteración.</td></tr>
<tr><td>OBJETOS</td><td>Partida, Jugador, Escena</td></tr>
</table>

MÉTODOS	CargarAnimaciones(); FlujoPreguntas(GameObject problema) ;	
VALIDACIONES	KinectStatus ();	Campos varios.

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

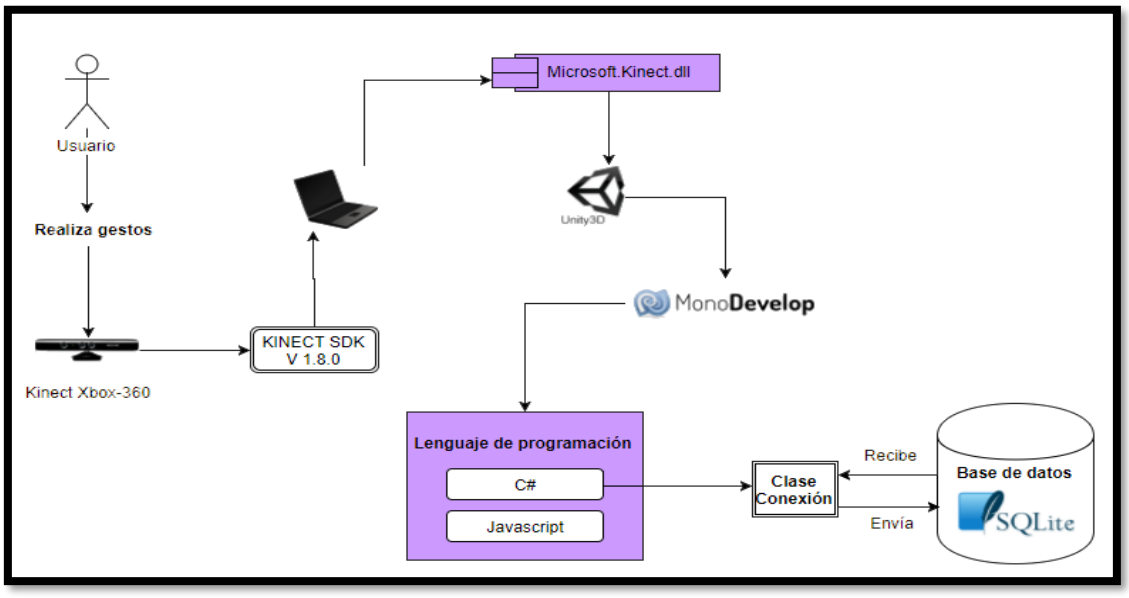
4.1.4. Fase de codificación

4.1.4.1 Integración

Dentro de esta fase se realizó la integración entre el sensor Kinect y el motor para desarrollo de videojuegos Unity 3D (figura 14). La arquitectura representa la interacción realizada por el usuario hacia el sistema, en donde el Kinect está conectado a una computadora en la que se encuentra el software desarrollado en Unity 3D y recibe los gestos realizados por el usuario.

La conexión entre Kinect y Unity 3D se realiza mediante una librería (Kinect.dll), la misma que está desarrollada en el lenguaje de programación C#, lenguaje que permitió el acceso a lectura y escritura de datos en el sistema manejador de base de datos SQLite, que es una base de datos embebida que ofrece la ventaja de ser portable. La forma de integración de estas tecnologías permite un adecuado funcionamiento del software.

Figura 14 Arquitectura final del prototipo del software



Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

4.1.5. Pruebas

4.1.5.1 Fase 1

Dentro de la primera fase se realizó las pruebas en un ambiente de baja luminosidad, con 3 usuarios de conocimientos básicos en programación para determinar la aceptabilidad del prototipo desarrollado. Una vez realizada las pruebas se detectó que los participantes no les resultaban familiares los términos técnicos usados, como nombre de las sentencias, el condicional if y ciclos de control como while, for y do while. Además, los participantes indicaron que los personajes y objetos dentro del juego deberían tener una animación.

4.1.5.1 Fase 2

En la segunda fase se reestructuraron los nombres de las sentencias con términos más comunes y de fácil interpretación, también se asignó una animación a los personajes, como movimiento real a las personas, humo espeso al bus y animación del bus en su recorrido por la ciudad, obteniendo un resultado favorable y atractivo para la vista de los participantes. En estas pruebas participaron los mismos sujetos de la primera fase para verificar el cumplimiento de sus expectativas iniciales. Además, también se incluyó a dos participantes de conocimiento intermedio en programación, los cuales manifestaron que el juego carecía de una historia que involucre más al usuario.

4.1.5.2 Fase 3

Dentro de la fase 3 se incluyó una pequeña historia dentro del juego, la cual solicita la resolución de problemas cotidianos que asocian el uso de sentencias de programación, haciendo que los participantes puedan determinar que sentencia y su gesto correspondiente a utilizar en el momento adecuado, con la finalidad de resolver una tarea determinada. También se realizó pruebas de funcionamiento con usuarios diferentes y las mismas tareas realizadas en las anteriores fases, en la cual no se obtuvieron más observaciones.

4.1.5.3 Fase 4

Una vez que se detectaron y corrigieron las posibles falencias del software, se realizaron las pruebas finales con 3 participantes con conocimientos básicos que resolvieron las tareas propuestas en las fases anteriores, en la cual no se evidenció problema alguno. De esta manera se pudo comprobar el correcto funcionamiento del software que coadyuve al proceso de enseñanza-aprendizaje de la lógica de programación.

CAPÍTULO V

RESULTADOS

5.1. RESULTADOS

5.1.1 Planteamiento de las características que debe tener la interfaz de usuario para facilitar el aprendizaje de la lógica de programación

Como primer paso para el análisis de la propuesta, es necesario conocer las opiniones/percepciones de otras personas sobre la idea de usar interacción sin contacto y juegos serios para apoyar el aprendizaje de programación. Para el efecto, se realizó un grupo de enfoque (Focus Group) [17] con el objetivo de obtener las características que posteriormente serían incluidas software. El investigador actuó como moderador guiando la realización del grupo de enfoque en base al plan previamente preparado. En el grupo de enfoque se discutió también sobre el escenario y los gestos que podrían utilizarse, complementándose con un conjunto de preguntas objetivas preparadas con anticipación de acuerdo a los objetivos previstos. Siete personas participaron en el grupo de enfoque, lo cual entra en el rango sugerido para el uso de esta técnica ("entre 6 y 10 personas" [17]). Aunque se invitaron a hombres y mujeres a participar, sólo tomaron parte hombre con edades entre los 22 y 24 años y con una experiencia básica en programación básica (junior). Así, la tabla 8 detalla los resultados obtenidos sobre las opiniones de los participantes.

Tabla 8. Resultados del Grupo de enfoque

OBJETIVO	PREGUNTAS
1.- Determinar el nivel de dificultad en el aprendizaje de las sentencias básicas de programación.	**1.- ¿Cuál sentencia resultó más complicada de aprender en programación?** Se obtuvo conocimiento respecto a qué sentencias de programación resultan complicadas de entender inicialmente. Todos los participantes manifestaron haber tenido problemas con las estructuras de repetición (While, Do While, For y Foreach). Estos problemas repercutían en la dificultad de interpretar el número de ciclos necesarios para resolver de manera exacta una tarea. **2.- ¿Por qué ésta sentencia le resultó complicada?** Se mencionaron diferentes causas. En primer lugar, la sentencia Foreach debido a la interpretación de

	la misma (no asimilaban la manera de recorrer colecciones de objetos y no variables). Los participantes también hicieron referencia al poco uso de la sentencia, comentando que por las enseñanzas del colegio en otros lenguajes entre ellos C++, en escasas ocasiones usaron ésta sentencia. Un segundo caso es While y Do While. Aquí existía una confusión; no tenían claro el concepto de aplicación: While se ejecuta o no, y Do While se ejecuta al menos una vez. Por otra parte, una causa a tener en cuenta es la dificultad de retener en la mente la cantidad de ciclos necesarios para realizar una tarea; el retener ésta información en la mente resulta complicado inicialmente. **3.- ¿Qué tiempo invirtió en dominar esta sentencia?** Los tiempos obtenidos para el dominio de las sentencias variaron desde lo micro a lo macro llegando a ser desde 1 hora y media hasta 3 meses.
2. Determinar un escenario donde se pueda representar las sentencias básicas.	**1.- ¿Qué escenario consideraría brinda facilidades para exponer los conceptos básicos de las sentencias de programación?** Al realizar esta pregunta los participantes hicieron referencia a qué escenarios les parecía idóneos. Entre los principales cuales se mencionaron: • Recorridos de buses • Ventas de Quesos de leche • Viaje a la playa • La Universidad **2.- ¿Por qué es adecuado el escenario propuesto?**

	Los participantes respondieron que los escenarios propuestos serían adecuados gracias a la similitud de acontecimientos ocurrentes dentro de ellos respecto a las sentencias de programación, y que les resultaría más sencillo interpretar la lógica implícita de una sentencia con un acontecimiento semejante dentro de la escena. **3.- ¿Qué facilidades brinda el escenario para el aprendizaje?** En este punto los participantes determinaron que un buen escenario era el que realizan los buses al recorrer una ciudad, dando a conocer sus ideas que facilitarían el entendimiento tales como: si un semáforo está en rojo tendrían que usar condicionales (IF) para detener el bus, o que en caso de que se acabase el combustible del bus tendrían que reponerlo comprando más. **4.- ¿Qué objetos ofrece el escenario que se asemejen con sentencias de programación?** Dentro del escenario de la venta de quesos de leche relacionaron los ingredientes y la cantidad de leche a utilizar para la preparación con los condicionales y ciclos. Por otra parte, en el caso de los buses se establecieron varios ejemplos como los semáforos, calles dañadas, peatones, paradas de los buses, entre otras causas como un alza en la tarifa de los pasajes. **5.- ¿Qué analogías se pueden aprovechar en este escenario para el aprendizaje de las sentencias de programación?** Los participantes determinaron la relación de semejanza entre los objetos de la escena y sentencias de programación, dando a conocer sus

	ejemplos tales como detenerse en casos de accidentes o por tráfico en hora pico.
3- Vincular las sentencias con objetos de similar comportamiento que se hallen dentro de la escena.	**1.- ¿Qué similitudes podrían existir entre el comportamiento de los objetos de la escena propuesta y las sentencias de programación?** Como ideas principales se citaron ejemplos como: • Los semáforos y calles en mal estado se pueden relacionar con los condicionales. • Las vueltas que tiene que cumplir un chofer al día para terminar su ciclo se asemejan con las sentencias repetitivas. **2.- ¿Qué ejemplo práctico podríamos desarrollar para lograr el aprendizaje más rápido de las sentencias de programación?** Esencialmente se citaron ejemplos ya comentados como los semáforos y calles. **3.- Finalmente, con todos los ejemplos obtenidos someterlos a votación para ver cuál resulta ser el más apropiado, según los participantes.** Determinaron que de los ejemplos propuestos, el recorrido de los buses era muy bueno y fácil de entender, teniendo en cuenta la semejanza que se podría lograr con el uso de los semáforos, calles, paradas de los buses y tráfico.
4.- Obtener la representación de gestos que mejor se asemejan a cada sentencia básica.	**1.- ¿Qué gesto podemos realizar para representar cada sentencia?** Los participantes especularon que los condicionales se podían representar colocando la mano derecha horizontal, y si tuviera un caso contrario levantarían la mano izquierda horizontalmente. Este ejemplo también lo propusieron de la misma manera, pero con las manos levantadas.

	Para los ciclos, con una mano (derecha) hacer un círculo o desplazamiento y con la otra mano especificar el tipo de ciclo de la siguiente manera: La mano izquierda por encima de la cabeza y la derecha haciendo circulo (do While). La mano derecha por encima de la cabeza y la derecha haciendo circulo (While). Ambas manos por encima de la cabeza y luego la derecha haciendo circulo (For). **2.- ¿Es el gesto indicado para la representación de la sentencia?** Los participantes expresaron que, mediante la representación antes realizada, el gesto resultaba fácil, y sin mucha fatiga. **3.- ¿Por qué es el mejor gesto?** Por su facilidad de realizarlos, opinaron los participantes.

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

En base a los resultados resumidos en la tabla 8, se realizó un análisis para obtener las características que se deberían incluir en el prototipo a desarrollar. A continuación, se resumen tales características.

- Utilizar un escenario que permita la interpretación entre objetos y sentencias de programación de manera sencilla.
- Representar las estructuras de control en programación mediante gestos cortos y sencillos realizados con la mano.
- Representar al usuario con un personaje para el uso de estructuras de control y cursor de mano para la navegación.
- Utilizar animaciones en objetos dentro del juego.
- Emitir sonidos de retroalimentación al usuario indicando el gesto realizado o en proceso.

5.1.2 Identificación de las tecnologías adecuadas para el desarrollo del prototipo del software

Con miras a determinar las tecnologías más adecuadas para el desarrollo del software, se procedió a realizar un análisis de las mismas. Para ello se elaboraron dos cuadros de comparación de las tecnologías existentes que cumplan las necesidades básicas para el desarrollo del software. Además, se tomó también en cuenta que las tecnologías a usar sean de fácil uso y mantenimiento, y que aporten positivamente al objetivo principal de la investigación. Así, la tabla 9 muestra las características de las diferentes tecnologías para el desarrollo de videojuegos de moda en la actualidad y la tabla 10 respecto a los sensores que permiten obtener el movimiento del usuario y así hacer seguimiento de los gestos realizados por ellos.

Tabla 9. Comparación entre motores para el desarrollo de videojuegos

CARACTERISTICAS	UNITY	CONSTRUCT 2	GODOT
ICONO			
Software libre	SI (versión estudiantil)	NO	SI(código abierto)
Idioma	MULTILINGÜE	INGLÉS	INGLÉS
Máx. Eventos	ILIMITADO	10	ILIMITADO
Máx. Capacidad	ILIMITADO	4	ILIMITADO
Búsquedas de eventos	SI	NO	NO
Exportar juegos	SI	SI	SI
Motor	2D Y 3D	2D Y 3D	2D Y 3D
Compilación en la nube	SI	NO	NO
Tienda de archivos	SI	NO	NO
Grabar y compartir videos de los usuarios	SI	NO	NO
Integración con Hardware	SI Mediante librerías	SI, con Script	Limitado
Comentarios adicionales	Una de las plataformas para el desarrollo de videojuegos 2D y 3D más completas de la actualidad, permitiendo desarrollo de juegos para múltiples plataformas (MAC, HTML5 iOS, Android, entre otras), tiene soporte para	Herramienta que hace sencillo el desarrollo de videojuegos, basado en html5 dirigido a personas con pocos conocimientos de programación usando un editor visual y sistema	Este software esta publicado bajo Licencia MIT la cual fue desarrollado por la comunidad de Godot. Actualmente funciona para Windows, OS X, Linux y BSD. Permite funcionalidades de exportación hacia

	mecanim, animation, juegos en Linux, etc.	lógico basado en comportamientos	PC, Dispositivos móviles y HTML5

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

En base a las características planteadas de los diferentes motores para el desarrollo de juegos, se determinó que la mejor opción sería Unity. Esta decisión se tomó teniendo en cuenta sus ventajas sobre las demás en cuanto al desarrollo 3D, su favorable documentación, foros de ayuda, y excelente integración mediante librerías con diferentes dispositivos, entre ellos sensores para la detección de articulaciones y esqueletos de las personas.

Tabla 10. Comparación de sensores para el reconocimiento de gestos

CARACTERISTICAS	KINECT FOR XBOX-360	Leap Motion	Intel RealSense
ICONO			
Desarrollador	Microsoft	Leap Motion	Intel
Rango de distancia usable	Entre 1,2 hasta 3,5 metros	Entre 0,02 a 0,5 metros	Entre 0,5 y 3,5 metros
Rastrear todo el cuerpo	Si	No	No
Precisión	1.3mm	1/100mm	800mm
Reconocimiento de voz	Si	No	Si
Cámaras	Si	2 monocromáticas	3 cámaras (RGB, infrarroja, profundidad)
Conectividad	USB con adaptador	USB 2.0	USB3 (de 130 x 20 x 7 mm)
Precio	$57,95 (Amazon store)	$ 75,83(Amazon store)	$ 148,42 (Amazon store)
Sistema de seguimiento	Todo el cuerpo, detecta hasta 6 en total de entre ellas 2 activas.	Solo manos	Manos y rostro 1 persona.

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Analizando las características y ventajas de los sensores de moda en la actualidad, Kinect supera en algunas características a Leap Motion entre las cuales se destacan el reconocimiento del cuerpo completo y rango de distancia. Sin embargo, Intel RealSense promete ser superior a Kinect en cuanto a las características y precisión para realizar tareas,

pero en cuanto a la compatibilidad con la versión 5 de la herramienta para desarrollo de videojuegos seleccionada presenta algunos inconvenientes. Por lo tanto, se determinó que el sensor a utilizarse sea Kinect de Microsoft.

5.1.3 Resultados del desarrollo del software

Una vez determinadas las características a incluir en el software, y la tecnología a utilizar para ello, se procedió a la construcción del software. Esta sección presenta de forma resumida las pantallas que forman parte del prototipo resultante.

En primer lugar, la pantalla para la selección del personaje (figura 15) permite seleccionar el personaje y cargar la escena donde el jugador quedó por última vez. Para seleccionar el jugador bastará con hacer el gesto Touch (clic) sobre su jugador.

Figura 15 Pantalla de inicio para seleccionar un jugador

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

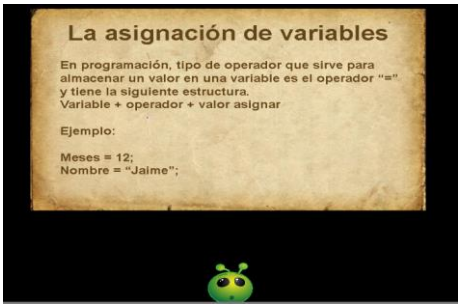
Mensajes de información: estas escenas son las encargadas de mostrar la información adecuada al usuario para que él/ella entienda las mecánicas, reglas y funcionamiento del juego y/o tareas (figura 16).

Figura 16 Escena para mostrar información mediante un personaje instructor

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Información de uso de sentencias: estas escenas presentarán el modo de uso de las sentencias, para que el usuario asimile y razone como se usarán dentro del juego en una determinada tarea (figura 17).

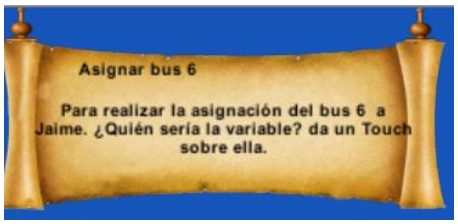
Figura 17 Escenas de información de variables, sentencias y operadores



Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Notas de información de las tareas: en este apartado de las escenas se mostrarán los pasos a seguir para la resolución de una tarea (problema algorítmico) mediante una serie de preguntas que harán razonar al usuario para llegar a la respuesta correcta (figura 18).

Figura 18 Apartado para muestra de instrucciones sobre las tareas



Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

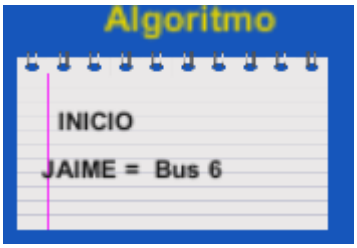
Gestos disponibles: esta sección dentro de la escena muestra al usuario qué gestos puede utilizar para resolver una determinada tarea (figura 19). El usuario deberá saber qué gesto utilizar en base al enunciado presentado en las notas de información.

Figura 19 Gestos disponibles dentro de una escena

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Información del algoritmo: esta sección de la escena es la encargada de mostrar visualmente y en tiempo real la secuencia de construcción del algoritmo que va creando el usuario en base a la resolución de la tarea propuesta; en la figura 20 se puede apreciar un ejemplo de ello.

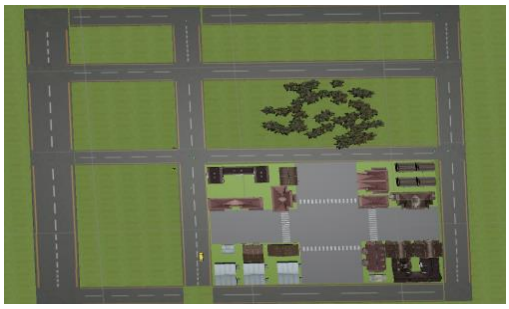
Figura 20 Algoritmo construido por el usuario



Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Ciudad de recorrido en 3D: es la pista donde se realizan distintas operaciones con las sentencias de programación. Se trata de una mini-ciudad en 3 dimensiones conformada por objetos cotidianamente encontrados por los conductores de autobuses, entre los que tenemos semáforos, paradas, personas (pasajeros), etc. En la figura 21 se puede apreciar la ciudad, mientras que en la figura 22 se muestra la perspectiva de la vista del bus dentro de la ciudad. Asimismo, en la figura 23 se aprecia la tarea que debe cumplir el usuario, y en la figura 24 se observa un objeto dentro de la escena, que en este caso es una parada de autobuses.

Figura 21 Vista aérea de la mini-ciudad



Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

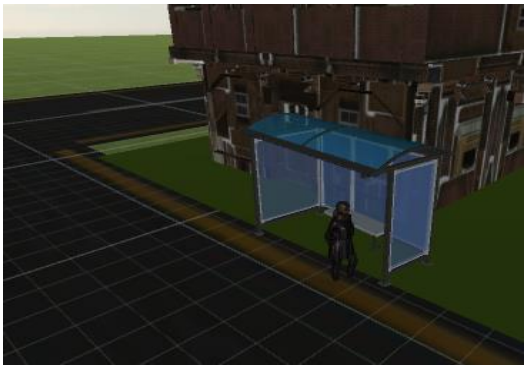
Figura 22 Recorrido del autobús dentro de la escena

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Figura 23 Tareas dentro de la escena

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Figura 24 Personajes mostrando interacción (esperando) dentro de la escena



Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

5.1.4 Estudio de usuarios para la evaluación de la utilidad y aceptabilidad de la propuesta

El análisis de la propuesta se completó con la realización de un estudio de usuarios. Para el efecto se tomaron en cuenta las recomendaciones para este tipo de estudios en Interacción Humano-Computador [18]. Entre estas recomendaciones se pueden citar la realización de pruebas previas, la organización de tareas, el número de participantes, entre otras.

El estudio fue realizado en la ciudad de Quito-Ecuador, en una de las oficinas del grupo de tecnología de KFC. Participaron un total de 16 especialistas en desarrollo de software, 2 mujeres (18.8%) y 14 varones (81.3%), todos con grado académico superior. Tres de los participantes indicaron tener experiencia media acerca del uso del Kinect y sus aplicaciones, mientras que el resto dijo que tenía experiencia nula o básica.

Procedimientos y tareas

Los participantes tuvieron que realizar 3 tareas específicas (T1, T2 y T3), las cuales incluyeron ejercicios relacionados con el uso de la sentencia IF, y las estructuras de repetición. Las tareas consistieron en resolver un problema propuesto por el prototipo, con el objetivo de observar la forma de interactuar entre los participantes y el software, resolviendo tareas respecto a sentencias de programación como asignaciones, condicionales, ciclos.

Para cada participante, se inició explicando el protocolo, objetivos y tareas por igual. Todos los participantes recibieron exactamente las mismas instrucciones, pero el orden de ejecución de las tareas fue aleatorio. La recolección de la información del estudio se realizó posterior a la sesión de tareas utilizando un cuestionario. El cuestionario fue elaborado en base a otro cuestionario utilizado en un estudio similar (ver [14]). La tabla 11 muestra los resultados correspondientes a las percepciones de los participantes obtenidas a partir del cuestionario luego de usar el software.

Tabla 11. Resultados preguntas cerradas

Ítem	Media (SD)
1.- Facilidad para realizar las tareas (1 = muy fácil; 5 = muy difícil)	2,9 (0,8)
2.- Esfuerzo mental al realizar las tareas (1 = muy alto; 5 = muy bajo)	2,9 (0,7)
3.- Esfuerzo físico empleado para realizar las tareas (1 = muy alto; 5 = muy bajo)	3,6 (0,9)
4.- Exactitud haciendo las tareas (1 = muy alto; 5 = muy bajo)	2,3 (0,5)
5.- Velocidad al hacer las tareas (1 = muy lento; 5 = muy rápido)	3,3 (0,7)

6.- Fatiga en la mano (1 = ninguna; 5 = muy alta)	1,9 (1,1)
7.- Fatiga en el brazo (1 = ninguna; 5 = muy alta)	2,0 (0,7)
8.- Fatiga en el hombro (1 = ninguna; 5 = muy alta)	1,4 (0,8)
9.- Fatiga en el cuello (1 = ninguna; 5 = muy alta)	1,5 (0,9)
10.- Confort general (1 = muy incomodas ; 5 = muy cómodas)	3,6 (0,8)
11.- Dificultad general de las tareas (1 = muy difícil; 5 = muy fácil)	3,1 (0,4)

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Los resultados de las percepciones determinados en la tabla 11 se refieren a la operación física, mental, comodidad, confort, precisión, dificultad para hacer las interacciones. Los participantes indicaron que al momento de realizar las tareas tuvieron leves fatigas, teniendo como más alta la fatiga en la mano, la cual corresponde a un nivel relativamente bajo. Del mismo modo indicaron que la fatiga mental y física fue moderada.

Además, dentro del estudio también se incluyeron preguntas abiertas, con las que se obtuvieron recomendaciones para el uso del software, aceptación y sus oportunidades de mejora, tal como se menciona a continuación:

En cuanto a las **recomendaciones para el uso del software**, los participantes indicaron que podrían ser más de un jugador, pensar en personas con discapacidad y verificación respecto a que el dispositivo de captura de gestos (Kinect) esté conectado correctamente dentro del software, como los comentarios destacados.

Por otra parte, se preguntó lo que **les gustó y no del software**. Los participantes destacaron como atractiva la propuesta, haciendo énfasis en la forma de interacción y aprendizaje mediante gestos con las manos. Por el contrario, los participantes dijeron que no les gustó en primera instancia la precisión del seguimiento del cursor con la mano, indicando falta de precisión y en otros aspectos que no era inclusivo. Esto, no obstante, es un aspecto que se espera sea superado a futuro conforme evolucionen los dispositivos de adquisición de gestos.

También se indagó sobre las **oportunidades de mejora para el prototipo del software.** Los participantes dieron a conocer sus opiniones como la posibilidad de hacer las tareas sentados, otorgar medallas posteriores de cumplir correctamente una tarea, la posibilidad de interactuar con dos usuarios activos, control por voz, ser más inclusivo como las opiniones destacadas, entre otras.

En base a los resultados expuestos se determinó que la propuesta de apoyar el aprendizaje la lógica de programación mediante un software que use el paradigma de la interacción sin contacto basada en gestos con las manos, tiene una acogida favorable.

CAPÍTULO VI

CONCLUSIONES Y RECOMENDACIONES

6.1 Conclusiones

En base a los resultados obtenidos se procede a concluir lo siguiente:

- Mediante la conducción de un grupo de enfoque se determinaron las características a implementar en el software. Este grupo de enfoque sirvió como base para la obtención de ideas y acciones incluidas en el prototipo. Posteriormente el análisis de estas ideas, junto con la revisión de referencias bibliográficas, permitió determinar que tecnologías y características estarían consideradas para ser incluidas dentro del prototipo del software.
- Al identificar y comparar entre sí las tecnologías existentes para la creación de videojuegos más usadas por los desarrolladores de videojuegos en la actualidad, se determinó que Unity 3D era la herramienta adecuada para el desarrollo del prototipo, tomando como base sus características favorables respecto a los demás. Por otra parte, MS-Kinect es el sensor que contiene las características convenientes para el trabajo propuesto, especialmente para integrarlo con Unity.
- En base a las características determinadas y al análisis de tecnología idónea, se desarrolló un software que hace uso del paradigma de interacción sin contacto para apoyar el aprendizaje de programación. Este software consta de una interfaz de usuario basada en gestos manuales, y combina el uso de técnicas y mecanismos usados en los juegos. El juego utiliza acontecimientos de la vida cotidiana y los relaciona con las sentencias de programación. De esta manera se busca apoyar el aprendizaje de programación.
- Haciendo uso del software desarrollado se llevó a cabo un estudio de usuarios con miras a evaluar la acogida de la propuesta. Un total de 16 participantes con conocimientos avanzados, que ejercen su profesión como desarrolladores profesionales colaboraron voluntariamente en la evaluación del prototipo. Luego de analizar los resultados obtenidos en base al análisis de la percepción de los participantes en cuanto a operaciones mentales, físicas, fatiga, entre otras, se concluyó que la propuesta tiene una aceptación favorable. Por lo tanto, el uso de aplicaciones como la propuesta puede constituir una herramienta de apoyo para mejorar el aprendizaje de la lógica de programación.

6.2 Recomendaciones

En base a los resultados obtenidos, y de acuerdo a las conclusiones antes expuestas, se puede realizar las siguientes recomendaciones.

- Considerar el uso de otros dispositivos compatibles con Unity 3D, que permitan una mayor precisión en cuanto al reconocimiento de los gestos realizados por el usuario. Esto será posible especialmente cuando tales dispositivos estén disponibles en el mercado.
- El análisis de las características incluidas en el prototipo puede ser mejorado. Para ello, la extracción de las características correspondientes puede realizarse con otros tipos de usuarios. Algunas opciones son estudiantes que hayan desertado en la asignatura de programación o que la hayan reprobado.
- Utilizar la versión profesional de Unity que proporciona mayor facilidad de usar e integrar componentes y objetos dentro del juego. Gracias a la mejora de compatibilidad que trae consigo, permite una mejor adaptación de trabajo con el sensor Kinect V2.0 y/o el uso de Leap Motion para aumentar la experiencia y características que puede ofrecer el prototipo del software.

CAPÍTULO VII

BIBLIOGRAFÍA

BIBLIOGRAFIA

[1] Code.org, «hourofcode,» 2015. [En línea]. Available: https://hourofcode.com/es.

[2] L. Kindergarten, «scratch,» [En línea]. Available: https://scratch.mit.edu/.

[3] E. Escudero, «Proceso de enseñanza-aprendizaje de los fundamentos,» Real, 13/07/2012.

[4] V. M. Pijuán, Por qué parece tan difícil programar, Diciembre de 2007.

[5] B. A. S. Moxo, Applying ludic educative software and microworlds to facilitate the teaching learning, Oaxaca-México, 2012.

[6] F. LLorens, F. J., C. J, V. Arnedo y P. Compañ, Gramificación del proceso de aprendisaje, 2016.

[7] B. A. S. Moxo, Applying ludic educative software and microworlds to facilitate the teaching-learning process, Cañada: 1, 2012.

[8] P. Santana, Interfaces Naturales de Usuario, La Experiencia de la Universidad de Colima, Colima.

[9] C. A. Z. Ospina, Fundamentos de programación,Guía de autoenseñanza, Caldas: RA-MA, 2006.

[10] L. Joyanes, FUNDAMENTOS DE PROGRAMACION, Madrid, 2008.

[11] J. A. F. Valls, Nuevas Técnicas de Interacción Basada en Movimiento Aplicadas a Procesos de Rehabilitación, Castilla-La Mancha, 2015.

[12] I. López, «JUEGOS SERIOS EN EL APRENDIZAJE,» 2015.

[13] M. Eguillor, «wonnova,» 2017.

[14] O. Erazo, N. Baloian, J. A. Pino y S. F. Ochoa, «Designing Hand Gesture Interfaces for Easing Students Participation From their Spot,» *Proceedings of IEEE 21st International Conference on Computer Supported Cooperative Work in Design (CSCWD),* pp. 133-138, 2017.

[15] O. Erazo y R. Pico, «Interfaces de usuario basadas en gestos manuales sin contacto para la sala de clases,» Quevedo, 2014.

[16] F. Llorens, F. Gallego, C. Villagrá y P. Rosique, «Gamificación del Proceso de Aprendizaje,» Alicante, e, 2016.

[17] A. Gibbs, «Focus groups,» *Social research update,* vol. 19, nº 8, pp. 1-8, 1997.

[18] S. McKenzie, Human-Computer Interaction: An Empirical Research Perspective, Amsterdam, 2013.

CAPÍTULO VIII

ANEXOS

7.1 Diccionario de datos

Tabla 12. Diccionario de datos- tabla Usuario

ENTIDAD	Usuario			
NOMBRE	**TIPO**	**LONGITUD**	**DESCRIPCION**	**RESTRIC**
IDUsuario	ENTERO		CLAVE PRIMARIA	NO NULLO
descripcion	TEXTO	100		NO NULLO
ruta	TEXTO	100		NO NULLO
IDEstado	ENTERO			NO NULLO

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 13. Diccionario de datos- tabla Partida

ENTIDAD	Partida			
NOMBRE	**TIPO**	**LONGITUD**	**DESCRIPCION**	**RESTRIC**
IDPartida	ENTERO		CLAVE PRIMARIA	NO NULLO
IDUsuario	TEXTO	100	CLAVE SECUNDARIA	NO NULLO
descripcion	TEXTO	100		NO NULLO
fecha	FECHA			NO NULLO
IDEstado	ENTERO			NO NULLO

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 14. Diccionario de datos- tabla Escena

ENTIDAD	Escena			
NOMBRE	**TIPO**	**LONGITUD**	**DESCRIPCION**	**RESTRIC**
IDEscena	ENTERO		CLAVE PRIMARIA	NO NULLO
escenaDescripcion	TEXTO	100	CLAVE SECUNDARIA	NO NULLO
descriprutacion	TEXTO	100		NO NULLO
orden	FECHA			NO NULLO
IDEstado	ENTERO			NO NULLO

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 15. Diccionario de datos- tabla DetallePartida

ENTIDAD	DetallePartida			
NOMBRE	**TIPO**	**LONGITUD**	**DESCRIPCION**	**RESTRIC**
IDDetallePartida	ENTERO		CLAVE PRIMARIA	NO NULLO
IDPartida	ENTERO		CLAVE SECUNDARIA	NO NULLO
IDEscena	ENTERO		CLAVE SECUNDARIA	NO NULLO
efectividad	DECIMAL			NO NULLO
IDEstado	ENTERO			NO NULLO

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 16. Diccionario de datos- tabla Modulo

ENTIDAD	Modulo			
NOMBRE	**TIPO**	**LONGITUD**	**DESCRIPCION**	**RESTRIC**
IDmodulo	ENTERO		CLAVE PRIMARIA	NO NULLO
numero	ENTERO			NO NULLO

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 17. Diccionario de datos- tabla Estado

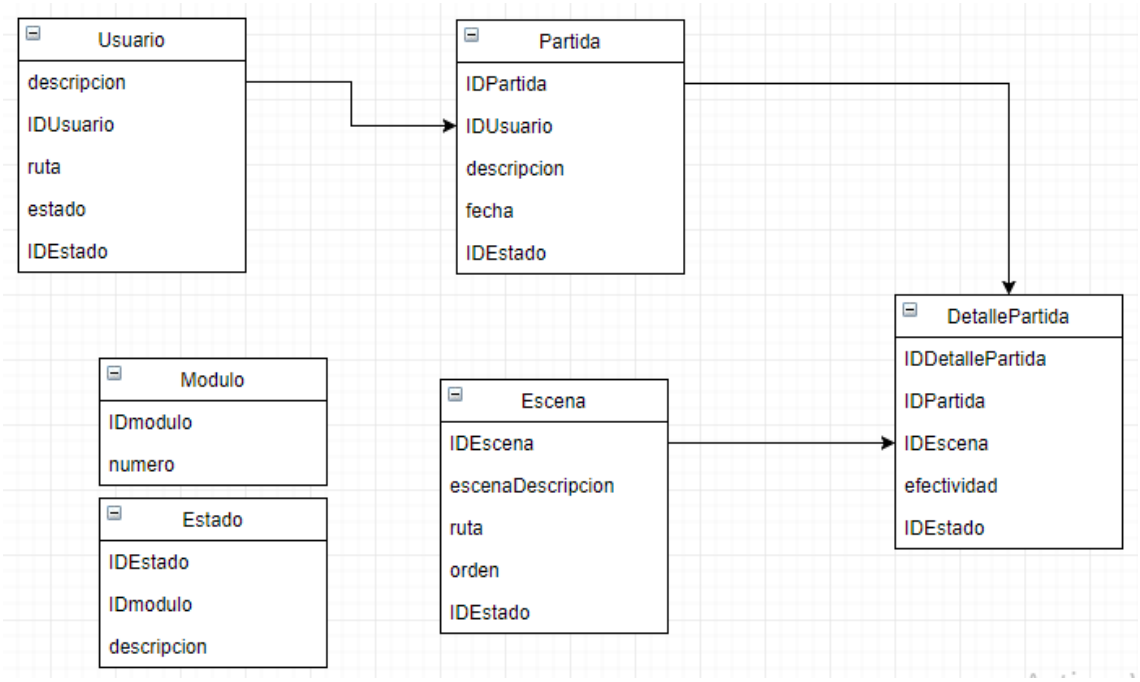
ENTIDAD	Estado			
NOMBRE	**TIPO**	**LONGITUD**	**DESCRIPCION**	**RESTRIC**
IDEstado	ENTERO		CLAVE PRIMARIA	NO NULLO
IDmodulo	ENTERO			NO NULLO
descripcion	TEXTO			NO NULLO

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

7.2 Modelo físico de la base de datos

Anexo 1

Figura 25 Diagrama de Base de datos

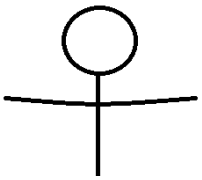
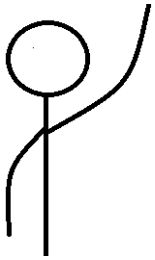
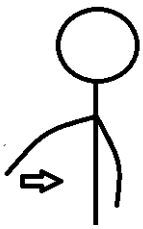


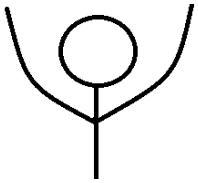
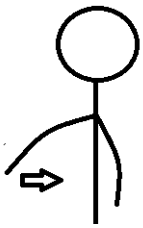
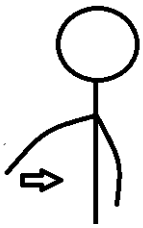
Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

7.3 Gestos usados dentro del software

Anexo 2

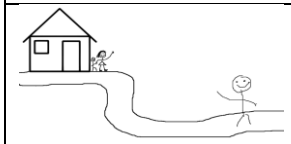
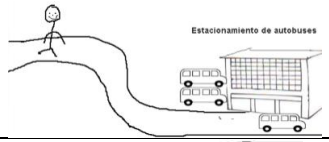
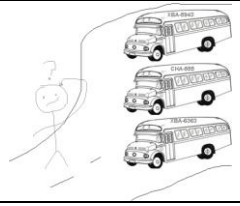
Tabla 18. Tabla de gestos

GESTO		**Representación**	**Modo de uso**
		If	El usuario coloca ambas manos horizontalmente a la altura del hombro por 1.5 segundos.
Inicio 	**Fin** 	**CICLO** While	Fase 1: el usuario levanta su mano izquierda sobre su cabeza por 1.5 segundos. Fase 2: el usuario desliza su mano a la altura del estómago de derecha a izquierda.

inicio 	**Fin** 	**CICLO** For	Fase 1: el usuario levanta ambas manos sobre su cabeza por 1.5 segundos. Fase 2: el usuario desliza su mano a la altura del estómago de derecha a izquierda.
inicio 	**Fin** 	**CICLO** While	Fase 1: el usuario levanta su mano derecha la altura del hombro por 1.5 segundos. Fase 2: el usuario desliza su mano a la altura del estimado de derecha a izquierda.
		Panel de opciones	EL usuario con su mano izquierda o derecha la agita de un lado a otro por 1.2 segundos

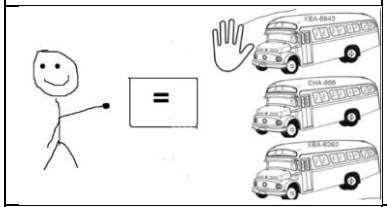
Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 19. Storyboard de presentación inicial

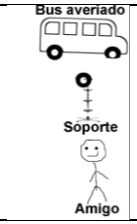
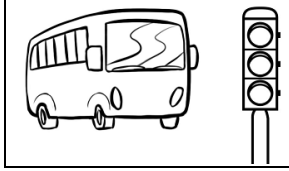
STORYBOARD - SENTENCIA BÁSICA: Ninguna				
Escena 1: INTRODUCCIÓN PARA EL APREDIZAJE DE LA ASIGNACIÓN Y COMPARACIÓN DE VALORES A UNA VARIABLE				
Escenario	Descripción	¿Cómo el usuario actúa en lugar de la sentencia?	Gesto que representa la sentencia	¿Cómo se pretende desarrollar la lógica de programación del usuario con respecto a la sentencia?
	El conductor sale de su casa camino a su trabajo, despidiéndose de su familia.	Ninguna solo visual	Ninguna solo visual	Ninguna solo visual
	El conductor llega al estacionamiento de autobuses. Recibe la asignación de un autobús para su día de labores en la empresa.	Ninguna solo visual	Ninguna solo visual	Ninguna solo visual
	El conductor no puede confirmar la placa del bus asignado debido a que éstas solo puedan verse desde el aire porque está en el techo del bus. La asignación del bus es aleatoria cada día. ayuda al conductor asignándole el autobús con la placa asignada	Ninguna solo visual	Ninguna solo visual	Ninguna solo visual

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 20. Storyboard para la escena de operadores básicos

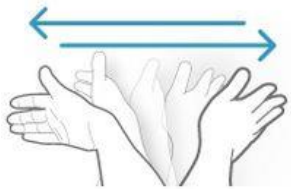
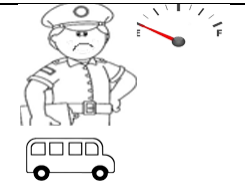
<table>
<tr><th colspan="5">STORYBOARD - SENTENCIA BÁSICA Asignación (=) y comparación (==)</th></tr>
<tr><td colspan="5">Escena 2: APREDIZAJE DE LA ASIGNACIÓN Y COMPARACIÓN DE VALORES A UNA VARIABLE</td></tr>
<tr><td>Escenario</td><td>Descripción</td><td>¿Cómo el usuario actúa en lugar de la sentencia?</td><td>Gesto que representa la sentencia</td><td>¿Cómo se pretende desarrollar la lógica de programación del usuario con respecto a la sentencia?</td></tr>
<tr><td>BÁSICO
  </td><td>Asignación (aleatoria) del autobús al conductor, en base a la placa por día. Se presentarán varios buses entre los cuales uno de ellos llevará la placa correspondiente a la asignación del día para el conductor.</td><td>Para asignarle el bus el usuario deberá hacer el gesto que representa al igual en el momento que visualice el número de placa asignado aleatoriamente.</td><td rowspan="2">Para realizar la asignación (=) el usuario debe usar el gesto que despliega el listado de operadores, mediante el gesto WAVE agitando su mano de izquierda a derecha.
  </td><td rowspan="2">La intención de esta acción es que el USUARIO RAZONE Y COMPRENDA EL ACTO DE COMPARAR INFORMACIÓN

(PlacaAsignada == BusPlaca)
y asignar valores a una variable

(busCONDUCTOR=busPLACA).
CÓDIGO: La sentencia IF para este caso estará implícita.
IF (PlacaAsignada == BusPlaca)
{
busCONDUCTOR= busPLACA
}</td></tr>
<tr><td>MEDIO
  </td><td>Asignación (usando un auxiliar) de un nuevo neumático al autobús debido a que éste tiene averiado uno de ellos. El nuevo neumático esta sobre una mesa de soporte. El conductor obtendrá la ayuda de una persona. No se puede dejar un neumático en el suelo.</td><td>Para asignar un neumático el usuario deberá hacer el gesto que representa al igual en el momento necesario para indicar que objeto sostendrá la llanta en ese momento.</td></tr>
<tr><td>AVANZADO
  </td><td>El conductor inicia su día de trabajo, en su recorrido se debe de tener en cuenta que se deberá detener en caso de que un semáforo esté en rojo. En ocasiones los semáforos suelen fallar y apagarse por lo que deberá estar atento al de la calle de alado.</td><td>Para armar una condición de decisión el usuario deberá hacer el gesto IF en el momento que se visualice el problema del semáforo.</td><td>Para realizar el condicional (IF) se deberán colocar ambas manos horizontales.
  </td><td>CÓDIGO:
IF (SemaforoEnRojo) {
DetenerAutobus
}</td></tr>
</table>

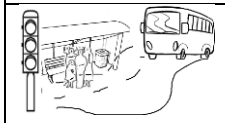
Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 21. Storyboard para las escenas de la estructura de control While

STORYBOARD - SENTENCIA BÁSICA Ciclo / While				
Escena 3: INTRODUCCIÓN PARA EL APRENDIZAJE DE ESTRUCTURAS DE CONTROL O REPETICIÓN CON CONDICIONES DE ITERACIÓN.				
Escenario	Descripción	¿Cómo el usuario actúa en lugar de la sentencia?	Gesto que representa la sentencia	¿Cómo se pretende desarrollar la lógica de programación del usuario con respecto a la sentencia?
BÁSICO 	El conductor debe de verificar si el autobús con el que trabajará tiene la gasolina suficiente para su recorrido cotidiano. Se conoce que el ticket recibido determina cuantos litros son suficientes para llenar el tanque de gasolina, ayuda al conductor determinando cuantos litros de gasolina tendrá que utilizar.	Para determinar cuántos litros de combustible y de qué tipo se destinaran al autobús el usuario deberá hacer el gesto que representen las estructuras de control de repetición y decisión en el momento que se presente la oportunidad de hacer uso de ellas.	Mano izquierda arriba indica que se usara un ciclo de repetir mientras (While)  Agrega una condición con ambas manos horizontales 	La intención de ésta acción es que el usuario razone y comprenda el uso de las estructuras repetitivas como lo es el caso de repetir mientras. Y determine cuando la acción de repetir terminará. (Se cumpla condición de parada) CÓDIGO: While (TanqueNoEsteLleno) { Tanque = agregarLitro
MEDIO 	El problema del autobús y la gasolina se complica. El conductor recibió un ticket en el cual no está determinado cuantos litros de gasolina son necesarios para llenar el tanque. Ayuda al conductor a verificar que se llene el tanque sin que le sobre ni le falte gasolina.			
AVANZADO 	El conductor tiene una incertidumbre, esto debido a que tiene que llenar el tanque de gasolina del autobús que se le asigno, pero no conoce cuántos litros de gasolina son necesarios para llenarlos y tampoco qué tipo de combustible usa (súper, extra, Diésel). Ayuda al conductor a solucionar este inconveniente.			

				}

STORYBOARD - ESTRUCTURA REPETITIVA: FOR				
Escena 4: INTRODUCCIÓN PARA EL APRENDIZAJE DE ESTRUCTURAS DE CONTROL O REPETICIÓN CON CONDICIONES DE ITERACIÓN.				
Escenario	Descripción	¿Cómo el usuario actúa en lugar de la sentencia?	Gesto que representa la sentencia	¿Cómo se pretende desarrollar la lógica de programación del usuario con respecto a la sentencia?
BÁSICO 	El conductor del autobús deberá realizar 4 de recorridos, deteniéndose solo si se encuentra con un semáforo en rojo.	Para realizar las vueltas del recorrido el usuario deberá identificar y hacer el gesto correspondiente al bucle FOR, debido a que se conoce con exactitud el número de ciclos que realizará.	Levantar la mano izquierda sobre la cabeza especifica el bucle FOR para posterior a ello realizar el gesto de deslizar hacia la izquierda donde cada giro será un ciclo. 	La intención de esta acción es que el usuario razone e identifique que bucle de repetición es adecuado para ciertas tareas específicas, donde comprenderá la relación de que cuando se conoce los ciclos a iterar de inicio a fin debe usar una sentencia con esa estructura como lo es el FOR CÓDIGO : FOR (Inicio = 1 hasta 3) { IF (Semanoforo == ROJO) { DetenerBus }}
MEDIO 	El conductor del autobús deberá realizar 4 vueltas de recorrido, deteniéndose en las paradas en las cuales existan más de 5 personas.			Dentro de esta acción el usuario deberá razonar y usar operadores relacionales para cumplir con la tarea propuesta. CÓDIGO: FOR (Inicio = 1 hasta 4) { IF (CantidadPersonasParada > 5) { DetenerBus }}

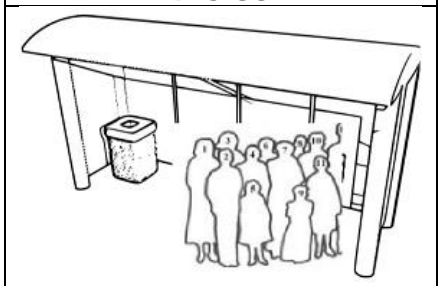
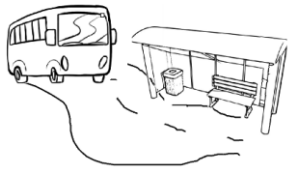
AVANZADO
 
	El conductor del autobús realizará 3 vueltas, se detendrá en semáforos en rojo y paradas donde hayan más de 3 mujeres			FOR (Inicio = 1 hasta 3)
				{
				IF (CantidadMujeresEnParada > 3)
				{ DetenerBus }
				}

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 22. Storyboard para la escena que muestra tareas de la estructura de control For

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

Tabla 23. Storyboard para la escena que muestra tareas de la estructura de control DO WHILE

<table>
<tr><th colspan="5">STORYBOARD – ESTRUCTURA DE REPETICION HACER MIENTTRAS (DO WHILE)</th></tr>
<tr><td colspan="5">Escena 5: INTRODUCCIÓN PARA EL APRENDIZAJE DE ESTRUCTURAS DE CONTROL O REPETICIÓN CON CONDICIONES DE ITERACIÓN.</td></tr>
<tr><th>ESCENARIO</th><th>DESCRIPCIÓN</th><th>¿CÓMO EL USUARIO ACTUA EN LUGAR DE LA SENTENCIA?</th><th>SEÑA QUE REPRESENTA LA SENTENCIA</th><th>¿CÓMO SE PRETENDE DESARROLLAR LA LOGICA DE PROGRAMACIÓN DEL USUARIO CON RESPECTO A LA SENTENCIA?</th></tr>
<tr><td>BÁSICO
  </td><td>El conductor se detiene en una parada de autobuses, deberá verificar que ingresen todas las personas mientras que estas suben una por una.</td><td>Para realizar la acción repetitiva, el usuario deberá hacer un gesto que representa al do While (hacer mientras) en el momento que se detenga en una parada y se encuentre con esta tarea específica.</td><td rowspan="3">Levantar la mano izquierda a la altura del hombro (horizontal) especifica el bucle Do While para posterior a ello realizar el gesto de deslizar hacia la izquierda donde cada giro será un ciclo.
  </td><td>La intención de esta acción es para que el usuario razone y comprenda el uso adecuado de esta sentencia la cual mantiene la lógica de hacer una estructura repetitiva y comprobar su condición de finalizado después de realizar por lo menos una acción.
CÓDIGO:
DO {
SubirPasajeroAlBus
} (AunQuedaOtraPersona == TRUE)</td></tr>
<tr><td>MEDIO
  </td><td>El conductor por ser nuevo no conoce bien la ciudad, deberá recorrer el ciclo de su ruta asignada mientras que hallan paradas de buses con personas, en caso de que no haya al menos una no realizará otro recorrido.</td><td>El usuario en esta acción deberá hacer el gesto do While (hacer mientras) en el momento que inicie el recorrido y se presente con esta tarea.</td><td>CÓDIGO:
DO {
ConducirAutobus

} HayParadas == TRUE)</td></tr>
<tr><td>AVANZADO
  </td><td>El conductor deberá realizar el recorrido del bus por la ciudad hasta que complete el número máximo de pasajeros.</td><td>En esta determinada acción el usuario deberá hacer el gesto do While (hacer mientras) en el momento que inicie el recorrido para que solo finalice al terminar su condición.</td><td>CÓDIGO:
DO {
ConducirAutobus

} (CompletoNumeroPasajeros == TRUE)</td></tr>
</table>

Fuente: Conocimiento propio

Elaborado por: Castro Mychael (2017)

7.4 Protocolo para la realización del grupo de enfoque de requerimientos del sistema

PROGRAMACIÓN DEL GRUPO DE ENFOQUE

A continuación, se detallan los 6 puntos a seguir durante el grupo de enfoque.

1.- Presentación

a) Presentación del moderador.- Para que los participantes que asistan al grupo de enfoque se sientan tranquilos y en confianza de realizar y/o formular preguntas o dudas.
b) Explicar el motivo de la reunión.
c) Presentar los participantes.- Una presentación rápida para que se conozcan entre todos y poder llamarlos por sus nombres.

2.- Explicación introductoria

En este punto se explicará a los participantes el objetivo de la reunión y que se los grabará en audio y video. También se les dirá que se sientan cómodos porque eso ayudará a obtener buenas respuestas de ellos, que en ningún caso serán erróneas. Además, se detallarán las reglas que se deban de seguir lo cual incluirá:

- Que hable 1 persona a la vez.
- Levantar la mano para indicar que quiere hablar.

3.- Presentación de una demostración

Se mostrará una demostración con la ayuda de un video y usando gestos con la ayuda del Kinect y una pequeña aplicación en Unity 3D.

4.- Preguntas

Con la finalidad de conocer qué gestos y escenario se consideran adecuados para representar condiciones, asignaciones y flujos a seguir en programación.

Tabla 24. Preguntas objetivas para la obtención de requerimientos de los participantes del grupo de enfoque

OBJETIVO	PREGUNTAS
1.- Determinar el nivel de dificultad en el aprendizaje de las sentencias básicas de programación	1.- ¿Cuáles son las sentencias que resultan complicadas de aprender en programación? 2.- ¿Indique del 1 al 5 siendo el 5 el grado más difícil que tan complicado es el aprendizaje de esas sentencias? 3.- ¿Por qué resultan ser complicadas? 4.- ¿Qué tiempo les tomo dominarlas?
2. Determinar un escenario donde se pueda representar las sentencias básicas	1.- ¿Qué escenario consideran adecuado para aprender programación? 2.- ¿Por qué es adecuado el escenario? 3.- ¿Qué facilidades brinda? 4.- ¿Qué objetos ofrece el escenario que se asemejen con sentencias de programación? 5.- ¿Por qué es el mejor escenario?
3- Vincular las sentencias con objetos de similar comportamiento que se encuentren dentro de la escena.	1.- ¿Qué similitudes hay entre el comportamiento de los objetos de la escena y las sentencias de programación? 2.- ¿Qué ejemplo práctico podría citar? 3.- ¿Es este ejemplo el mejor? 4.- ¿Facilitaría el ejemplo la comprensión?
4.- Obtener la representación de gestos que mejor asemejan a cada sentencia básica.	1.- ¿Qué gesto podría realizar para representar cada sentencia? 2.- ¿Es el gesto indicado? 3.- ¿Por qué es el mejor gesto?

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

5.- Cuestionario demográfico para los participantes

Se les entregó a los participantes el cuestionario demográfico para que brinden su ayuda llenándolo.

Tabla 25. Encuesta demográfica para los participantes del grupo de enfoque

Edad:	
Sexo:	☐ Masculino ☐ Femenino ☐ Otros
Ocupación:	
Semestre/Grado o título:	
Experiencia en programación:	☐ Menos de un año ☐ De 1 a 2 años ☐ Más de 2 años
Experiencia adquirida:	☐ Autodidacta ☐ Laboral ☐ Educativa
Lenguaje de programación que domina:	
Experiencia en aplicaciones basadas en gestos sin contacto.	☐ Básica ☐ Media ☐ Avanzada

Fuente: La investigación
Elaborado por: Castro Mychael (2017)

6.- Agradecimientos por la participación

Agradecer a las personas por participar y por mostrar su calidad en las respuestas obtenidas a partir de las preguntas.

7.4.1 Resultados del cuestionario demográfico del grupo de enfoque para la adquisición de requerimientos del software

Los datos obtenidos en este cuestionario se exponen a continuación:

1. **Edad de las personas**

Cuadro #1

Edad	Cantidad
22	2
23	1
24	4
Total:	7

Un total de 7 personas con una edad promedio de 23.28 años fueron los partícipes del grupo de enfoque, todos hombres debido a que no asistieron mujeres. Los participantes son: egresados, recientemente graduados y otros cursando los distintos semestres carrera Ingeniería en Sistemas y afines como Telemática de la Universidad Técnica Estatal de Quevedo.

2. **Sexo de las personas que asistieron**

Cuadro #2

Sexo	Cantidad
Masculino	7
Femenino	0
Total:	7

Participaron un total de 7 personas, todos hombres no asistieron mujeres.

3. **Ocupación.**

Cuadro #3

Ocupación	Cantidad
Técnico Administrativo	1
Estudiante Sistemas	4
Desarrollos de Sistemas	1
Técnico Administrativo	1
Total:	7

Cada participante especificó su ocupación en la actualidad.

4. **Semestre/ Grado o Titulo**

Cuadro #4

Grado o Titulo	Cantidad
Egresado	2
Ingeniero en Sistemas	1
Noveno semestre	2
Decimo semestre	2
Total:	7

Dentro de los participantes estuvieron 2 egresados, 1 recientemente graduado y el resto se encontraba cursando algún semestre de estudio actualmente.

5. Experiencia en programación.

Cuadro #5

Programación	Cantidad
De 1 a 2 años	0
Más de 2 años	7
Total:	7

Todos los participantes indicaron tener más de dos años de experiencia (estudiantil) en programación de nivel básica e intermedia.

6. Experiencia adquirida.

Cuadro #6

Experiencia adquirida	Cantidad
Autodidacta	3
Laboral	1
Educativa	5

La experiencia de los participantes en su mayoría es educativa, siguiéndole los que han sido autodidactas los cuales han ido adquiriendo su conocimiento mediante videos en YouTube y solo un participante indicó que los conocimientos adquiridos fueron laborales.

7. Experiencia en aplicaciones basadas en gestos sin contacto

Cuadro #7

Experiencia	Cantidad
Básica	7
Media	0
Avanzada	0

Los participantes indicaron conocimientos básicos acerca del dispositivo (Kinect para Xbox-360).

8. Lenguaje de programación que domina

Cuadro #8

Lenguajes	Cantidad
ASSEMBLER	2
C#	4
JAVA	3
PHP	1

Los participantes indicaron su lenguaje de programación preferido, entre los cuales se indicaron aquellos que comúnmente se instruyen en la Universidad Técnica Estatal de Quevedo.

7.5 Protocolo de seguimiento para los participantes del estudio de usuarios

PROTOCOLO A SEGUIR EN EL ESTUDIO DE USUARIOS

A continuación, se detallan los 6 puntos a seguir durante el estudio de usuarios. Los usuarios participantes son profesionales en el área de programación y desarrollo de sistemas.

1.- Presentación

d) Presentación del moderador.- Para que los participantes se sientan tranquilos y en confianza de realizar y/o formular preguntas o dudas.
e) Explicar el motivo de la reunión.

2.- Explicación introductoria

En este punto se explicará a los participantes el objetivo de la reunión y que se los grabará en audio y video. Además, incentivarles a que se sientan cómodos, lo cual ayudará a obtener buenas respuestas de ellos. Además, se detallarán las reglas que se deban de seguir lo cual incluirá:

- Los participantes deberán realizar 3 tareas específicas T1 a T3.
- Las instrucciones serán la mismas para todos los participantes.
- Cada participante realizará las 3 tareas, pasando uno a la vez.

3.- Las tareas

Las tareas que los participantes evaluarán, serán 3 en total, en donde a cada participante las tendrá que realizar en un orden diferente al de los demás compañeros.

Las tareas serán:

a) Determine el operador adecuado para realizar la asignación del neumático de repuesto al autobús que tiene un neumático averiado, usando un amigo el cual sostendrá el neumático. **(ASIGNACION para el intercambio de variables usando auxiliar)**
b) Realizar el gesto adecuado para que el autobús se detenga en el momento que encuentre un semáforo en rojo. **(IF modo sencillo)**
c) Determine el tipo de ciclo y cuantas interacciones serían necesarias para para contar el grupo de personas en una parada de autobús **(Ciclo FOR al conocer el inicio y fin del conteo.)**

4.- Cuestionario de evaluación del producto

Por favor, valore de 1 a 5 (siendo 1= muy difícil, 2= Difícil, 3= Normal, 4 = fácil, 5= Totalmente fácil) los siguientes aspectos:

1.- Facilidad para realizar tareas

◯ 1. Muy difícil
◯ 2. Relativamente difícil
◯ 3. Normal
◯ 4. Relativamente fácil
◯ 4. Muy fácil

2.- Esfuerzo mental al realizar las tareas

◯ 1. Muy alto
◯ 2. Relativamente alto
◯ 3. Normal
◯ 4. Relativamente bajo
◯ 4. Muy bajo

3.- Esfuerzo físico empleado para realizar las tareas

◯ 1. Muy alto
◯ 2. Relativamente alto
◯ 3. Normal
◯ 4. Relativamente bajo
◯ 4. Muy bajo

4.- Exactitud haciendo las tareas

◯ 1. Muy alto
◯ 2. Relativamente alto
◯ 3. Normal
◯ 4. Relativamente bajo
◯ 4. Muy bajo

5.- Velocidad al hacer las tareas

◯ 1. Muy lento
◯ 2. Relativamente lento
◯ 3. Normal
◯ 4. Relativamente rápido
◯ 5. Muy rápido

6.- Fatiga de la mano

◯ 1. Ninguna
◯ 2. Relativamente baja
◯ 3. Normal
◯ 4. Relativamente alta
◯ 5. Muy Alta

7.- Fatiga del brazo

◯ 1. Ninguna
◯ 2. Relativamente baja
◯ 3. Normal
◯ 4. Relativamente alta
◯ 5. Muy Alta

8.- Fatiga del hombro

◯ 1. Ninguna
◯ 2. Relativamente baja
◯ 3. Normal
◯ 4. Relativamente alta

◯ 5. Muy Alta

9.- Fatiga del cuello

◯ 1. Ninguna
◯ 2. Relativamente baja
◯ 3. Normal
◯ 4. Relativamente alta
◯ 5. Muy Alta

10.- Confort general

◯ 1. Muy incomodas
◯ 2. Relativamente incomodas
◯ 3. Normal
◯ 4. Relativamente cómodas
◯ 5. Muy cómodas

11.- Dificultad general de las tareas

◯ 1. Muy difícil
◯ 2. Relativamente difícil
◯ 3. Normal
◯ 4. Relativamente fácil
◯ 5. Muy fácil

5.- Cuestionario demográfico realizado a los participantes

Tabla 26. Encuesta demográfica realizada a los participantes del estudio de usuarios

Edad:	
Sexo:	☐ Masculino ☐ Femenino ☐ Otros
Ocupación/Cargo:	
Nivel de educación	☐ Bachillerato ☐ Universitaria-Licenciatura ☐ Universitaria-Ingeniería

	☐ Universitaria-Master ☐ Educación Superior - Doctorado o superior
Nivel de experiencia	☐ Junior ☐ Semi Senior ☐ Senior ☐ Arquitecto de software
Experiencia en programación	☐ Menos de 1 año ☐ De 1 a 3años ☐ Más de 3 años
Experiencia adquirida	☐ Autodidacta ☐ Laboral ☐ Educativa
Lenguaje de programación que domina	
Experiencia en aplicaciones basadas en gestos sin contacto	☐ Básica ☐ Media ☐ Avanzada

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)

6.- Agradecimientos por la participación

Agradecimiento a las personas participantes, por sus comentarios.

Figura 26 Ejemplos de participantes del estudio de usuarios (programadores nivel avanzado grupo KFC Quito-Ecuador)

Fuente: Conocimiento propio
Elaborado por: Castro Mychael (2017)