



Universidad de las Ciencias Informáticas

Facultad 4

**Título: Desarrollo de componentes para la
personalización de un simulador de
apendicectomía autoadaptativo basado en
Realidad Virtual**

**Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas**

Autor:

Dionis Pentón García

Tutor:

Dr. C. Omar Correa Madrigal

Ing. Alieski Véliz Vega

La Habana, septiembre 2022



“La realidad virtual nos va a permitir entrar y navegar dentro de la imagen. Antes la imagen servía para transformar el mundo; ahora la imagen virtual es el mundo.”

Philippe Quéau

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis que tiene por título: “Desarrollo de componentes para la personalización de un simulador de apendicectomía autoadaptativo basado en Realidad Virtual” y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los 16 días del mes de noviembre del año 2022.



Dionis Pentón García



Ing. Alieski Véliz Vega



Dr. C. Omar Correa Madrigal

DATOS DE CONTACTO

Nombre y apellidos del tutor: Omar Correa Madrigal

Institución: Universidad de las Ciencias Informáticas

Título: Doctor en ciencias

Correo electrónico: ocorrea@uci.cu

Nombre y apellidos del tutor: Alieski Véliz Vega

Institución: Universidad de las Ciencias Informáticas

Título: Ingeniero en ciencias informáticas

Correo electrónico: alieskivelizvega@gmail.com

AGRADECIMIENTOS

La realización de este trabajo se agradece a todas las personas que me han apoyado en mi vida académica como cotidiana. Es muy difícil poder mencionar a todas aquellas personas que me han ayudado en la realización de este proyecto así que si me falta alguno pido disculpas. Entre los primeros que debería mencionar están los tutores y profesores que de alguna forma u otra han influido en este por lo que empiezo por la persona que me envió este tema Alieski Véliz Vega que obviamente sin él no hubiese podido entrar en este proyecto donde me pude enfrentar a algo similar a un desarrollo de algo útil para la vida de las personas que se dedican, valga la redundancia, a salvar vidas; el siguiente es Omar Correa Madrigal que dada la situación en la que me encontraba pudo dedicar parte de su tiempo a asesorarme con este tema que a mi parecer estaba bastante complicado y siempre aportando ideas para la finalización de la aplicación. El documento siempre ha sido mi punto débil al igual que la exposición, en estos casos tengo q agradecer a las profesoras Yadira Ramírez y Yorgelis por brindarme consejos sobre estos temas. Por otra parte, otras personas han estado presentes en el desarrollo completo de este tema de tesis como mis compañeros del apartamento que siempre tratamos de ayudarnos entre todos: Kevin, Jorge, Dairon, José Javier, Alejandro, Fernando y Jonathan. Además de ellos se encuentra mi novia Samira de las Mercedes Enriquez González que siempre ha estado impulsándome para que terminara la tesis, aunque me haya enfadado con ella varias veces yo sé que siempre el objetivo ha sido ayudarme y esa es una de las razones por la que la amo con todo mi corazón. Y por último y no menos importante quisiera agradecer a toda mi familia que siempre me ayudaban en lo que podían a pesar de no saber absolutamente nada de mi tema exceptuando a mi tía Lilian Marta Pentón García que fue uno de los referentes para yo coger este tipo de carrera. Gracias a todos ellos hoy puedo decir que he terminado mi tesis, de corazón.

DEDICATORIA

La dedicatoria va principalmente a mi familia específicamente a mi madre Tania Pentón García y a la persona que considero como un padre a pesar de no serlo Pedro M. González García que siempre han tratado de que me supere en la vida desde que era un niño para, con orgullo, poderles decir que gracias a ellos he podido llegar aquí. También se lo dedico a mi novia Samira para poder decirle que ya puedo jugar dota2 tranquilo sin ninguna presión jaja.

Resumen:

La Realidad Virtual (RV) proporciona la capacidad de entrenar a las personas para tratar con situaciones complejas sumergiéndolos en un entorno virtual. Las tendencias actuales han producido aplicaciones muy innovadoras de RV en sectores como la educación, el turismo, la medicina, el deporte y la industria militar. Especialmente con fines educativos, la RV genera experiencias que promueven el aprendizaje de forma más innovadora. A través de estas experiencias, aumenta la motivación y compromiso con las actividades, dejando una impresión que favorece la memorización del contenido. La formación basada en RV promueve la adquisición de habilidades en un entorno seguro con menos índices de riesgos. Los entrenamientos pueden repetirse hasta que el estudiante se encuentre preparado para realizar los procedimientos en situaciones reales. Un problema común con los sistemas de RV para la formación es que los estudiantes experimentan las mismas rutinas de entrenamiento, ya que estos sistemas no están personalizados para patrones de aprendizaje individuales. A partir de esta problemática se plantea la necesidad de desarrollar componentes para la personalización de un simulador autoadaptativo para independizarlo del contexto de aplicación. La formación médica, especialmente en cirugía ha sido un área que ha explorado las ventajas de la simulación. La necesidad de la simulación quirúrgica parte de la seguridad y repetitividad de los procedimientos y sus ventajas sobre el paradigma tradicional de entrenamiento.

Palabras claves: *cirugía; formación; medicina; realidad virtual; simulador.*

Abstract

Virtual Reality (VR) provides the ability to train people to deal with complex situations by immersing them in a virtual environment. Current trends have produced highly innovative applications of VR in sectors such as education, tourism, medicine, sports and the military industry. Especially for educational purposes, VR generates experiences that promote learning in more innovative ways. Through these experiences, motivation and engagement with the activities increase, leaving an impression that favors memorization of the content. VR-based training promotes the acquisition of skills in a safe environment with lower risk rates. Training can be repeated until the trainee is prepared to perform the procedures in real-life situations. A common problem with VR systems for training is that students experience the same training routines, as these systems are not customized for individual learning patterns. From this problem arises the need to develop components to a self-adaptive simulator to make it independent of the application context. Medical training, especially in surgery, has been an area that has explored the advantages of simulation. The need for surgical simulation is based on the safety and repeatability of the procedures and its advantages over the traditional training paradigm.

Keywords: *medicine; simulator; surgery; training; virtual reality.*

Índice de Contenido

RESUMEN:	VIII
ABSTRACT	IX
ÍNDICE DE CONTENIDO	X
ÍNDICE DE FIGURAS	XII
ÍNDICE DE TABLAS	XIII
INTRODUCCIÓN:.....	1
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA	7
1.1 Conceptos Fundamentales	7
1.2 Simulador de realidad virtual en la medicina	8
1.3 Aprendizaje adaptativo basado en realidad virtual.....	10
1.4 Simulador de apendicetomía basado en realidad virtual con un enfoque centrado en el usuario.	15
1.5 Lenguaje de programación	17
1.6 Herramientas a utilizar	20
1.7 Metodologías de desarrollo	24
CONCLUSIONES PARCIALES:	27
CAPÍTULO 2: DISEÑO DE LA PROPUESTA SOLUCIÓN.....	28
2.1 Descripción de la propuesta de solución	28
2.2 Levantamiento de requisitos.....	30
2.3 Fase de planeación.....	31
2.3.1 Plan de iteraciones	33
2.3.2 Plan de entrega.....	34
2.4 Fase de diseño.....	34
2.4.1 Descripción de la arquitectura	35
2.4.2 Patrones de diseño	36
2.4.3 Descripción de las tarjetas CRC	37
2.4.4 Estándares de Codificación	38
CONCLUSIONES PARCIALES	39
CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBA.....	40

Índice de Contenido

3.1 Fase de codificación	40
3.1.1 Primera iteración.....	40
3.1.2 Segunda iteración	41
3.1.3 Tercera iteración	41
3.2 Fase de prueba	42
3.2.1 Pruebas de Aceptación	42
3.3 Resultados del proceso de pruebas	43
CONCLUSIONES PARCIALES	46
CONCLUSIONES GENERALES	48
RECOMENDACIONES	49
GLOSARIO DE TÉRMINOS	50
REFERENCIAS BIBLIOGRÁFICAS	51
ANEXO 1: HISTORIAS DE USUARIO	56
ANEXO 2: TARJETAS CRC	57
ANEXO 3: TAREAS DE INGENIERÍA	59
ANEXO 4: PRUEBAS DE ACEPTACIÓN	60

Índice de Figuras

Figura 1: Estructura de interacción de componentes (Vaughan, Gabrys, y Dubey 2018)	10
Figura 2: Enfoque centrado en el usuario. (Véliz Vega, Correa Madrigal, y Kugurakova 2021)	14
Figura 3: Incisión del paciente	17
Figura 4: Escena del Simulador Mein (Sala de cirugía)	22
Figura 5: Interfaz de usuario Canvas/UserPanel (elaboración propia)	23
Figura 6: Metodología XP (León 2022)	26
Figura 7: componentes del simulador de RV (Véliz Vega, Correa Madrigal, y Kugurakova 2021)	30
Figura 8: Arquitectura de software por capas: elaboración propia	35
Figura 9: Iteraciones de pruebas	45

Índice de Tablas

Tabla 1: resume los componentes más destacados para la adaptación de SRV de una muestra de trabajos estudiados (Vega et al., 2021a).	13
Tabla 2: Seleccionar un perfil de usuario	32
Tabla 3: Crear las variables errores y tiempo	33
Tabla 4: Plan de Iteraciones	34
Tabla 5: Plan de entrega	34
Tabla 6: Tarjeta CRC VRTK_SDKManager	37
Tabla 7: Tarjeta CRC VRTK_SnapDropZone	38
Tabla 8: Crear perfil de usuario	41
Tabla 9: TI. Crear variable de error y tiempo	41
Tabla 10: TI. Crear interfaz de usuario	41
Tabla 11: Crear perfil de usuario	43
Tabla 12: Errores detectados	45
Tabla 13: Crear una interfaz de usuario	56
Tabla 14: Envío de Json a servidor externo	56
Tabla 15: Interfaz de puntuación y nivel de aprendizaje del usuario	57
Tabla 16: Tarjeta CRC ToolsCollisionsManager	57
Tabla 17: Tarjeta CRC PersonajeScript	58
Tabla 18: Tarjeta CRC Timer	58
Tabla 19: Tarjeta CRC ExportJsonScript	59
Tabla 20: Tarjeta CRC ImportJsonScript	59
Tabla 21: TI. Envío de Json al servidor externo	59
Tabla 22: TI. Interfaz de puntuación y nivel de aprendizaje	60
Tabla 23: Crear variable de error y tiempo	60
Tabla 24: Crear interfaz de usuario	61
Tabla 25: Envío de Json al servidor externo	61
Tabla 26: Crear interfaz de puntuación y nivel de aprendizaje	62

Introducción:

La necesidad de nuevas formas de enseñar y aprender introduce retos en las nuevas tecnologías para apoyar la obtención de habilidades y conocimientos. La Realidad Virtual (RV) proporciona la capacidad de entrenar a las personas para tratar con situaciones complejas sumergiéndolos en un entorno virtual. Las tendencias actuales han producido aplicaciones muy innovadoras de RV en sectores como la educación, el turismo, la medicina, el deporte y la industria militar. La formación médica, especialmente en cirugía ha sido un área que ha explorado las ventajas de la simulación. La necesidad de la simulación quirúrgica parte de la seguridad y repetitividad de los procedimientos y sus ventajas sobre el paradigma tradicional de entrenamiento. El paradigma tradicional de formación quirúrgica se basa en la realización de procedimientos supervisados por expertos y bajo el principio de observar, ayudar y operar. A pesar de la eficiencia de este modelo, presenta el inconveniente de dependencia del nivel técnico del tutor experto y su capacidad para transferir las competencias hacia el alumno.

Para la formación en cirugía se realizan prácticas que incluye trabajar sobre cadáveres, cuya ventaja es la anatomía correcta, sin embargo tienen un uso único y no hay sangrado; los modelos de plástico presentan un bajo costo y uso repetido, pero con falta de realidad; los maniqués computarizados tienen aplicaciones múltiples, de uso repetido, son muy útiles para rutinas de aprendizaje, en entrenamiento individual o en equipo, con ensayo de eventos serios pero tienen un alto costo con incisiones preestablecidas; y los simuladores de RV, tienen uso ilimitado, pueden manejar múltiples escenarios diversos y complejos.

Las tecnologías de RV tienen como características comunes inmersión, percepción e interacción con el entorno. (Mendez et al. 2020)(Hudson et al. 2019) Los entornos inmersivos proporcionan un estímulo multisensorial, lúdico, multimedia e interactivo, y permiten controlar las condiciones de estimulación y repetir acciones como en el mundo real. Investigaciones realizadas en el contexto de entornos inmersivos en 3D muestran una fuerte conexión como tendencia entre la educación virtual y los entornos inmersivos (Comas-González et al. 2017). La tecnología in-

mersiva favorece la activación de herramientas cognitivas que ayudan en la elaboración de lo que el estudiante considera aprendizaje significativo. La inmersión puede reflejar una disminución de la carga cognitiva y un aumento de la actividad imaginativa que mejora la capacidad de conceptualizar.

El desarrollo de las tecnologías de RV ha marcado un creciente interés de formar habilidades en estudiantes y profesionales a través de entrenamientos en simuladores inmersivos. (Bernardo 2018) (Xin y Xiuyue 2018). El empleo de esta tecnología aporta más confianza para realizar los procedimientos quirúrgicos a partir de la experiencia realista que aporta la inmersión, añadiendo que el entrenamiento permite a los estudiantes conectarse a experiencias muy cercanas a las reales dentro de un entorno seguro, controlado, estandarizado con menos índices de riesgos, en especial si la práctica de la actividad real es costosa y riesgosa, y puede repetirse hasta que se adquiera un suficiente nivel de habilidad y se encuentre preparado para realizar los procedimientos en situaciones reales.

A través de estas experiencias, aumenta la motivación y compromiso con las actividades, dejando una impresión en la psiquis que favorece la memorización del contenido. Estudios realizados abordan la satisfacción de los estudiantes desde la experiencia del aprendizaje a través de entornos virtuales inmersivos destacando: La adquisición de nuevas habilidades y la satisfacción de aplicarlas eficazmente; motivación para aprender y mostrar un mejor rendimiento; la adquisición de conocimientos a partir de la exploración de un entorno interactivo no real; estimulación sensorial y sensación de presencia en ese entorno. (M. Bachen et al. 2017) (Comas-González et Al. 2017)

La formación médica, especialmente en cirugía ha sido un área que ha explorado las ventajas de la simulación. La necesidad de la simulación quirúrgica parte de la seguridad y repetitividad de los procedimientos y sus ventajas sobre el paradigma tradicional de entrenamiento. Por otra parte, enfrentar a los alumnos de forma precoz a realizar procedimientos de alta complejidad puede conducir a experiencias de aprendizaje insatisfactorias tanto para el estudiante como para el tutor, añadiendo posibles consecuencias que perjudiquen el paciente. (Inchausti et al. 2022)

El nivel de semejanza de los procedimientos reales y su simulación depende también de variables como el tiempo y los errores cometidos. La Apendicectomía, por ejemplo, es un procedimiento de cirugía común, su bajo nivel de dificultad es favorable para realizar simulaciones a través de la RV. Algunos trabajos referentes apuntan al tiempo de intervención que en promedio se realiza en un intervalo de 30 minutos y hasta 2 horas con ocurrencia de complicaciones.(Ali, Anwar, y Akhtar 2017) (Baumgarten, et al. 2018) (Hudson et al. 2019) (Malhotra et al. 2020). Sin embargo, para simular este procedimiento y que sea lo más semejante posible a la situación real requiere de capturar datos reales, de forma que se puedan comparar los datos capturados en la simulación y el entorno real, y así mejorar la simulación. En este sentido, desarrollar un simulador autoadaptativo para prácticas de Apendicectomía, requiere de un estudio de normalización de las variables, principalmente del tiempo y los posibles errores durante el procedimiento. De esta forma un modelo que otorgue calificaciones y ubique a los estudiantes según su nivel de conocimiento tendría menor grado de incertidumbre.

Un sistema de aprendizaje adaptativo se refiere a su capacidad de producir cambios en su entorno para situaciones específicas, cuando sea necesario. El aprendizaje adaptativo permite personalizar el aprendizaje de cada alumno utilizando los recursos ofrecidos desde una plataforma, cubriendo el contenido en profundidad de una manera racional y sirviendo como herramienta para el alumno y el profesor (M. del C 2018). Estos sistemas se apoyan en técnicas de recopilación de datos para aprender de cada alumno y proporcionan una adaptación al contenido que muestran. Particularmente en la educación, la atención a la personalización del aprendizaje constituye el reto en el momento de hacer un simulador autoadaptativo.

El desarrollo de la tecnología ha enriquecido el campo de aplicación de entornos inmersivos. Si bien esta tecnología no se ha desarrollado en Cuba, se cuenta con un elevado capital humano especializado en el área de la medicina y al menos una Facultad de Medicina en cada provincia. A partir de las capacidades económicas con que cuenta el país, la formación médica apoyada en la RV favorecería la disminución de tiempos de aprendizaje en los estudiantes, el ahorro de instrumental

médico, y el aumento de la capacidad para tomar decisiones ante situaciones complejas en un entorno simulado que pudieran producirse en el entorno real. Asimismo, la posibilidad de hacer un entrenamiento público permitiría mayor acceso a grupos más amplios de estudiantes y profesionales. El simulador provee una experiencia motivadora que se percibe casi real, sin embargo, estas experiencias se comportan muy semejantes en cada persona.

Un problema común con los sistemas de RV para la formación es que los estudiantes experimentan las mismas rutinas de entrenamiento, ya que estos sistemas no están personalizados para patrones de aprendizaje individuales (Vaughan, Gabrys, y Dubey 2018). A partir de esta problemática se plantea la necesidad de desarrollar simuladores autoadaptativo independientemente del contexto de aplicación. Los planteamientos anteriormente expuestos evidencian que el desarrollo de simuladores inmersivos autoadaptativo para la formación en cirugía constituye una valiosa aplicación. Sin embargo, a pesar de los avances logrados en el desarrollo de simuladores de este tipo se considera aún insuficientes los aportes prácticos en esta temática, con carencia de metodologías o estrategias más detalladas para producir simuladores bajo este principio (Véliz, Correa, y Kugurakova 2021). Se desarrollan simuladores que integran la IA en el mismo simulador. La integración de la IA en el propio simulador dificulta el enriquecimiento, reentrenamiento y reutilización de los modelos implementados para el aprendizaje autoadaptativo a través de lógica difusa.

Sobre la base de la situación expuesta se identifica el siguiente **problema científico**: ¿Cómo contribuir a la personalización de un simulador de cirugía de apendicectomía basados en RV a las prácticas de los estudiantes de medicina para ahorrar tiempo de estudio y ayudar a evitar errores y complicaciones a la hora de la cirugía real?

Para darle solución al problema planteado se determinó como **objetivo general** de la investigación desarrollar componentes para la personalización de un simulador de apendicectomía autoadaptativo basado en Realidad Virtual.

Para ello se tendrá como **objeto de estudio** los simuladores autoadaptativos basados en realidad virtual.

Se identifica además como **campo de acción** la personalización de simuladores auto adaptativos de cirugía basados en realidad virtual.

Para alcanzar este objetivo general se plantean **objetivos específicos** como los que se describen a continuación.

1. Construir las bases teóricas y técnicas para la personalización del aprendizaje adaptativo en simuladores de realidad virtual.
2. Realizar el análisis, diseño e implementación de componentes para la personalización de un simulador autoadaptativo de realidad virtual para la formación de profesionales de la especialidad de cirugía.
3. Definir un enfoque de aprendizaje adaptativo centrado en el usuario para simuladores de cirugía basados en realidad virtual.
4. Realizar pruebas de calidad de software a la propuesta de solución para verificar el correcto funcionamiento de la misma

Durante la investigación se aplicaron los siguientes métodos científicos:

Histórico-lógico: se utilizó para hacer un estudio detallado de los trabajos anteriores referentes

a simuladores quirúrgicos, como fue la consulta de tesis sobre el simulador quirúrgico Kheipros de la Universidad de Ciencias Informáticas que hacen referencia al tema de la investigación.

Además de artículos sobre simulador autoadaptativo de realidad virtual.

Inductivo - Deductivo: Se realizó un estudio del proceso de desarrollo de simuladores en diferentes proyectos de la universidad obteniéndose elementos que pudieran aplicarse a este proyecto.

Análisis-síntesis: se realizó el análisis de documentos y teorías referentes a este tema, seleccionando los elementos concretos que se ajustan al desarrollo de la investigación.

Métodos empíricos

Dentro de este método se utilizó:

Entrevista: se entrevistaron expertos en el tema para obtener información acerca del procedimiento en cuestión, además de diferentes profesores, líderes de proyectos y estudiantes.

Observación: fue utilizada para identificar el impacto de diferentes implementaciones del modelo desde la interacción con estudiantes y especialistas en cirugía. Con este método se logró corregir importantes características del modelo.

Métodos cualitativos

El **grupo focal** se utilizó como método complementario para valorar los resultados desde la opinión de los especialistas en cirugía.

Estructura del Documento

El documento se encuentra dividido, en resumen, introducción, tres capítulos, conclusiones, recomendaciones, glosarios de términos, acrónimos, referencias bibliográficas y apéndices. En el **Capítulo 1** se resume los fundamentos teóricos principales relacionados con el aprendizaje adaptativo en simuladores virtuales.

Por otra parte, el **Capítulo 2** presenta la propuesta de modelo de aprendizaje inmersivo autoadaptativo. El **Capítulo 3** recoge las validaciones realizadas al modelo propuesto.

Capítulo 1: Fundamentación Teórica

Introducción:

En el presente capítulo se exponen los referentes fundamentales del trabajo. Estos facilitan la comprensión del objeto de estudio, así como los análisis derivados de la investigación para una propuesta coherente del modelo. Se detallarán las características de los Simuladores Quirúrgicos realizando un estudio del arte y los temas fundamentales recogidos en este capítulo son simulación y RV, enfoques de aprendizaje adaptativo basados en RV, entre otros que conformarán la base cognoscitiva para desarrollar la solución deseada.

1.1 Conceptos Fundamentales

Cuando está en desarrollo una investigación tiende a existir problemas en la comunicación y entendimiento sobre el tema. Esto se debe al manejo de diferentes términos y conceptos que pueden ser desconocidos para el personal externo, de ahí la importancia de definir y argumentar los que pueden ser los principales conceptos que se utilizarán a lo largo de la presente investigación.

Realidad virtual: Es una tecnología especialmente adecuada para la enseñanza debido a su facilidad para captar la atención de los estudiantes mediante su inmersión en mundos virtuales relacionados con las diferentes ramas del saber, lo cual puede ayudar en el aprendizaje de los contenidos de cualquier materia (Calderón, Tumino, y Bournissen 2020). Se puede considerar como una forma humana de visualizar, manipular e interactuar con ordenadores y datos complejos

La **simulación** es una técnica que aporta destreza, habilidad mental y capacidad de respuesta asertiva cuando indudablemente se necesita y es absolutamente necesaria e impostergable (Parra y García 2020). La simulación no es más que una representación de una actividad de la vida real, siendo uno de los objetivos principales la obtención de habilidades a través de experiencias de aprendizaje.

Capítulo 1: Fundamentación Teórica

Los **simuladores quirúrgicos** constituyen entornos virtuales que representan una alternativa de formación ya que permitirán la creación de ambientes de simulación interactivos en tres dimensiones donde el cirujano tiene las mismas percepciones visuales y táctiles que, durante una situación con un paciente real, se reduce de esta manera los errores quirúrgicos durante el período de aprendizaje del cirujano que pondrían en peligro la vida del paciente. (Casal 2018) (Vasquez 2020)

Autoadaptabilidad: Hace referencia a los sistemas informáticos, constituye la capacidad para ajustarse automáticamente al contexto de ejecución en un ciclo cerrado de retroalimentación. (Véliz Vega, Correa Madrigal, y Kugurakova 2021)

Aprendizaje adaptativo: Se define como la personalización educativa de técnicas de aprendizaje, que surgen tras un proceso de diferenciación que identifica las necesidades específicas del estudiante y ofrece diferentes posibilidades. Por lo que se debe diseñar una nueva experiencia de aprendizaje para los estudiantes y lograr por medio del análisis de los datos, la adaptabilidad del aprendizaje de acuerdo a las necesidades, fortalezas o debilidades y que posteriormente facilite anticiparse al aprendizaje. (Lozano y González 2018)

Personalización: Preparación para las necesidades específicas de cada persona a la que se destina. Esta tiene como objetivo principal implicar al estudiante en el proceso de aprendizaje de manera directa y activa.¹

1.2 Simulador de realidad virtual en la medicina

En el mundo actual existen diversos simuladores para mejorar la eficiencia médica. A continuación, se describen algunos de ellos.

Brain Advanced Computer Surgery Simulator o BACSIM, es un sistema desarrollado desde hace más de seis años con la intención de construir una estación de trabajo que recreara un escenario de procedimiento quirúrgico, que utiliza desde

¹ <https://www.dicenlen.eu/es/diccionario/entradas/personalizacion>

Capítulo 1: Fundamentación Teórica

realidad virtual y hasta modelos matemáticos de biomecánica de tejidos para representar técnicas de alto riesgo. El sistema tiene la capacidad de entrenar a profesionales para que estos experimenten la microcirugía, respetando las escalas reales de los instrumentos y utilizando un visor tridimensional que permite al cirujano visualizar una escena similar a la que vería en un entorno verdadero en el microscopio electrónico. El sistema cuenta tanto con elementos físicos como virtuales, representando el sitio quirúrgico, con dos robots con los que se manipulan los elementos virtuales, dando incluso una sensación de tacto del tejido, una experiencia similar a estar dentro de un quirófano. El módulo está instalado de forma permanente en el hospital General de México, donde hasta ahora 15 cirujanos han realizado evaluaciones experimentales con residentes, arrojando resultados que permiten evaluar el simulador como entrenamiento de aneurismas². En los tejidos blandos del cerebro y las estructuras anatómicas, como vascularidades, lóbulos, tumores y cráneo. (Rodríguez 2022) (Sánchez 2022).

Otro simulador de gran utilidad es el simulador robot Da Vinci tiene como objetivo reforzar las habilidades quirúrgicas de los médicos y ofrecer a los pacientes la opción de una intervención mínimamente invasiva, incluso cuando se trata de un procedimiento de elevada complejidad. El robot consiste en tres módulos: una consola ergonómica en la que el cirujano se sienta y opera el robot; un carro quirúrgico que sostiene y moviliza los instrumentos; y una torre de visión que controla las cámaras. Sus brazos son capaces de girar 520 grados y sostener simultáneamente tres utensilios quirúrgicos. Además, cuenta con una cámara tridimensional que se introduce en el cuerpo del paciente a través de cuatro o cinco incisiones de poco más de un centímetro. La excepcional precisión de sus movimientos brinda numerosas ventajas en comparación con la cirugía laparoscópica convencional, tanto para el médico como para el paciente, por ejemplo, visión tridimensional y reducción del temblor fisiológico, menor daño a tejidos, hasta una menor estancia hospitalaria lo que al final se traduce en beneficios económicos para la salud pública. Es importante señalar que Da Vinci no es un robot autónomo, por lo que su

² Ensanchamiento anormal de las paredes de una arteria. (Aramendi 2019)

Capítulo 1: Fundamentación Teórica

operación depende completamente de las instrucciones que recibe por parte de los médicos. (González 2018)

1.3 Aprendizaje adaptativo basado en realidad virtual

En un entorno virtual no inmersivo, el usuario tiene la sensación de ver todo desde una ventana. Sin embargo, utilizando visores de RV, el usuario experimenta estar dentro del entorno virtual, aunque no se encuentre físicamente dentro del mundo virtual. La sensación de inmersión es posible a través de la alineación de los sentidos hacia este entorno. Para lograr una sensación de presencia es necesario configurar los recursos visuales de tal manera que no se perciba la mediación entre la persona y el objeto virtual. En este sentido, la calidad gráfica influye en el diseño de este entorno. (Véliz Vega, Correa Madrigal, y Kugurakova 2021)

Los elementos básicos de la simulación de entrenamiento basada en RV incluyen dispositivos hápticos, agentes autónomos, tecnologías adaptativas, evaluación y clasificación de retroalimentación, y pantalla montada en la cabeza. Un estudio realizado en este campo apunta a la evolución de las tecnologías autoadaptables en el entrenamiento de RV (Vaughan, Gabrys, y Dubey 2018) .La figura 1 muestra estos elementos.

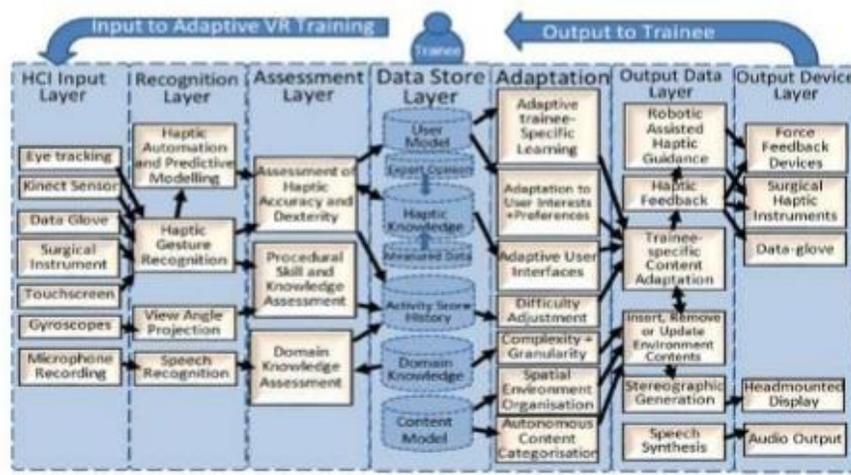


Figura 1: Estructura de interacción de componentes (Vaughan, Gabrys, y Dubey 2018)

Existen diversos enfoques de aprendizaje adaptativo con realidad virtual, dentro de ellos se encuentra el propuesto por Bad, Wojtok y Martin Hirsch. Este enfoque

Capítulo 1: Fundamentación Teórica

consta de cuatro componentes el modelo de usuario, el de detección de emociones, el de detección de interacciones y por último el de optimización. En este enfoque se toman en cuenta las preferencias y objetivos de los usuarios, las emociones son supervisadas con la interacción del usuario. (Wojtok, Hirsch, y Bab 2018)

Otro enfoque es el marco de trabajo desarrollado por Zahabi y Razak Abdul. Ellos consideraron tres componentes fundamentales para adaptar un sistema de entrenamiento basado en realidad virtual, que son la medición del rendimiento del alumno, la variable adaptativa y la lógica adaptativa (Zahabi, Abdul, y A 2020). El aprendizaje se realiza mediante la recopilación de medidas de rendimiento y cinéticas, además de la información del alumno.

- Medidas de rendimiento: hace referencia al tamaño del error y tipo de error en realizar una actividad.
- Información del alumno: no es más que la experiencia, edad, estilo de aprendizaje del estudiante.
- Medidas cinéticas: es el rango de movimiento y fuerza empleada en realizar la tarea.

Este modelo agrupa un amplio número de variables de adaptación clasificadas en dos grupos: antes del entrenamiento y durante el entrenamiento, y otras variables recogidas en dos componentes referentes a las medidas de comportamiento y la lógica adaptativa. (Zahabi, Abdul, y A 2020)

Variables	Componentes	Área de aplicación	Aporte	Referencia
Reconocimiento de gestos, habilidad de	Siete (7) capas: entrada de Interacción	Múltiple	Elementos básicos de la simulación de	(Vaughan et al., 2016b)

Capítulo 1: Fundamentación Teórica

procedimiento y evaluación de conocimiento, dominio y medida de conocimiento, preferencias e intereses de usuario, retroalimentación háptica, ajuste de dificultad, adaptación de interfaces	Humano Computador (HCI); reconocimiento; evaluación; recolección de datos; adaptación; salida de datos; salida de dispositivos		entrenamiento basada en RV	
Comportamiento cognitivo, perceptivo y psicomotor, habilidades y retención	Modelado; comparación; entrenamiento óptico	Formación médica militar	Marco de trabajo	(K. C. Siu et al., 2016)
Sistema de tutoría inteligente, habilidades	Modelo detallado de una ciudad; tráfico de inteligencia artificial; motor de conducción física que interactúa con el conductor	Conducción de vehículos	Sistema	(Ropelato, Zünd, Magnenat, Menozzi, & Sumner, 2017)
Objetivos, preferencias, conocimiento,	Modelo de usuario; Optimización;	Educación	Enfoque	(Wojtok et al., 2018)

Capítulo 1: Fundamentación Teórica

emociones e interacción del usuario	detección de emociones; detección de interacción			
Hábitos y habilidades de usuario, optimización	Preevaluación; optimización; entrenamiento; posevaluación	Conducción de vehículos	Enfoque	(Lang, Wei, Xu, Zhao, & Yu, 2018)
Medidas de comportamiento de usuario, sistema de reglas, métodos de optimización, retroalimentación, adaptación de contenido, refuerzo de aprendizaje, datos fisiológicos	Medidas de comportamiento; lógica adaptativa; variable adaptativa	Conducción de vehículos	Marco de trabajo	(Zahabi et al.,2020)

Tabla 1: resume los componentes más destacados para la adaptación de SRV de una muestra de trabajos estudiados (Vega et al., 2021a).

Entre los trabajos estudiados predomina la concepción de variables referentes a un modelo de usuario, aunque no explícitamente agrupadas en un solo componente. Entre las principales variables asociadas a este modelo se destacan las medidas de comportamiento, habilidades, conocimientos, y preferencias. En trabajos analizados persiste una relación entre el comportamiento y las emociones del usuario (Véliz Vega, Correa Madrigal, y Kugurakova 2021).

Para esta investigación se concebirá el uso de un modelo que se centre en el usuario, se representa gráficamente en la figura 2 el modelo a utilizar propuesto por (Véliz Vega, Correa Madrigal, y Kugurakova 2021).

Capítulo 1: Fundamentación Teórica

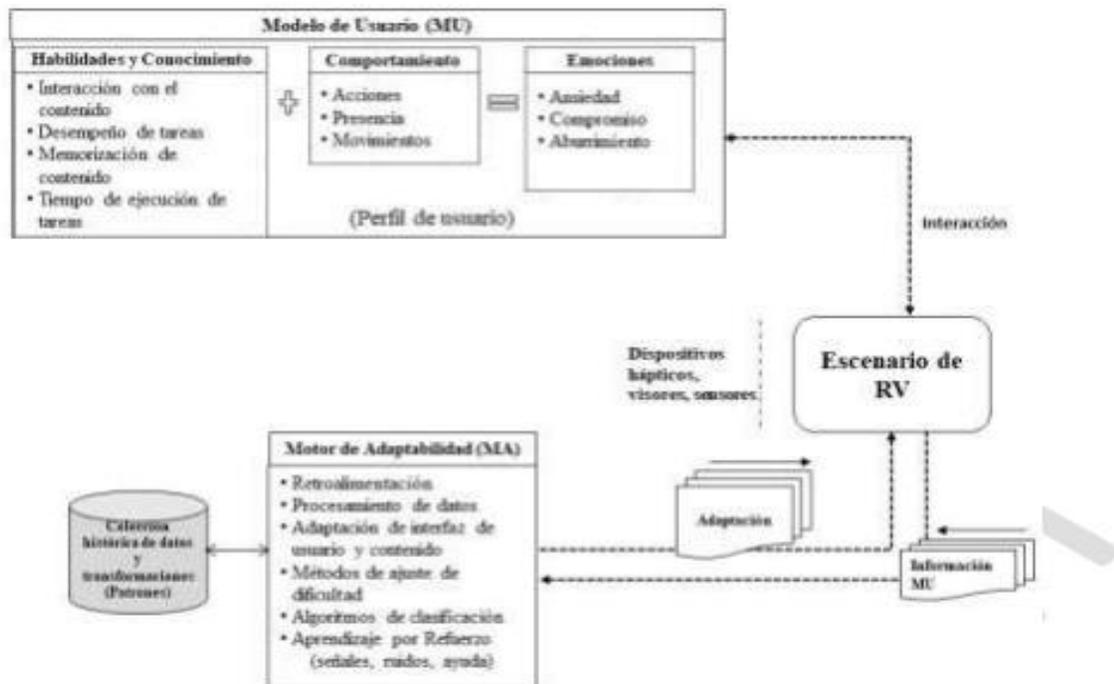


Figura 2: Enfoque centrado en el usuario. (Véliz Vega, Correa Madrigal, y Kugurakova 2021)

Como se puede apreciar en la figura 2 el enfoque del proyecto general está compuesto por tres componentes, el modelo de usuario (MU), el escenario de realidad virtual (ERV) y el motor de adaptabilidad (MA).

El MU provee características del usuario que se dividen en tres grupos de variables.

- **Habilidades y conocimiento:** en este subcomponente se agrupan datos que se relacionan con las habilidades y los conocimientos de los usuarios, cuando estos realizan una actividad y adquieren conocimiento.
- **Emociones:** estas variables brindan los datos emocionales del usuario obtenidos durante el entrenamiento. Para la obtención de las emociones se utilizan dispositivos hápticos y sensores.
- **Comportamiento:** esta variable nos muestra la fase de aprendizaje que se encuentra el usuario para realizar la tarea, según datos históricos almace-

Capítulo 1: Fundamentación Teórica

nados en relación con el comportamiento del usuario para enfrentarse a dicha actividad.

El escenario ERV no es más que la forma de comunicar el MU y el MA. En el escenario de Realidad Virtual para las prácticas de cirugía en un quirófano virtual es más efectivo el uso de guantes hápticos. Ya que los mismos aportan una mejor sensación de tomar los instrumentos con la mano como en una operación real aumentando así la sensación de inmersión y una mejor forma de obtención de datos. El MA procesa la información obtenida y se apoya en una colección de datos que contiene patrones de comportamiento para la optimización, actualización del contenido y la interfaz de usuario. Durante el proceso de adaptación, el motor puede llevar al proceso de aprendizaje por refuerzo para generar representaciones eficientes frente a tareas complejas. (Véliz Vega, Correa Madrigal, y Kugurakova 2021).

A partir de este enfoque para este trabajo de diploma se decidió centrarse en un perfil de usuario reducido específicamente en las habilidades y conocimiento del modelo de usuario y su interacción con el entorno de realidad virtual.

1.4 Simulador de apendicetomía basado en realidad virtual con un enfoque centrado en el usuario.

El desarrollo de este trabajo forma parte de un proyecto en conjunto con la Universidad de Kazan de Rusia por lo que para la realización de este se trabaja sobre el simulador creado por este convenio, por lo que todo lo relacionado al desarrollo de este será basado en este simulador.

El simulador ejecuta una operación básica de extracción del apéndice. Está compuesto por 3 etapas fundamentales que son: limpieza y apertura de la zona afectada, extracción del apéndice y por último sutura de la herida. Está compuesto por varias escenas entre las cuales se encuentran: "Settings", "RoomExc", "Main",

Capítulo 1: Fundamentación Teórica

“ControllerTutorial”. “Settings” se encarga de las configuraciones de los ajustes del simulador, “RoomExc” en la cual se elige si se desea empezar la operación o conocer la distribución de la sala de operaciones, “ControllerTutorial” es un pequeño ejemplo de manipulación de los instrumentos con los cuales se va a interactuar en la operación, “Main” es la escena en la cual se interactúa con los objetos o herramientas y el paciente para dar cumplimiento a la operación. La simulación consta de 128 pasos para el cumplimiento de la cirugía entre los cuales se encuentran por ejemplo la limpieza de la zona de la piel de la operación utilizando el objeto “thumb forceps”, un instrumento que sirve a modo de pinza, y un algodón cubierto de yodo, la apertura del abdomen del paciente utilizando el objeto “scalpel”, un instrumento que tiene función de cúter, cortando la piel y la grasa subcutánea, la extracción del apéndice y la sutura de la herida. Además, presenta un contador en forma de tiempo con el cual se puede ver la duración de la tarea. Este simulador no presenta ninguna forma de individualizar el proceso quirúrgico en función del usuario que está realizando la simulación por lo que para dar respuesta al problema científico se trabajó sobre él.

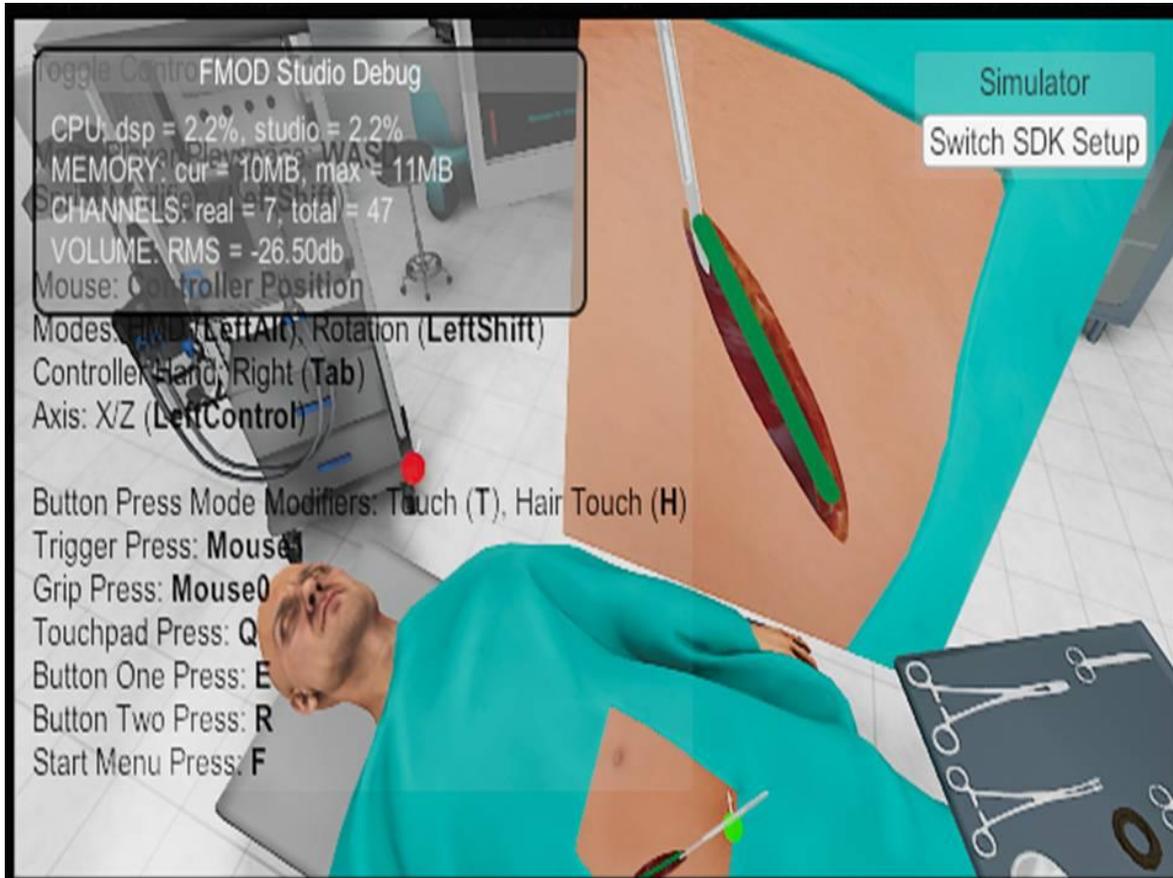


Figura 3: Incisión del paciente

La figura 3 muestra uno de los pasos por los que transcurre el simulador, específicamente el paso de incisión donde se procede a cortar la piel para poder acceder al apéndice.

1.5 Lenguaje de programación

Un lenguaje de programación consiste en un conjunto de órdenes o comandos que describen el proceso deseado. Cada lenguaje tiene sus instrucciones y enunciados verbales propios, que se combinan para formar los programas de cómputo. Los lenguajes de programación no son aplicaciones, sino herramientas que permiten construir y adecuar aplicaciones. (de la Vega 2019)

Existen diversos lenguajes de programación como los siguientes:

Java: Es uno de los lenguajes de programación más disruptivos hasta la fecha. Ha logrado la independencia de la plataforma mediante el desarrollo de Java Virtual

Capítulo 1: Fundamentación Teórica

Machine (JVM), que abstraigo el sistema operativo de bajo nivel de los desarrolladores y dio el primer lenguaje de programación “Escribir una vez, ejecutar en cualquier lugar”. En los últimos años, Java ha perdido algunos de sus mercados debido a los lenguajes modernos altamente amigables para los desarrolladores y al surgimiento de otros lenguajes, especialmente Python, JavaScript. Además, JVM no es muy amigable con la nube debido a su voluminoso tamaño. Oracle ha introducido recientemente altas tarifas de licencia para JDK, lo que afectará aún más la popularidad de Java. Y a pesar de todo, Java sigue siendo el lenguaje de programación número uno para diversas empresas porque constituye para muchos un lenguaje de programación potente, rico en funciones, multi paradigmático e interpretado con una curva de aprendizaje moderada. Es compatible con versiones anteriores, lo cual es un requisito crucial para las aplicaciones comerciales y el tiempo de ejecución de Java JVM es una obra maestra de ingeniería de software y una de las mejores máquinas virtuales de la industria. Además, JVM proporciona a Java un recolector de basura muy avanzado. (Carranza 2022)

C++: Es una mejoría sobre muchas de las características de C, y proporciona capacidades de programación orientada a objeto que promete mucho para incrementar la productividad, calidad y reutilización del software. Al igual que Java, C++ también se está modernizando y adaptando constantemente con cambios en el hardware o el ecosistema. El C++ moderno es un lenguaje mucho más seguro y productivo en comparación con las versiones anteriores de C++, también brinda control total sobre el hardware subyacente y puede ejecutarse en cada plataforma y aprovechar todo tipo de hardware, ya sea contenedor, nube, dispositivos móviles o micro controlador. Es increíblemente rápido y se usa mucho en sistemas críticos para el rendimiento y con recursos limitados. Muchos programas fantásticos como Java Virtual Machine, Chrome V8 Engine, muchas bases de datos están escritos en C++. (Vicente y Roldán 2018)

C#: Es el lenguaje más popular de Microsoft, inspirado en C y C++ de propósito general y es uno de los lenguajes más populares de la industria del software. Es el lenguaje de cabecera de Microsoft, aunque se puede usar en múltiples plataformas de desarrollo de aplicaciones de todo tipo. Es un lenguaje de tipado estático y

Capítulo 1: Fundamentación Teórica

multiparadigma, aunque principalmente orientado a objetos. Microsoft presenta actualizaciones muy frecuentes, por lo que resulta bastante evolucionado, ofreciendo herramientas poderosas para los desarrolladores. Con C# podemos desarrollar todo tipo de proyectos, desde programas de consola, servicios web, hasta programas de interfaz gráfica, todo bajo la plataforma NET de Microsoft. Además, es posible desarrollar aplicaciones para móviles mediante la plataforma de código abierto Xamarin y juegos usando el motor de Unity, entre otras cosas. Su sintaxis es fácil de aprender y muy intuitiva, lo que hace que cualquier persona familiarizada con algún otro tipo de lenguaje de programación aprenda en poco tiempo a utilizarlo. (Muñoz 2018)

Para dar respuesta a la problemática de esta investigación se decidió utilizar el lenguaje C# debido a que es el empleado en el proyecto vinculado al simulador al cual se le van a incorporar componentes. Las características de sintaxis sencilla que facilita al desarrollador la escritura de código y orientación a componentes facilitaron la elección de C# como lenguaje de programación.

Python³: Es un lenguaje de programación potente y fácil de aprender. Tiene estructuras de datos de alto nivel eficientes y un simple pero efectivo sistema de programación orientado a objetos. La elegante sintaxis de Python y su tipado dinámico, junto a su naturaleza interpretada lo convierten en un lenguaje ideal para el desarrollo rápido de aplicaciones en muchas áreas, para la mayoría de las plataformas. También es apropiado como un lenguaje para extender aplicaciones modificables (Alvares 2020). Además de que es libre y gratuito, tiene incorporado diversas librerías dentro de las que se encuentran las siguientes.

Requests: Librería que te permitirá hacer peticiones por http de una manera sencilla. Cuando se hace una petición, requests automáticamente decodificará el contenido extraído de un servidor y la mayoría de caracteres únicos serán decodificados correctamente. (Reitz 2022c)

Scrapy: Es framework que te permitirá rastrear sitios web y extraer datos estructurados que pueden utilizarse para una amplia gama de aplicaciones, como la ex-

³ Tutorial de Python <https://docs.python.org/es/3.9>

Capítulo 1: Fundamentación Teórica

tracción de datos, el procesamiento de información o el archivo histórico. Fue diseñado originalmente para el rastreo web, también se puede usar para extraer datos con APIs como los Amazon Associates Web Services o como un rastreador web de propósito general. (Reitz 2022a)

NumPy: es el paquete más usado para computación científica con Python. Contiene, entre otras cosas un poderoso objeto de matriz N-dimensional, funciones sofisticadas y herramientas para la integración de código C / C ++ y Fortran. (Reitz 2022b)

Flask: es un micro framework escrito en Python y concebido para facilitar el desarrollo de aplicaciones Web. Incluye un servidor web de desarrollo. Además, tiene un depurador y soporte integrado para pruebas unitarias. Es compatible con python3 y wsgi. Soporta el uso de cookies seguras y sirve para construir servicios web como APIs REST o aplicaciones de contenido estático. (Pallets 2022)

1.6 Herramientas a utilizar

Visual Studio Code: es un editor de código en términos sencillos. Es un editor libre que ayuda al programador a escribir código, a depurar y corregir el código utilizando el método inteligente. En términos normales, facilita a los usuarios a escribir el código de una manera fácil. Entre las principales características de este editor podemos mencionar que tiene una amplia variedad de temas para configurar el entorno de trabajo, soporte para múltiples lenguajes de programación (Aeschlimann 2018).

Una de las características por las cuales se decidió hacer uso de este editor son la gran compatibilidad con la mayoría de lenguajes de programación y que detecta fácilmente, si hay algún fallo o referencia entre lenguajes.

El uso de este editor trae consigo muchos beneficios para el desarrollo de cualquier producto dentro los que se destacan la detección y autocompletado de código. Está conectado con Git⁴ o puede conectarse con cualquier otro repositorio pa-

⁴ Es el sistema de control de versiones moderno más utilizado del mundo. Es un proyecto de código abierto maduro y con un mantenimiento activo que desarrolló originalmente Linus Torvalds, el famoso creador del kernel del sistema operativo Linux, en 2005. (Quemada 2019)

Capítulo 1: Fundamentación Teórica

ra sacar o guardar las instancias. Además, viene con soporte incorporado para aplicaciones web.

Unity: es lo que se conoce como un motor de desarrollo o motor de juegos. El término motor de videojuego, hace referencia a un software el cual tiene una serie de rutinas de programación que permiten el diseño, la creación y el funcionamiento de un entorno interactivo; es decir, de un videojuego. Dentro de las funcionalidades típicas que tiene un motor de videojuegos se encuentran renderizar gráficos 2D y 3D, simular las leyes de la física, animaciones, sonidos, inteligencia artificial y programación o scripting (Rosales 2022). En la presente investigación se hará uso de esta herramienta que te permite crear animaciones para diversas plataformas (PC, videoconsolas, móviles, etc.) mediante un editor visual y programación vía scripting, y pudiendo conseguir resultados totalmente profesionales.

La razón más importante por la que se va a utilizar este motor es debido a que la versión del software, del proyecto en el que se basa este trabajo de curso, fue implementada con este motor de desarrollo. Unity se conforma de varios elementos que permiten realizar diferentes tareas como las mencionadas anteriormente. Dentro de esos elementos se encuentran los siguientes.

Escenas: Contienen todos los objetos de la animación. Pueden ser usadas para crear un menú principal, niveles individuales o cualquier otro entorno. Cada archivo de escena, es como un nivel único. En cada escena, se coloca el ambiente, obstáculos, y decoraciones, el diseño esencial y la construcción que tendrá la animación (Pérez 2018). La figura 4 muestra la sala de cirugía la cual es una escena del simulador propuesto.

Capítulo 1: Fundamentación Teórica

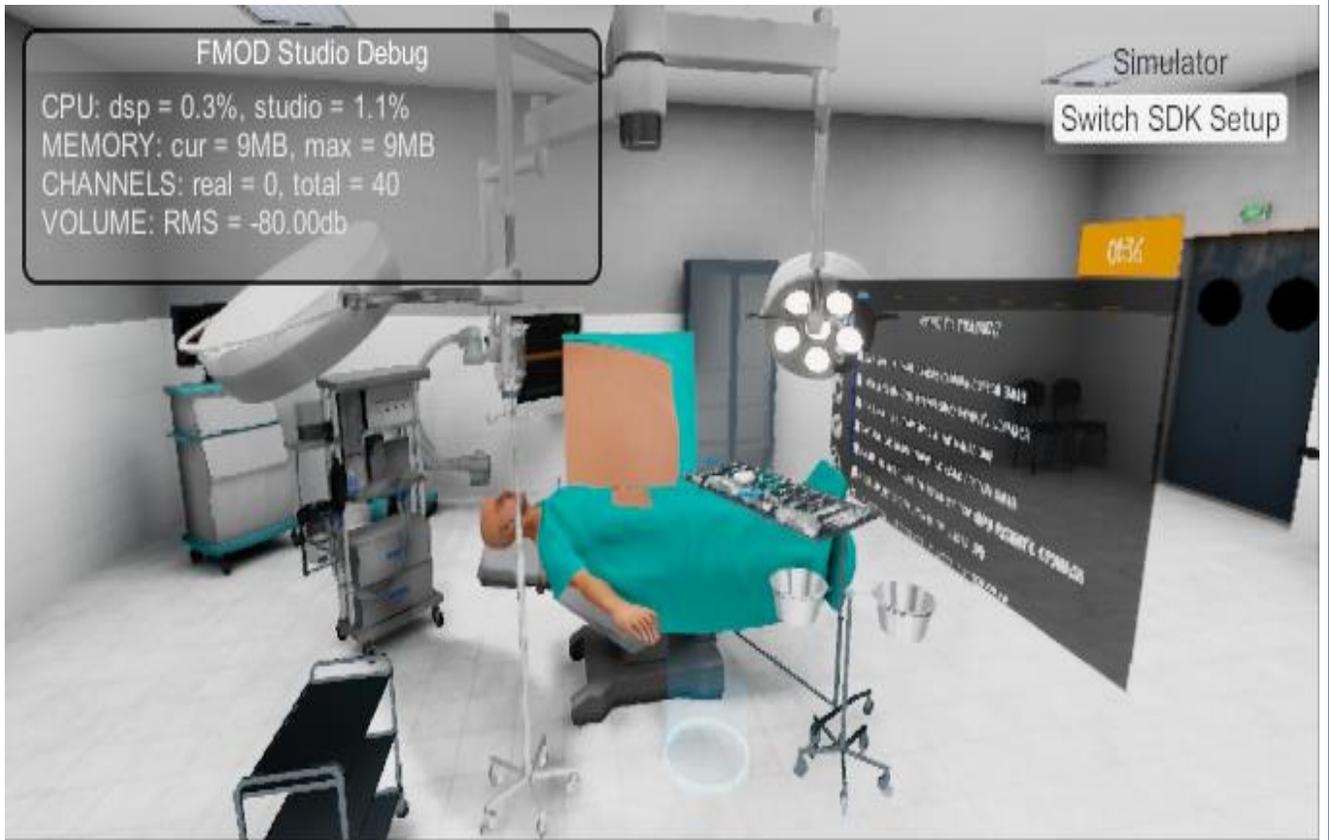


Figura 4: Escena del Simulador Mein (Sala de cirugía)

GameObject: Son objetos fundamentales en Unity que representan personajes y el escenario. Estos no logran nada por sí mismos, pero funcionan como contenedores para Componentes que implementan la verdadera funcionalidad. Por ejemplo, un objeto Light es creado al adjuntar un componente Light a un GameObject.

Script: Es un documento que contiene instrucciones de programación en un determinado lenguaje, que es el que entiende la herramienta que estamos utilizando para desarrollar cualquier aplicación que requiera programación (A. L. Hernández 2022). En nuestro caso vamos a utilizar scripts dentro del motor Unity, estos scripts estarán escritos en lenguaje C# y la mayor parte del tiempo los scripts cumplen una determinada función, utilizando los objetos y componentes de la jerarquía y realizando operaciones lógicas y matemáticas necesarias para el desarrollo de cualquier animación.

Capítulo 1: Fundamentación Teórica

MonoBehaviour: Es una clase de programación, esto quiere decir que es un conjunto de variables, objetos y funciones que cumplen una determinada función. Cuando se crea un nuevo script en Unity, lo que se hace en realidad es crear una nueva clase de programación que tendrá el mismo nombre que le dimos al Script y que además extenderá de MonoBehaviour, es decir que heredará su comportamiento de la clase MonoBehaviour.(Pérez 2018)

Interfaz: Es el modo en que el usuario puede interactuar con el simulador en este caso. Por ejemplo, el menú de inicio o los botones para poder mover los personajes. La figura 5 es una interfaz que corresponde al simulado de la solución.

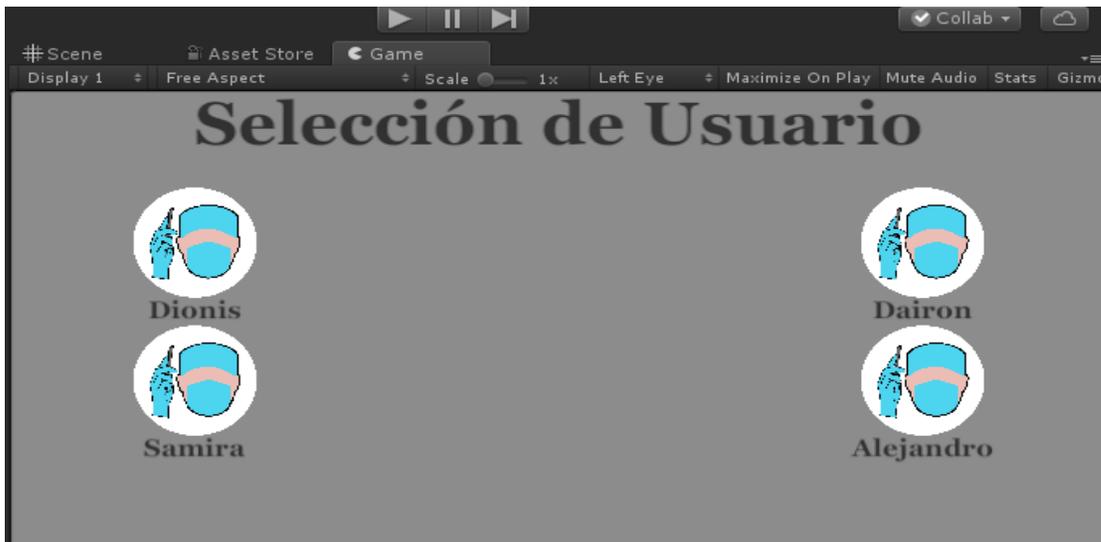


Figura 5: Interfaz de usuario Canvas/UserPanel (elaboración propia)

En esta escena se muestra inicialmente la interfaz de selección de usuario es en la cual se selecciona el estudiante que realizara el entrenamiento. Además, se valida si es realmente el usuario que realizara la actividad para pasar a la interfaz de ajustes, en donde se escoge el idioma y experiencia con el simulador. En caso contrario se regresa a la interfaz de usuario. En la opción de experiencia de usuario es recomendable si el usuario tiene poca experiencia pasar por un entrenamiento previo, es decir seleccionar la escena de entrenamiento.

API: Una interfaz de programación de aplicaciones (API), es un mecanismo que permite la comunicación e intercambio de información entre sistemas. En el contexto de los Datos Abiertos, el término generalmente se refiere a APIs sobre la

Capítulo 1: Fundamentación Teórica

Web, denominadas en algunos ámbitos API Web, que es un medio habitual para soportar el intercambio de información dentro y entre organizaciones. Esta característica implica que una API ofrece un conjunto de funcionalidades sobre un servidor en la Web para ser utilizadas por aplicaciones cliente mediante el uso de procedimientos estándar. (García 2020) (Espinosa 2018) En otras literaturas el término API hace referencia a las funcionalidades que aporta un cierto servicio a un software facilitando que pueda ser utilizado por otro software para mejorar sus resultados. Normalmente no es un resultado en sí mismo, sino que sirve de enlace entre un software ya creado y otro al que este le puede resultar útil, lo que se conocería como una interacción “software a software”. (Estévez y Lamela 2018)

Para la utilización del **motor autoadaptativo** que constituye uno de los componentes del simulador se utilizaría una API Rest empleando la librería flask, para capturar los datos enviados desde el simulador de cirugía hacia el modelo difuso. De forma que el modelo se consulta a través de la API al no estar implementado en el propio simulador. De esta forma, el simulador solo provee los parámetros de entrada y el modelo devuelve una respuesta.

1.7 Metodologías de desarrollo

El desarrollo de software no es una tarea fácil. Prueba de ello es que existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo. Por una parte, tenemos aquellas propuestas más tradicionales que se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán. Otra aproximación es centrarse en otras dimensiones, como por ejemplo el factor humano o el producto software. Esta es la filosofía de las metodologías ágiles, las cuales dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas (Pressman 2010). En la década de los noventa surgieron metodologías de desarrollo de software ligeras, más adelante nombradas como metodologías ágiles, que buscaban reducir la probabilidad de fracaso por subestimación de cos-

Capítulo 1: Fundamentación Teórica

tos, tiempos y funcionalidades en los proyectos de desarrollo de software. (Cerna et al. 2021)

Dentro de las metodologías ágiles se destacan las descritas a continuación.

SCRUM: Es un marco de trabajo por el cual las personas pueden acometer problemas complejos adaptativos, a la vez que entregar productos del máximo valor posible productivo y creativo. Scrum es ligero fácil de entender y extremadamente difícil de llegar a dominar. Scrum es un marco de trabajo de procesos que ha sido usado para gestionar el desarrollo de productos complejos desde principios de los años 90. El marco de trabajo Scrum consiste en los Equipos Scrum, roles, eventos, artefactos y reglas asociadas. Cada componente dentro del marco de trabajo sirve a un propósito específico y es esencial para el éxito del uso de esta metodología. (Schwaber y Jeff 2018)

Kanban: Se basa en el desarrollo incremental, dividiendo el trabajo en partes (historias de usuario, tareas, etc.). Uno de los principales aportes es que utiliza técnicas visuales para ver la situación de cada parte. El termino Kanban se refiere a una "tarjeta de señal" que representa una unidad de trabajo, la cual se mueve a través del flujo de la organización solo cuando existe la capacidad de tomar la tarea en el siguiente paso del proceso. Kanban provee transparencia en el trabajo y también en el proyecto. Todo proceso debe ser adaptado a las características organizacionales en donde va a ser aplicado. Es recomendable utilizar Kanban combinado con otras metodologías, como por ejemplo Scrum o XP. (Reyes y Pineda 2020)

XP: Es una metodología ágil para el desarrollo de software y consiste básicamente en ajustarse estrictamente a una serie de reglas que se centran en las necesidades del cliente para lograr un producto de buena calidad en poco tiempo, centrada en potenciar las relaciones interpersonales como clave para el éxito del desarrollo de software. La filosofía de XP es satisfacer al completo las necesidades del cliente, por eso lo integra como una parte más del equipo de desarrollo. Promueve el trabajo en equipo, preocupándose en todo momento del aprendizaje de los desarrolladores y estableciendo un buen clima de trabajo. Este tipo de programación es

Capítulo 1: Fundamentación Teórica

la adecuada para los proyectos con requisitos imprecisos, muy cambiantes y donde existe un alto riesgo técnico. XP está diseñada para el desarrollo de aplicaciones que requieran un grupo de programadores pequeño, donde la comunicación sea más factible que en grupos de desarrollo grandes. La comunicación es un punto importante y debe realizarse entre los programadores, los jefes de proyecto y los clientes. (Borja 2018) (Salazar et al. 2018)

Para guiar el desarrollo de la presente investigación se propone XP como metodología a utilizar debido a que da lugar a una programación sumamente organizada que ocasiona eficiencias en el proceso de planificación y pruebas. Además, cuenta con una tasa de errores muy pequeña lo que propicia la satisfacción del programador. (Salazar et al. 2018)

En la figura 6 se puede apreciar las fases de esta metodología, además de los artefactos que propone.

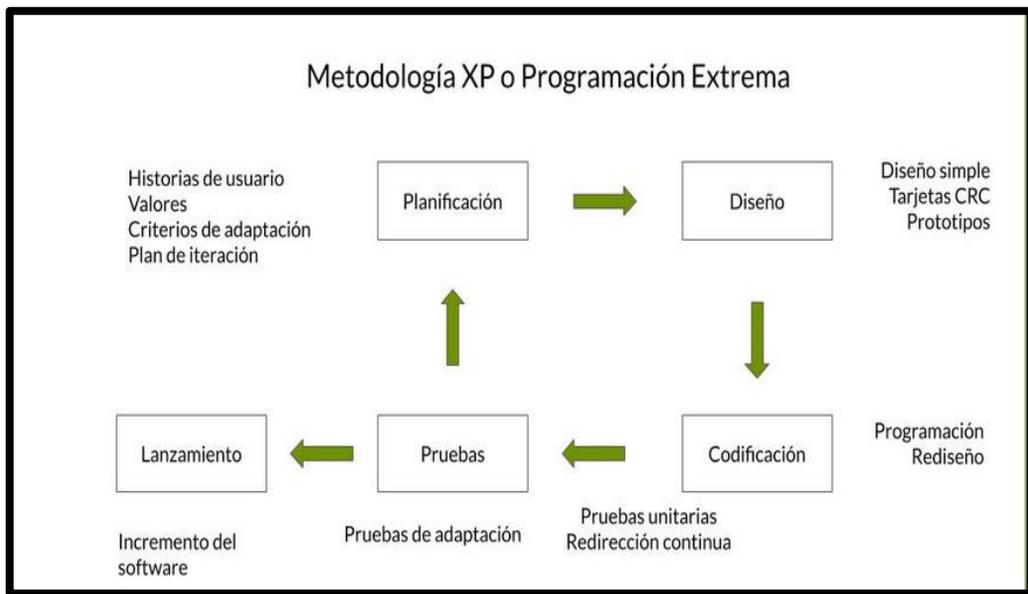


Figura 6: Metodología XP (León 2022)

Conclusiones parciales:

- El análisis de las definiciones abordadas en este capítulo, constituyen la base teórica para el desarrollo de la solución propuesta.
- El estudio de los simuladores quirúrgicos existentes, permitió identificar elementos semejantes que podrían añadirse en el desarrollo de la solución a la problemática planteada.
- El estudio de las metodologías y tecnologías utilizadas en el de diversos simuladores y software permitió seleccionar las más adecuadas a las características de la solución propuesta.

Capítulo 2: Diseño de la propuesta solución

Introducción

En el presente capítulo se describen los elementos fundamentales teniendo en cuenta el desarrollo de la solución propuesta. En el capítulo anterior quedo definida XP como metodología para el desarrollo de la solución. El uso de esta metodología en la investigación, se evidencia en la organización del trabajo en función de las fases que la misma define, además se hace referencia también a los artefactos correspondiente a cada una de ellas, entre los que destacan Historias de Usuario (HU) y Plan de iteraciones. En este capítulo quedará definido los patrones de diseño y la arquitectura que permitirá entender mejor la estructura de la solución.

2.1 Descripción de la propuesta de solución

Para dar cumplimiento al objetivo planteado se determina realizar complementos para la personalización de un simulador de cirugía autoadaptativo que podrá lanzar alertas, a partir de las ejecuciones históricas de las simulaciones del usuario almacenando información de los errores cometidos, en el simulador. Al estar centrado en el modelo de usuario, específicamente, en las variables relacionadas con las habilidades y conocimiento, que son: interacción con el contenido, desempeño de tareas, memorización de contenidos y tiempo de ejecución de tareas. Al trabajarse sobre el simulador antes descrito la propuesta solución esta estrechamente vinculada a este por lo que se deben utilizar muchas de las clases y escenas que están presentes en el. En este simulador el usuario tendrá un perfil relacionado al nombre de usuario, además debe poder seleccionar su perfil a través de una interfaz en la escena "Settings" del simulador donde se encuentran los personajes a seleccionar, luego pasa a otra interfaz de decisión y luego pasa a la de ajustes. Después pasa a la escena "RoomExc" donde se elige realizar la operación y cambia a la escena "Main" donde al comienzo de esta escena se cargan los personajes en sus simulaciones actual, la cual esta vacía ya que esta recién creada para iniciar la simulación, y anterior ,donde se encuentran

Capítulo 2: Diseño de la propuesta solución

los errores cometidos en esa simulación anterior, que serán notificados a modo de alertas mientras se ejecuta la simulación actual en la cual se detectarán los posibles errores, entre los cuales se encuentran: utilización de una herramienta inadecuada a la tarea a realizar en el paso actual, al igual que soltar un objeto fuera del lugar donde le corresponde, y demás variables relacionadas con las variables del modelo de usuario, como el nivel de memorización que se calcula de la siguiente forma: cantidad de errores cometidos en la simulación actual entre 128(pasos totales de la simulación) por 100 lo que da como resultado el tanto por ciento de la memorización de los pasos de la simulación; en el proceso quirúrgico desarrollado. También se debe exportar un archivo Json, que no es más que un conjunto de variables serializadas que se almacenan en un string el cual pasa a ser un archivo con formato “.data” el cual contiene dichas variables escritas en una cadena encerrada entre llaves, las asignaciones estas identificadas con “:” y seguidas de su valor y separadas por comas cada una de estas como por ejemplo el archivo “exportable.data” que contiene el nombre del usuario, la cantidad de errores cometidos en la simulación actual y el tiempo de ejecución de la operación; esto se carga en el correspondiente motor de adaptabilidad creado en el proyecto general, la información necesaria para obtener un archivo importable que contenga la puntuación que daría respuesta para identificar el nivel de aprendizaje y la evaluación al culminar el ejercicio. Luego en la escena “Punctuation” se mostraría la interfaz que cuenta con la puntuación alcanzada y el nivel de aprendizaje en el que se encuentra el usuario al finalizar la simulación. La (Figura 7) muestra la relación de los componentes necesarios para dar solución a la propuesta planteada.

Capítulo 2: Diseño de la propuesta solución

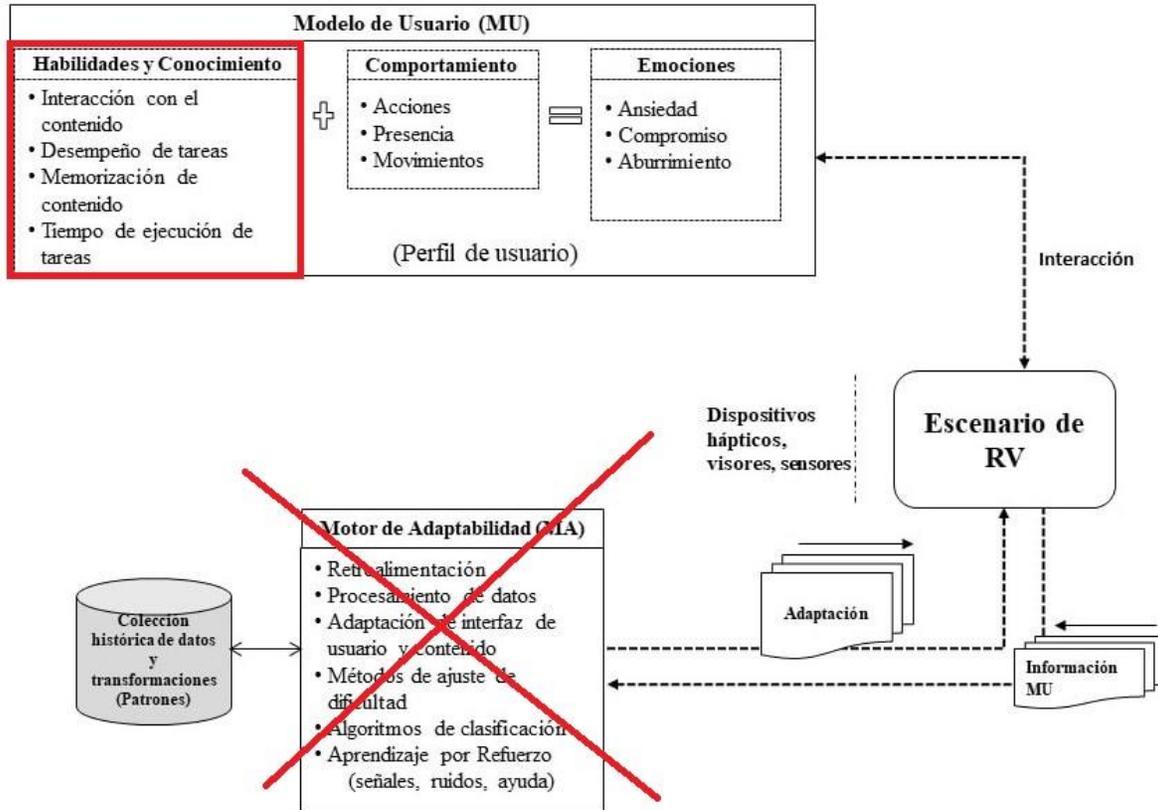


Figura 7: componentes del simulador de RV (Véliz Vega, Correa Madrigal, y Kugurakova 2021)

2.2 Levantamiento de requisitos

El levantamiento de requerimientos es una de las etapas iniciales y primordiales para el desarrollo de un sistema de información que reflejan la necesidad de los clientes de un sistema que ayude a resolver algún problema.

Requisitos funcionales

1. El simulador debe permitir seleccionar un perfil de usuario.
2. El simulador debe contar con las variables errores y tiempo.
3. El simulador debe tener una interfaz de usuario.
4. El simulador debe permitir el envío de Json al servidor externo.
5. El simulador debe contar con una interfaz que muestre la puntuación y el nivel de aprendizaje del usuario.

Requisitos no funcionales

Usabilidad:

- La interfaz debe ser intuitiva y garantice la comodidad.
- Permitir el acceso rápido a la información por parte de los usuarios y la fácil adaptación de usuarios sin experiencia.
- Adaptar el sistema al lenguaje y términos utilizados por los usuarios en la rama abordada con vista a una mayor comprensión del simulador.

Escalabilidad:

- El sistema debe ser extensible y facilitar el añadido de nuevas características.
- Su integración a otros simuladores debe ser lo mejor posible.

Soporte:

- El sistema debe ser mejorado, así como tener incorporado la capacidad de tener asesoramiento en cualquier error que tenga.

Restricciones del diseño y la implementación:

- El sistema debe ser desarrollado con las herramientas definidas anteriormente para el desarrollo inicial del simulador.

Hardware:

- El sistema debe tener compatibilidad con dispositivos de HMD con visión estereoscópica.

2.3 Fase de planeación

La metodología XP plantea la planificación como un dialogo continuo entre las partes involucradas en el proyecto. Un proyecto comienza recopilando historias de usuarios (HU), las que sustituyen a los tradicionales casos de uso(Borja 2018). Los planes en XP se diferencian de las metodologías tradicionales en la simplici-

Capítulo 2: Diseño de la propuesta solución

dad del plan y que los planes son realizados por las mismas personas que realizarán el trabajo.

Las HU son las sustitutas de los documentos de especificación funcional, y los casos de uso. Son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. Este artefacto debe poder ser programado en un tiempo entre una y tres semanas. Si la estimación es superior a tres semanas, debe ser dividida en dos o más historias. Si es menos de una semana, se debe combinar con otra historia. (Guevara y Palomino 2018)

A continuación, se muestran algunas de las historias de usuarios de la presente investigación el resto se encuentra en el Anexo 1.

Historia de usuario	
No 1: Seleccionar un perfil de usuario.	
Prioridad: alta	Nivel de complejidad: alto
Estimación: 1 día	Iteración asignada: 1
Descripción: Se selecciona un perfil donde se encontrarían los diferentes usuarios que pueden participar en la simulación.	
Observaciones:	

Tabla 2: Seleccionar un perfil de usuario

Historia de usuario	
No 2: Crear las variables errores y tiempo.	
Prioridad: alta	Nivel de complejidad: alto
Estimación: 1 día	Iteración asignada: 2

Capítulo 2: Diseño de la propuesta solución

Descripción: Se crean las variables cantidad de errores, errores, tiempo y nombre encargadas de brindar la información, más adelante, al sistema adaptativo de lógica difusa.

Observaciones:

Tabla 3: Crear las variables errores y tiempo

2.3.1 Plan de iteraciones

Las historias de usuarios obtenidas son desarrolladas y probadas en un ciclo de iteración, de acuerdo al orden preestablecido. Al comienzo de cada ciclo, se realiza una reunión de planificación de la iteración. Cada historia de usuario se traduce en tareas específicas de programación. Asimismo, para cada historia de usuario se establecen las pruebas de aceptación. Estas pruebas se realizan al final del ciclo en el que se desarrollan, pero también al final de cada uno de los ciclos siguientes, para verificar que subsiguientes iteraciones no han afectado a las anteriores. (Borja 2018)

La siguiente tabla muestra el plan de iteraciones establecido donde se hizo uso de la técnica de estimación del esfuerzo por puntos de HU. Los puntos de historia son una de las métricas más relevantes para la medición del trabajo realizado, dado que están relacionados directamente con las funcionalidades a ser desarrolladas y no solo con el tiempo de desarrollo requerido. (Méndez y Diego 2018)

Iteraciones	Historia de usuario a implementar	Tiempo de trabajo (Días)
Iteración 1	Seleccionar un perfil de usuario.	1
Iteración 2	Crear las variables errores y tiempo. Envío de Json a servidor externo.	3

Capítulo 2: Diseño de la propuesta solución

Iteración 3	Crear una interfaz de usuario. Interfaz de puntuación y nivel de aprendizaje del usuario.	2
--------------------	--	---

Tabla 4: Plan de Iteraciones

2.3.2 Plan de entrega

El cronograma de entregas establece qué historias de usuario serán agrupadas para conformar una entrega, y el orden de las mismas. Típicamente el cliente es el encargado de ordenar y agrupar según sus prioridades las historias de usuario (R. G. Á. Hernández y Pérez 2018). El cronograma de entregas que a continuación se muestra se realizó en base a las estimaciones de tiempos de desarrollo.

Producto	1ra Iteración	2da Iteración	3ra Iteración
Componentes para simulador de apendicectomía autoadaptativo basado en realidad virtual	Versión beta del producto	Versión alfa del producto	Versión final del producto
Fecha de inicio	20/6/2022	21/6/2022	27/7/2022
Fecha de fin	21/6/2022	24/6/2022	28/7/2022

Tabla 5: Plan de entrega

2.4 Fase de diseño

En esta fase, los requerimientos se transforman en un diseño o modelo sobre el cual se construirá el sistema. El resultado del diseño es la identificación de las partes del sistema y la estructura arquitectónica que satisface las necesidades que vienen desde los requerimientos funcionales y no funcionales, que ayudarán a tener una idea más clara en la fase de codificación. (Montero, Cevallos, y Cuesta 2018)

Capítulo 2: Diseño de la propuesta solución

2.4.1 Descripción de la arquitectura

Al igual que con diversos términos en ingeniería de software, no existe una definición universal del concepto de arquitectura de software. Sin embargo, el Instituto de Ingeniería de Software propone la definición siguiente que tiende a ser aceptada. (Maceda y Elizondo 2018)

“La arquitectura de software de un sistema es el conjunto de estructuras necesarias para razonar sobre el sistema. Comprende elementos de software, relaciones entre ellos, y propiedades de ambos.”

Para el desarrollo de la solución propuesta se utilizará el patrón arquitectónico por capas. Este patrón se centra en una distribución jerárquica de las roles y responsabilidades proporcionando una separación efectiva de las preocupaciones es decir cada rol se encarga de lo que le corresponde. Aterrizando en la investigación, se defino dos capas como se muestra en la figura 8.A continuación se describe este patrón arquitectónico.

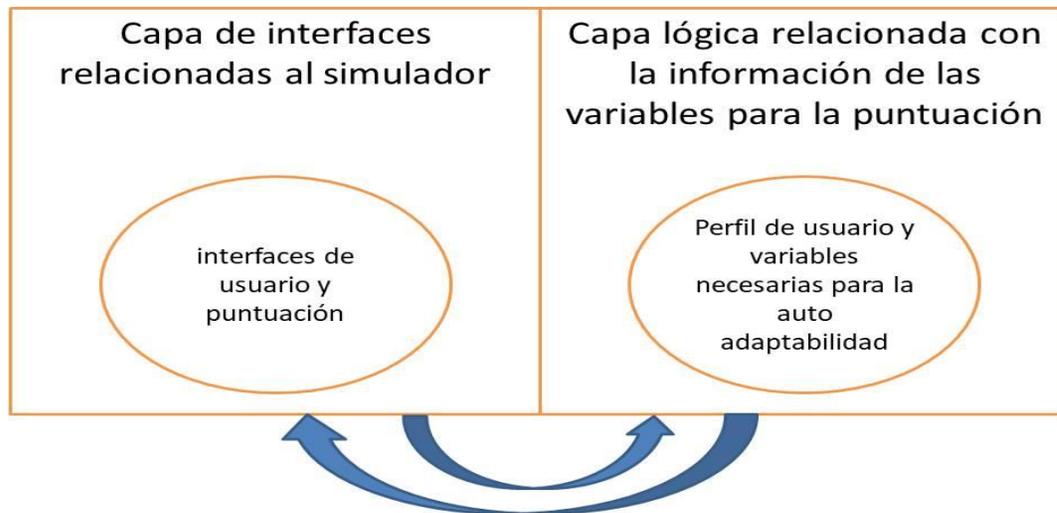


Figura 8: Arquitectura de software por capas: elaboración propia

Capítulo 2: Diseño de la propuesta solución

Capa de interfaces relacionadas con el simulador: En esta capa se visualizan las interfaces con las que el usuario puede interactuar, aquí se encuentran el paquete que contiene las interfaces de la solución. Estas tienen dependencia del paquete de componentes de UI (User Interface) de Unity.

Capa lógica relacionada con la información de las variables para la puntuación: En esta capa se encuentra la lógica de cada componente de la solución relacionados con la información necesaria para obtener la puntuación alcanzada al finalizar la cirugía.

2.4.2 Patrones de diseño

Los patrones de diseño son micro arquitecturas o módulos de construcción de alto nivel. Un programa construido bajo esta arquitectura es probable que exhiba buenas propiedades como modularidad, separación de tareas y mantenibilidad. La idea principal detrás de los patrones de diseño es apoyar la reutilización de la información de diseño, lo que permite a los desarrolladores plantear bases sólidas en la arquitectura del software a construir, minimizando la probabilidad de fracaso del desarrollo del proyecto. (Pressman 2010)

A continuación, se describen los patrones GRASP (*General Responsibility Assignment Software Patterns*, por sus siglas en inglés) y GoF (*Gang of Four*, por sus siglas en inglés) que se utilizarán en la solución.

Patrones de diseño GRASP

Experto: se pone en práctica con el uso de clases que poseen responsabilidades específicas a cumplir, de acuerdo con la información que manejan. El uso de este patrón se evidencia en la clase `VRTK_SDKManager`.

Creador: se refleja en las clases que tienen la responsabilidad de instanciar objetos de otras clases. El uso de este patrón se evidencia en la clase `Persona`.

Patrones de diseño GOF

Capítulo 2: Diseño de la propuesta solución

Mediador: es un patrón de diseño de comportamiento que te permite reducir las dependencias caóticas entre objetos. El patrón restringe las comunicaciones directas entre los objetos, forzándolos a colaborar únicamente a través de un objeto mediador (Shvets 2021). Este patrón es utilizado en la clase MonoBehaviour, la cual contiene todos los métodos y funcionalidades que permiten la relación entre todos los objetos o clases.

2.4.3 Descripción de las tarjetas CRC

Las tarjetas CRC resumen el significado de una clase durante la asignación de responsabilidades a las clases, antes de elaborar diagramas para estructurar el conjunto de clases. Son útiles para simular escenarios, aunque no como documentación definitiva y permanente (Pressman 2010). A continuación, se detalla algunas de las tarjetas CRC de las clases identificadas el resto se encuentra en [Anexo 2](#).

Tarjeta CRC	
Clase: VRTK_SDKManager	
Responsabilidad	Clases relacionada
<ul style="list-style-type: none">- Se encarga de configurar todo para utilizar los distintos SDK soportados.- permite cambiar el SDK Setup utilizado en tiempo de ejecución.- Una clase de ayuda que simplemente contiene referencias a la forma como a la información del método sobre el que se define el atributo.	<ul style="list-style-type: none">- MonoBehaviour- SDKSetup- Script(A)LeftController- Script(A)RightController

Tabla 6: Tarjeta CRC VRTK_SDKManager

Capítulo 2: Diseño de la propuesta solución

Tarjeta CRC	
Clase: VRTK_SnapDropZone	
Responsabilidad	Clases relacionada
<ul style="list-style-type: none">- Esto establece una zona predefinida donde un objeto interactivo existente puede ser soltado y al soltarlo se ajusta a la posición, rotación y escala de la transformación de la zona de caída.- es un objeto de juego que contiene un colisionador de disparo por defecto (Sphere Collider) y un cuerpo rígido cinemático para asegurar que se produzcan colisiones.	<ul style="list-style-type: none">- MonoBehaviour- VRTK_SDKManager- VRTK_InteractiveObject- ToolsCollisionsManager

Tabla 7: Tarjeta CRC VRTK_SnapDropZone

2.4.4 Estándares de Codificación

- El estándar de codificación empleado en el desarrollo de la solución fue definido por el equipo de desarrollo de la misma. A continuación, se muestran algunas características de dicho estándar.
- Las clases comenzarán con mayúscula y en caso que sean más de una palabra serán escritas continuas y cada letra inicial en mayúscula. Ejemplo: `public class PersonajeScript.`
- Las variables se escribirán siempre con la primera palabra en minúscula y en caso de que sean necesarias dos o más palabras serán escritas continuas e iniciarán con mayúscula. Ejemplo: `Bool ultimoEntrenamiento.`
- Los métodos al igual que las clases siempre se escribirán con mayúscula y en caso de usar más de una palabra, estas deben estar unidas y cada inicio de palabra con mayúscula. Ejemplo: `void PersonaEntrenada.`

Conclusiones parciales

- El empleo de la metodología XP en función de dirigir el proceso de desarrollo permitió realizar un trabajo organizado y estructurado, generando los artefactos de las fases de planeación y diseño.
- El uso de una arquitectura por capas y el empleo de patrones de diseño, garantizan obtener una solución con poca dependencia entre clases y fácilmente escalable.

Capítulo 3. Implementación y prueba

Introducción:

En el presente capítulo se describe la fase de codificación y prueba de la metodología de software empleada, en la cual se les realizaron pruebas a los componentes del simulador desarrollado, antes de ser entregado al cliente. En el capítulo también se realiza una validación de los resultados del simulador en relación al cumplimiento de objetivo general de la investigación.

3.1 Fase de codificación

En esta fase la programación se realiza a dos manos, es decir en parejas en frente del mismo ordenador. Incluso, a veces se intercambian las parejas. De esta forma se asegura que se realice un código más universal, con el que cualquier otro programador podría trabajar y entender. Y es que deber parecer que ha sido realizado por una única persona. Así se conseguirá una programación organizada y planificada. (León 2022)

A continuación, se muestran las 3 iteraciones planificadas anteriormente con sus respectivas tareas de ingeniería.

3.1.1 Primera iteración

En esta iteración se implementará la HU crear perfil de usuario, que permite obtener la información de los usuarios que realizan la actividad de simulación. A continuación, se muestra la tarea de ingeniería correspondiente a esta HU.

Tarea	
Número de tarea: 1	Número de HU: 1
Nombre de la tarea: Seleccionar un perfil de usuario	
Tipo de tarea: Implementación	Estimación: 1
Descripción: se implementó el script con las características del usuario y los métodos CargarPersonaje() y GuardarPersonaje() para cargar los datos del personaje de una escena a otra.	

--

Tabla 8: Crear perfil de usuario

3.1.2 Segunda iteración

En esta iteración se implementará las HU crear variables de error y tiempo y envío de Json al servidor externo. A continuación, se muestra una de las tareas de ingeniería correspondiente a estas HU el resto está en el [Anexo 3](#).

Tarea	
Número de tarea: 2	Número de HU: 2
Nombre de la tarea: Crear variable de error y tiempo	
Tipo de tarea: Implementación	Estimación: 1
Descripción: Se implemento las variables de error y tiempo estas se deben ir incrementando a medida que se cometa algún error en el ejercicio de simulación	

Tabla 9: TI. Crear variable de error y tiempo

3.1.3 Tercera iteración

En esta iteración se implementará las HU 3 y 5 relacionadas con crear interfaz de usuario e interfaz de puntuación y nivel de aprendizaje. A continuación, se muestra una de las tareas de ingeniería correspondiente a estas HU el resto en el [Anexo 3](#).

Tarea	
Número de tarea: 4	Número de HU: 3
Nombre de la tarea: Crear interfaz de usuario	
Tipo de tarea: Diseño	Estimación: 1
Descripción: Se diseño una interfaz con botones que sugieren el personaje con el cual iniciar el ejercicio de simulador.	

Tabla 10: TI. Crear interfaz de usuario

3.2 Fase de prueba

El uso de pruebas para comprobar las funcionalidades y por tanto los códigos de estas constituye uno de los pilares de la metodología definida en el capítulo 1. La metodología XP divide a las pruebas en dos grupos. Las pruebas unitarias diseñadas por los programadores, son las encargadas de verificar el código y la existencia de posibles errores. Además de las pruebas unitarias XP propone la realización de pruebas de aceptación, estas son especificadas por el cliente y se realizan con el fin de evaluar si al final de cada iteración se obtuvo la funcionalidad requerida, además de comprobar que dicha funcionalidad sea la esperada por el cliente final. Seguidamente se describen las pruebas de aceptación propuestas por XP.(Pressman 2010)

3.2.1 Pruebas de Aceptación

Las pruebas de aceptación se centran en las características y funcionalidades generales del sistema que son visibles y revisables por parte del cliente. Las pruebas de aceptación se derivan de las historias de usuario. Los clientes se encargan de escribir las pruebas funcionales para cada historia de usuario que deba validarse. (Borja 2018)

A continuación, se muestra un caso de prueba de aceptación para la HU 1. El resto de los casos de pruebas se encuentran disponibles en el [Anexo 4](#).

Caso de Prueba de Aceptación	
Código: P1HU1	Número de HU: 1
Nombre: Seleccionar un perfil de usuario	
Descripción: Se debe probar que se guardo correctamente la información correspondiente a un usuario	
Condiciones de ejecución:	
1. Se debe haber seleccionado un usuario de la interfaz de usuario	

Capítulo 3: Implementación y prueba

Entrada/Pasos de ejecución:
1. Se selecciona un usuario de la interfaz
Resultado esperado: al presionar el usuario correctamente se dirige a la pantalla de configuraciones
Evaluación de la prueba: Resultado satisfactorio.

Tabla 11: Crear perfil de usuario

3.3 Resultados del proceso de pruebas

Durante cada iteración se encontraron no conformidades significativas y no significativa. La mayor parte de las no conformidades están asociadas a problemas de interfaz y funcionales. Se identificaron un total de 7 no conformidades relacionadas con los siguientes errores, mal funcionamiento de los botones de la interfaz de usuario, mal conteo de la variable error y tiempo, además que la interfaz de usuario no coincidía con las interfaces propias del simulador. La tabla 12 muestra las no conformidades detectadas y cuales fueron solucionadas.

Elemento	No	No conformidad	Significativa	No Significativa	Recomendación	Estado NC	Respuesta del equipo de desarrollo
Interfaz de usuario	1	No coincide la interfaz implementada con la pantalla de fondo porque los colores utilizados no son visibles para el usuario		X		Resuelta	se está conforme con el error encontrado porque no es tan significativo para el autor de la investigación y no aporta un gran cambio al desarrollo de la funcionalidad de registrar el usuario
Botones del	2	Al presionar un botón referente a	X			Resuelta	Se encontró el error, se revisó el código y se

Capítulo 3: Implementación y prueba

perfil de usuario		un usuario no se dirigía a la pantalla de configuraciones					solucionó
Variable error	3	Al realizar el ejercicio de simulación existen posibles errores que no se tomaba en cuenta	X			Resuelta	Se reviso el código solucionándose este error
Variable tiempo	4	El simulador permite realizar el ejercicio sin medir el tiempo para ello. Al realizar el ejercicio de simulación la variable tiempo no se actualiza	X			Resuelta	Se reviso el código solucionándose este error
Perfil usuario	5	El simulador en la interfaz de usuario no permite crear un perfil de usuario nuevo. El ejercicio de simulación solo se puede realizar con los perfiles por defecto que trae el simulador		X	Añadir la funcionalidad que permita crear un perfil de usuario nuevo	Pendiente	Se detecto el error y se trabaja para solucionarlo
Interfaz de puntuación y nivel de aprendizaje	6	La interfaz de puntuación no muestra la puntuación obtenida en el ejercicio de simulación	X			Resuelta	Se encontró el error, se revisó el código y se solucionó
Interfaz	7	La interfaz de	X			Resuelta	Se encontró el error, se revisó el código y se solucionó

Capítulo 3: Implementación y prueba

de puntuaci ón y nivel de aprendi zaje	puntuación no muestra el nivel de aprendizaje alcanzado por el usuario en su ejercicio de simulación					
---	--	--	--	--	--	--

Tabla 12: Errores detectados

Las pruebas además de verificar la codificación eficiente, garantiza la calidad de la aplicación. Como se puede observar en la figura 9 los errores más detectados fueron los funcionales, que fueron solucionados para brindarle al cliente un producto lo más semejante posible a lo solicitado.

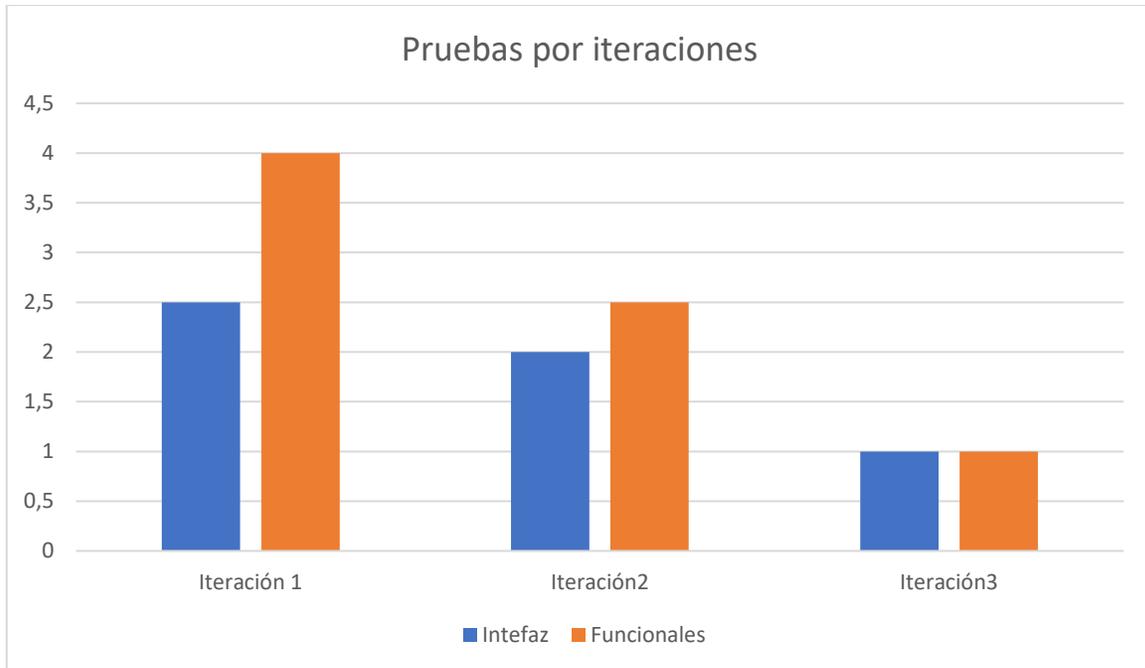


Figura 9: Iteraciones de pruebas

La tabla 12 muestra las no conformidades encontradas, como se puede apreciar en la figura 9 y la tabla 12 la NC de permitir crear un perfil de usuario nuevo es no significativa ya que no es imprescindible para el cliente ni el autor de la investigación.

Conclusiones Parciales

Al concluir este capítulo, se puede arribar a las siguientes conclusiones parciales:

- La realización de las tareas de ingeniería permitió dar cumplimiento a las historias de usuario definidas en la presente investigación.
- La aplicación de las pruebas de aceptación propuestas por la metodología XP, además de las funcionales realizadas permitió verificar el correcto funcionamiento del código y validar las funcionalidades que garantizan el cumplimiento del objetivo de la investigación.

Conclusiones Generales

A partir del desarrollo de la presente investigación se concluye:

- El análisis realizado del estado del arte fundamentó la necesidad de llevar a cabo el desarrollo de un simulador autoadaptativo que garantice la realización de rutinas individuales a los usuarios.
- El uso de la metodología XP como guía del proceso de desarrollo de software, así como las herramientas y tecnologías tales como el motor de juego Unity, el lenguaje de programación C# y el editor de texto Visual Studio Code, facilitó el correcto desarrollo de la propuesta solución.
- Los requisitos del sistema, la arquitectura por capas y el uso de los patrones de diseño, permitieron obtener más óptimo de la aplicación.
- La realización de las pruebas funcionales y de aceptación demostró que la implementación satisface los requisitos propuestos por el cliente, lo cual da cumplimiento al objetivo trazado en la presente investigación.

Recomendaciones

Teniendo en cuenta los resultados obtenidos en la presente se recomienda:

- Permitir que los usuarios se puedan crear un perfil de usuario nuevo.
- Aumentar el número de errores que se pueden cometer en el ejercicio de simulación.

Glosario de Términos

autoadaptabilidad: referente a los sistemas informáticos, constituye la capacidad para ajustarse automáticamente al contexto de ejecución en un ciclo cerrado de retroalimentación.

gamificación: concepto referido a la transformación de los procesos rutinarios en juegos.

incisión: Corte hecho en un cuerpo o una superficie con un instrumento cortante o agudo

inmersión: fenómeno asociado a la capacidad humana de sentirse parte de un entorno virtual.

realidad virtual: recreación de entornos u objetos como una imagen compleja, generada por computadora.

retroalimentación háptica: forma para que la tecnología interactúe con el tacto, generalmente por vibración, en lugar de la vista o el sonido, aunque las vibraciones pueden producir un sonido audible.

quirófano: Sala de un establecimiento hospitalario especialmente acondicionada para realizar operaciones quirúrgicas.

Referencias Bibliográficas

1. Aeschlimann, Martin. 2018. «Visual Studio Code». Eclipse Demo Camp, Zürich.<https://code.visualstudio.com/>.
2. Ali, Roshan, Muhammad Anwar, y Jamshed Akhtar. 2017. «Laparoscopic Versus Open Appendectomy In Children: A Randomized Controlled Trial fom a Developing Country». 10.1016/j.jpedsurg.2017.11.022.
3. Alvares, Alberto Cuevas. 2020. *Comenzar a programar en Python*.
4. Aramendi. 2019. «ANEURISMA ARTERIAL».
5. Barbier, Giacomo, Carlos Francisco Rodríguez Herrera, y Carlos Francisco Rodríguez. 2020. «Algoritmo de corte y control de fuerza para un proceso de corte de láminas implementado por un robot».
6. Baumgarten, Heron D., Erin G. Brown, Kathryn W. Russell, N. Scott Adzick, y Pablo Laje. 2018. «Laparoscopic appendectomy — Outcomes of senior trainees operating without supervision versus experienced pediatric surgeons». <https://doi.org/10.1016/j.jpedsurg.2018.10.092>.
7. Bernardo, Antonio. 2018. «Virtual Reality and Simulation in Neurosurgical Training». <https://doi.org/10.1016/j.wneu>.
8. Borja, Yolanda López. 2018. «Metodología Ágil de Desarrollo de Software – XP».
9. Calderón, Sebastián Javier, Marisa Cecilia Tumino, y Juan Manuel Bournissen. 2020. «Realidad virtual: impacto en el aprendizaje percibido de estudiantes de Ciencias de la Salud», 2020. Tecnología, Ciencia y Educación.
10. Carranza, Alexandra. 2022. «que-es-java». crehana.com. 2022. <https://www.crehana.com/blog/desarrollo-web/que-es-java/>.
11. Casal, María del Carmen. 2018. «La simulación como metodología para el aprendizaje de habilidades no técnicas en Enfermería». Doctoral.
12. Cerna, Fernando Flores-, Víctor-Manuel Sanhueza- Salazar, Héctor-Moisés Valdés- González, y Lorenzo Reyes- Bozo. 2021. «Metodologías ágiles: un análisis de los desafíos organizacionales para su implementación», 2021. <http://revistas.udistrital.edu.co/ojs/index.php/revcie/index>.

Referencias Bibliográficas

13. Comas-González, Zhoe, Sabel Cheverri-Ocampo, Ronald Zamora-Musa, Jeimy Velez, Roman Sarmiento, y Martha Orellana. 2017. «Tendencias recientes de la Educación Virtual y su fuerte conexión con los Entornos Inmersivos», 2017.
14. Espinosa, Guillermo Cid. 2018. «Programación de interfaz gráfica en app designer para el control vectorial de motores de imanes permanentes».
15. Estévez, Sheila Plaza, y Nerea Ramírez Lamela. 2018. «API de servicios web orientados a accesibilidad».
16. Fernández Ruiz, Leonardo Rafael, y Lester Oscar Rodríguez González. 2017. «Técnica de Corte para Objetos Deformables».
17. García, Carlos de la Fuente. 2020. «Guía práctica para la publicación de Datos Abiertos usando APIs». <http://datos.gob.es/es/aviso-legal>.
18. González, Manuel Amaya-. 2018. «Diseño, simulación y control de un asistente robótico para cirugía laparoscópica». Universidad de Perú.
19. Guevara, Delicia Esmeralda Lucero, y Nora Bertha La Serna Palomino. 2018. «Automatización de requisitos: historias de usuario geNeradas a Partir de uN modelo orieNtado a objetivos». 10.26439/interfases2018.n011.2953.
20. Hernández, Alejandro López. 2022. «Qué son los Scripts y para qué sirven». GameDevTraum. 2022. <https://gamedevtraum.com/es/programacion-informatica/teoria-de-programacion/que-son-los-scripts-y-para-que-sirven-ejemplos-de-utilizacion-en-unity/>.
21. Hernández, Ramón Guillermo Álvarez, y Yamilis Fernández Pérez. 2018. «SoftQuality, sistema informático de ayuda a la toma de decisiones para la evaluación y selección de productos de software». Universidad de las Ciencias Informáticas.
22. Hudson, Sarah, Sheila Matson-Barkat, Nico Pallamin, y Guillaume Jegou. 2019. «Interaction and immersion in a virtual reality experience». <https://doi.org/10.1016/j.jbusres.2018.10.062>.
23. Inchausti, Cristina, Héctor e Cantele, Miguel Vassallo, Inés villegas, Alexis Sánchez, y Arlene Méndez. 2022. «programa de entrenamiento para la cirugía laparoscópica de acalasia en modelos inertes y orgánicos. estudio observacional».

Referencias Bibliográficas

24. León, Cesar. 2022. «Metodología XP o Programación Extrema». 2022. <https://www.sinnaps.com/blog-gestion-proyectos/metodologia-xp>.
25. Lozano, María del Carmen Morillo, y González. 2018. «Adaptive Learning».
26. M. Bachen, Christine, Pedro Ramos, Chad Raphael, y Amanda Waldron. 2017. «How do presence, flow, and character identification affect players' empathy and interest in learning from a serious computer game?»
27. M. del C, Morillo Lozano. 2018. «Aprendizaje adaptativo». 2018. Morillo Lozano, M. del C.
28. Maceda, Humberto Cervantes, y Perla Velasco Elizondo. 2018. «Arquitectura de software .Coceptos y ciclo de desarrollo.»
29. Malhotra, Lavina, Elizabeth M. Pontarelli, Gary G. Grinberg, Richard S. Isaacs, y James P. Morris. 2020. «Cost analysis of laparoscopic appendectomy in a large integrated healthcare system». <https://doi.org/10.1007/s00464-020-08266-0>.
30. Méndez, Erwin R., y Marta Fernández Diego. 2018. «estimación de esfuerzo en proyectos de desarrollo de software con metodologías ágiles».
31. Montero, Bryan Molina, Harry Vite Cevallos, y Jefferson Dávila Cuesta. 2018. «Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software», 2018. Espirales revista multidisciplinaria de investigación ISSN: 2550-6862.
32. Muñoz, Miguel. 2018. «introducción a C#». Primera edición. <https://ticapacitacion.com>.
33. Pallets. 2022. «Flask Documentation».
34. Parra, Juan Sebastián Martínez, y Ángel Alberto García. 2020. «Simulación quirúrgica como herramienta de aprendizaje y seguridad del paciente. Revisión de la literatura».
35. Pérez, Luis. 2018. «MonoBehaviour». Unity Technologies. 2018. <https://docs.unity3d.com/es/530/ScriptReference/MonoBehaviour.html>.
36. Pressman, Roger.S. 2010. *Ingeniería de software un enfoque practico*. 7 ma.
37. Quemada, Juan. 2019. «Gestión de Proyectos con Git». <http://git-scm.org/book/>.
38. Reitz, Kenneth. 2022a. «Scrapy Documentation». 2022b. «NumPy User Guide». 2022c. «Requests Documentation».

Referencias Bibliográficas

39. Reyes, Nora Tatiana Quesada, y José Luis Andrade Pineda. 2020. «Estudio sobre Metodologías Ágiles en los Proyectos Software. Propuesta de Plan de Implantación para Pymes».
40. Rodríguez, Marcelo. 2022. «El proyecto que se basa en tecnología robótica y realidad virtual está a punto de ser patentado». *El universal Ciudad de México*, 2022.
41. Rosales, Jorge. 2022. «Qué es Unity y para qué sirve». 2022. <https://www.masterd.es/blog/que-es-unity-3d-tutorial>.
42. Salazar, Juan Camilo, Álvaro Tovar, Juan Carlos Linares, Alexander Lozano, y Lizeth Valbuena. 2018. «Scrum versus XP: similitudes y diferencias». Publicación de la Facultad de Ingeniería Technol.Investig.Academia TIA. <https://revistas.udistrital.edu.co/ojs/index.php/tia>.
43. Sánchez, Isac. 2022. «bacsim-brain-advanced-computer-surgery-simulator». mexicanist. 2022. <https://www.mexicanist.com/l/bacsim-brain-advanced-computer-surgery-simulator/>.
44. Schwaber, Ken, y Sutherland Jeff. 2018. «La Guía Definitiva de Scrum: Las Reglas del Juego».
45. Shvets, Alexander. 2021. «Sumérgete en los patrones de diseño». <https://refactoring.guru/es/design-patterns/book>.
46. Vázquez, Vázquez Daniel Eduardo. 2020. «El uso de simuladores de bajo costo por residentes de cirugía general, para el desarrollo de destrezas en cirugía laparoscópica básica en el hospital general de Atizapán Dr. Salvador González Herrejón en el periodo de mayo-agosto». Universidad autónoma del estado de México.
47. Vaughan, Neil, Bodgan Gabrys, y Venketesh N. Dubey. 2018. «An Overview of Self-Adaptive Technologies Within Virtual Reality Training».
48. Vega, Joel Ayala de la. 2019. «Lenguaje de Programación Estructurada». Universidad Autónoma del estado de México.
49. Véliz Vega, Alieski, Omar Correa Madrigal, y Vlada Kugurakova. 2021. «Aprendizaje adaptativo basado en Simuladores de Realidad Virtual». <http://rcci.uci.cu>.

Referencias Bibliográficas

50. Vicente, Benjumea, y Manuel Roldán. 2018. «Fundamentos de Programación con el Lenguaje de Programación C++». Lenguajes y Ciencias de la Computación Universidad de Málaga.
51. Wojtok, A. J, Martin Hirsch, y S Bab. 2018. «Adaptive learning in Virtual Reality: Current state and new approach. Retrieved», junio de 2018. <https://go-study-europe.de/wpcontent/uploads/2018/11/IRC2018-proceedings-final-web>.
52. Xin, Wang, y Wang Xiuyue. 2018. «Virtual Reality Training System for Surgical Anatomy». <https://doi.org/10.1145/3293663.3293670>.
53. Zahabi, M, Razak Abdul, y M A. 2020. «Adaptive virtual reality-based training: a systematic literature review and framework». *Virtual Reality*, 2020.

Anexo 1: Historias de Usuario

Historia de usuario	
No 3: Crear una interfaz de usuario.	
Prioridad: media	Nivel de complejidad: media
Estimación: 1 día	Iteración asignada: 3
Descripción: Debe ser una interfaz que permita seleccionar el usuario que realizará la actividad de simulación	
Observaciones:	

Tabla 13: Crear una interfaz de usuario

Historia de usuario	
No 4: Envío de Json a servidor externo.	
Prioridad: alta	Nivel de complejidad: alto
Estimación: 2 días	Iteración asignada: 2
Descripción: debe permitir exportar un archivo Json en el cual esta la información del usuario que acaba de completar la simulación	
Observaciones:	

Tabla 14 Envío de Json a servidor externo

Historia de usuario	
No 5: Interfaz de puntuación y nivel de aprendizaje del usuario.	
Prioridad: media	Nivel de complejidad: media
Estimación: 1 día	Iteración asignada: 3
Descripción: la interfaz debe permitir mostrar la puntuación y el nivel de aprendizaje del usuario que realizo la simulación	
Observaciones:	

Tabla 15: Interfaz de puntuación y nivel de aprendizaje del usuario

Anexo 2: Tarjetas CRC

Tarjeta CRC	
Clase: ToolsCollisionsManager	
Responsabilidad	Clases relacionada
<ul style="list-style-type: none"> - Es el encargado de detectar las colisiones de los instrumentos definidos por el simulador. 	<ul style="list-style-type: none"> - MonoBehaviour - VRTK_InteractableObject - PersonajeScript - VRTK_SnapDropZone

Tabla 16: Tarjeta CRC ToolsCollisionsManager

Tarjeta CRC	
Clase: PersonajeScript	
Responsabilidad	Clases relacionada
<ul style="list-style-type: none"> - Es el encargado de gestionar la información del usuario que esta realizando la operación. - Es el que contiene las variables necesarias para la puntuación. 	<ul style="list-style-type: none"> - MonoBehaviour - Timer - ExportJsonScript - ImportJsonScript - ToolsCollisionsManager

Tabla 17:Tarjeta CRC PersonajeScript

Tarjeta CRC	
Clase: Timer	
Responsabilidad	Clases relacionada
<ul style="list-style-type: none"> - Es el encargado de gestionar la información del tiempo del usuario que está realizando la operación. 	<ul style="list-style-type: none"> - MonoBehaviour - PersonajeScript

Tabla 18:Tarjeta CRC Timer

Tarjeta CRC	
Clase: ExportJsonScript	
Responsabilidad	Clases relacionada

<ul style="list-style-type: none"> - Es el encargado de enviar la información del usuario en forma de Json para que pueda interactuar con la API para generar la puntuación. 	<ul style="list-style-type: none"> - MonoBehaviour - PersonajeScript
---	--

Tabla 19: Tarjeta CRC ExportJsonScript

Tarjeta CRC	
Clase: ImportJsonScript	
Responsabilidad	Clases relacionada
<ul style="list-style-type: none"> - Es el encargado de cargar la información del usuario en forma de Json para que pueda interactuar con la interfaz de puntuación. 	<ul style="list-style-type: none"> - MonoBehaviour - PersonajeScript

Tabla 20: Tarjeta CRC ImportJsonScript

Anexo 3: Tareas de Ingeniería

Tarea	
Número de tarea: 3	Número de HU: 4
Nombre de la tarea: Envío de Json al servidor externo	
Tipo de tarea: Implementación	Estimación: 2
Descripción: Se implemento el método ExportarJson() para guardar los datos de del usuario que realizo la simulación	

Tabla 21: TI. Envío de Json al servidor externo

Tarea

Número de tarea: 5	Número de HU: 5
Nombre de la tarea: Interfaz de puntuación y nivel de aprendizaje	
Tipo de tarea: Diseño	Estimación: 1
Descripción: Se diseño una interfaz con los campos de números de errores, tiempo y puntuación obtenida en la simulación.	

Tabla 22:TI.Interfaz de puntuación y nivel de aprendizaje

Anexo 4: Pruebas de Aceptación

Caso de Prueba de Aceptación	
Código: P2HU2	Número de HU: 2
Nombre: Crear variable de error y tiempo	
Descripción: Se debe probar que se guarde y muestre la cantidad de errores cometidos, así como el tiempo empleado en el ejercicio de simulación	
Condiciones de ejecución:	
1 - el usuario debe haber realizado el ejercicio de simulación	
2 – el usuario debe haber cometido errores de los que se tiene identificado como error en el ejercicio de simulación	
Entrada/Pasos de ejecución:	
Resultado esperado: muestra el tiempo empleado en realizar la simulación y la cantidad de errores cometidos	
Evaluación de la prueba: Resultado satisfactorio.	

Tabla 23:Crear variable de error y tiempo

Caso de Prueba de Aceptación	
Código: P3HU3	Número de HU: 3
Nombre: Crear interfaz de usuario	
Descripción: Se debe probar que se muestra los usuarios con los cuales realizar la simulación y que permite realizar el ejercicio con el usuario	

seleccionado
Condiciones de ejecución:
Entrada/Pasos de ejecución:
Resultado esperado: se muestra una interfaz con los usuarios disponibles para realizar la simulación
Evaluación de la prueba: Resultado satisfactorio.

Tabla 24: Crear interfaz de usuario

Caso de Prueba de Aceptación	
Código: P4HU4	Número de HU: 4
Nombre: Envío de Json al servidor externo	
Descripción: Se debe probar que se exporto correctamente la información correspondiente a un usuario	
Condiciones de ejecución:	
1. el usuario debe haber terminado el ejercicio de simulación	
Entrada/Pasos de ejecución:	
Resultado esperado: se exporta correctamente los datos del usuario que realizo la simulación	
Evaluación de la prueba: Resultado satisfactorio.	

Tabla 25: Envío de Json al servidor externo

Caso de Prueba de Aceptación	
Código: P5HU5	Número de HU: 5
Nombre: Crear interfaz de puntuación y nivel de aprendizaje	
Descripción: Se debe probar que se muestra la puntuación y el nivel de aprendizaje obtenido en la simulación.	
Condiciones de ejecución:	
1- Se debe haber terminado de realizar la simulación	
Entrada/Pasos de ejecución:	
Se importa el Json con los datos que se guardaron del usuario y necesarios en la interfaz	

Resultado esperado: se muestra la puntuación y nivel de aprendizaje alcanzado en la simulación
Evaluación de la prueba: Resultado satisfactorio.

Tabla 26: Crear interfaz de puntuación y nivel de aprendizaje