TRABAJO FINAL DE GRADO

Sistema VR configurable para entrenamiento en enfermería

AUTOR
Jordi Nieto Maldonado

DIRECTOR
Carlos Andújar Gran

Especialidad de Computación
29 de Abril del 2020

Grado en
Ingeniería Informática

Universidad Politécnica
de Cataluña | BarcelonaTech

Facultad de Informática de Barcelona

Contenido

Índice de figuras

Índice de tablas

1. Resumen

En el proceso de aprendizaje de los profesionales del campo médico, resulta esencial la realización de sesiones prácticas en las que el estudiante pueda practicar las metodologías aprendidas aplicándolas en primera persona. La realización de estas sesiones están sujetas a la disponibilidad de los espacios y materiales adaptados para su ejecución, dificultando la cantidad de veces que los aprendices pueden repetir los procedimientos para interiorizarlos.

El objetivo del proyecto es el desarrollo de un sistema que permita el entrenamiento y la evaluación de personal de enfermería mediante la configuración de sesiones en un entorno virtual. El sistema permitirá definir y personalizar sesiones de entrenamiento (como por ejemplo preparar la mesa quirúrgica y entregar instrumental al cirujano), proporcionando una experiencia completamente inmersiva mediante el uso de gafas y controladores de Realidad Virtual.

En el procés d'aprenentatge dels professionals del camp mèdic, resulta essencial la realització de sessions pràctiques en què l'estudiant pugui practicar les metodologies apreses aplicant-les en primera persona. La realització d'aquestes sessions estan subjectes a la disponibilitat dels espais i material adaptats per la seva execució, dificultant la quantitat de vegades que els aprenents poden repetir els procediments per interioritzar-los.

L'objectiu del projecte és el desenvolupament d'un sistema que permeti l'entrenament i l'avaluació de personal d'infermeria mitjançant la configuració de sessions en un entorn virtual. El sistema permetrà definir i personalitzar sessions d'entrenament (com per exemple preparar la taula quirúrgica i entregar instrumental al cirurgià), proporcionant una experiència completament immersiva mitjançant l'ús d'ulleres i controladors de Realitat Virtual.

In the learning process of medical professionals, it is essential to carry out practical sessions in which the student can practice the methodologies learned by applying them in first person. The realization of these sessions is subject to the availability of the spaces and materials adapted for their execution, hindering the number of times that apprentices can repeat the procedures to internalize them.

The objective of the project is the development of a system that allows the training and evaluation of nursing personnel by configuring sessions in a virtual environment. The system will define and customize training sessions (such as preparing the surgical table and delivering instruments to the surgeon), providing a fully immersive experience through the use of Virtual Reality headsets and controllers.

2. Introducción

2.1. Sistemas de realidad virtual

Actualmente nos encontramos ante la era de la revolución tecnológica, una época en la que los avances en ciencia e ingeniería están siendo constantes, con nuevas metodologías y técnicas surgiendo casi a diario. Cada una de estas novedosas tecnologías aporta un nuevo mar de posibilidades y aplicaciones prácticas, que pueden realizar grandes mejoras a cualquiera de los procedimientos que nuestra sociedad realiza cada día.

Buena parte de estos avances se están realizando en el ámbito de la computación gráfica, con continuas mejoras en las tecnologías de renderizado y procesamiento de escenarios virtuales. Todos estos cambios han permitido el desarrollo de muchas tecnologías gráficas: desde videojuegos y consolas hasta películas enteras realizadas por computador, o incluso complejas aplicaciones que podemos utilizar a diario en nuestros dispositivos móviles. Aun así, cabe destacar que uno de los sectores que mayor beneficio está consiguiendo son las tecnologías de inmersión en entornos virtuales, entre ellas la *Realidad Virtual*, que está aprovechando todos los avances en tecnología *3D* y capacidad de cómputo para implementar sistemas que años atrás sólo formaban parte del cine de ciencia ficción. Y es que muchos de los límites tecnológicos con los que se encontraba este sector se han visto resueltos: desde el aumento en la capacidad de procesamiento gráfico y la resolución de las pantallas, hasta la variedad de tecnologías inalámbricas y de seguimiento del cuerpo en tiempo real; todo ello haciendo posible la invención de sistemas con elevadas prestaciones tecnológicas que sean económicamente accesibles para todo tipo de público.

Si bien la audiencia de este tipo de tecnología solía ser bastante concreta, con la comercialización de dispositivos aptos para todo el público se ha generado un aumento drástico en la oferta y las posibilidades del uso de esta tecnología en ámbitos que hasta el momento no se habían contemplado. Es cierto que seguramente haya gente a la que esta tecnología no le termine de convencer, ya sea por temas de espacio o por adaptabilidad personal, pero los nuevos sistemas que se están desarrollando son cada vez más precisos y portables, mostrando que este mercado todavía tiene mucho más que ofrecer. También cabe destacar que estos dispositivos están siendo publicitados sobre todo en el ámbito de los videojuegos, pero sus aplicaciones también son más variadas, siendo muy útiles en terapia psicológica para superar miedos [10], en fisioterapia para estimular la realización de ejercicios físicos [11] o en educación para realizar un aprendizaje práctico o incluso realizar clases online en una clase virtual [12].

Pero no todo había sido siempre así, de modo que para poder contrastar correctamente los avances en las tecnologías de simulación, primero deberemos echar un vistazo a los primeros sistemas satisfactorios de simulación y su evolución. Empezando por el ocio, se considera que *Sensorama*[13] fue el primer dispositivo de realidad virtual en proporcionar una vista estereoscópica de un objeto *3D*, consiguiendo incluso una experiencia multisensorial en algunas de sus pruebas [14]. Si bien es un hito remarcable, cabe destacar que las dimensiones de este dispositivo dificultaban su transporte, y la observación debía hacerse desde una posición concreta que se encontraba fijada en la estructura. No sería hasta la época de los años 90, que se conseguirían comercializar cascos de realidad virtual como los empezamos a conocer ahora: suficientemente ligeros como para llevarlos sin soportes auxiliares y con sistemas básicos de seguimiento. Estos dispositivos empezaron siendo utilizados en máquinas recreativas que mostraban escenarios virtuales simples, o como accesorios en algunas de las videoconsolas comercializadas por *SEGA* y *Nintendo* en aquella época [15, 16].

Figura 1: *El primer sistema VR para el ocio fue Sensorama, mientras que Sega VR fue el primero en comercializarse para consola. Fuentes: [1, 2]*

Ahora bien, si nos fijamos en el ámbito laboral, los sistemas de simulación tuvieron una evolución mucho más rápida y concreta. El interés por parte de grandes empresas en simular escenarios de trabajo reales en ámbitos tan importantes como la aviación, el ejército y la medicina [17] hizo que, desde el desarrollo de los primeros sistemas gráficos en los años 70, se realizara un gran esfuerzo en mejorar esta metodología de práctica. Gracias al interés en este tipo de tecnología, se consiguieron diseñar complejos sistemas de simulación de vuelo para entrenar a militares y pilotos [18], que permitían simular las diferentes tareas e imprevistos que se deberían solucionar en la realidad mientras el aprendiz se encontraba en un entorno seguro. Uno de los primeros sistemas en conseguir este hito fue el conocido como *Super Cockpit* [19], capaz de proyectar mapas *3D* generados por ordenador y que permitía la interacción a tiempo real por parte de los pilotos, incorporando un sistema de seguimiento de movimientos y de control aéreo. La utilización de estos sistemas de prueba permitió realizar una formación previa a los pilotos, que intentaba minimizar la probabilidad de errores humanos en un entorno real, evitando así los daños físicos que esto podría ocasionar y el gran impacto económico y social que supondría [20].

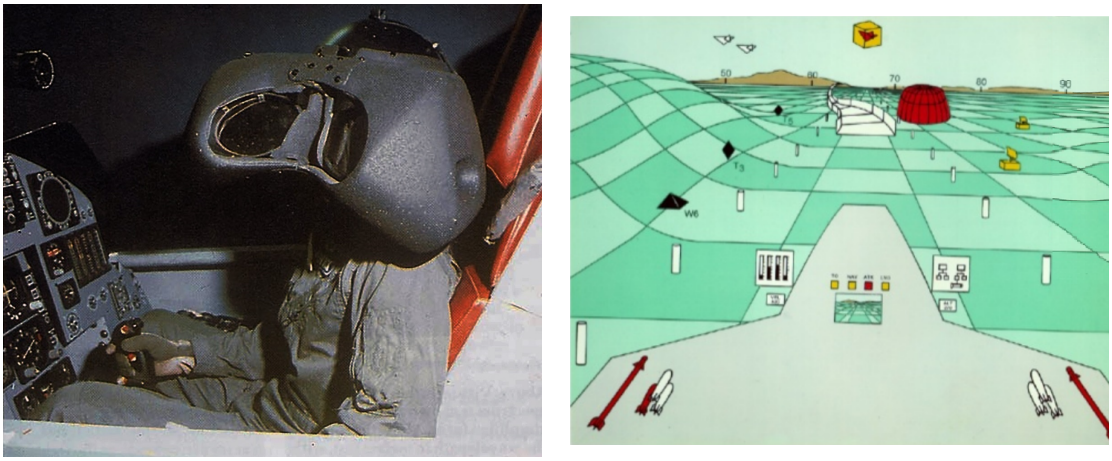


Figura 2: *Simulador de vuelo Super Cockpit. Fuente: [3]*

Si observamos más detenidamente las características de los sistemas de simulación virtual que se estuvieron desarrollando durante el inicio de la era gráfica, podemos notar que la mayoría de estos simuladores eran utilizados en trabajos que exclusivamente requerían el uso de volantes, *joysticks*, botones y palancas. Esto era, en buena parte, debido a la carencia tecnológica de hace unos años para realizar un control preciso en el movimiento completo de nuestro cuerpo [21].

Ahora, en pleno siglo *XXI* y con el gran avance en tecnología de Realidad Virtual, podemos realizar fácilmente un seguimiento del movimiento de cabeza y manos (tanto de posición como de orientación), lo que permite situar al usuario en primera persona y que sienta la experiencia como si fuera real. Este cambio abre un enorme abanico de oportunidades que hasta el momento eran impensables; y es que gracias a la aplicación de esta tecnología en el ámbito profesional, se están desarrollando sistemas precisos de simulación que evitan depender de espacios adaptados, disponer físicamente de herramientas específicas y tener que modificar los escenarios para cada simulación.

La simulación práctica de procedimientos mediante el uso de Realidad Virtual también se está empezando a aplicar en el ámbito de la medicina, donde podemos encontrar desde proyectos que permiten estudiar la anatomía hasta simulaciones completas de cirugías [22]. Varios estudios confirman que el uso de este sistema de práctica es una herramienta muy eficaz para ayudar en el proceso de aprendizaje, dadas evidencias de mejora en la rapidez, precisión y economía de movimientos de los profesionales médicos que lo han utilizado [23, 24]. Con resultados tan positivos, es altamente probable que dentro de unos años la enseñanza quirúrgica mediante Realidad Virtual sea tan importante y necesaria como lo son hoy en día los simuladores de vuelo en la aviación [25].

2.2. Contextualización

Este proyecto ha sido desarrollado en colaboración con el grupo de investigación *VIRVIG* (acrónimo de *Visualización, Realidad Virtual e Interacción Gráfica*) y el proyecto *3D4LIFE*, del que C. Andújar es investigador principal. Este centro de investigación está compuesto por investigadores tanto de la *Universidad Politécnica de Cataluña* como de la *Universidad de Girona*; se trata de una *Unidad de Investigación Conjunta* (JRU) de la *Comisión Europea* con el soporte y financiación del *Ministerio de Economía, Industria y Competitividad* del *Gobierno Español*.

Uno de los muchos objetivos del proyecto *3D4LIFE* enmarca la interacción *3D* para *Realidad Virtual* y *Realidad Aumentada* en la Industria 4.0 y las ciencias de la vida; más concretamente, se está estudiando y evaluando el uso de la Realidad Virtual para realizar entrenamientos colaborativos en las ciencias de la salud.

También cabe destacar la colaboración del personal médico y docente del *Hospital Sant Joan de Déu*, entre ellos especialmente la Dra. Esther Insa Calderón y la enfermera Isidora Andújar Tore (Dori), que han aportado sus conocimientos técnicos y su tiempo libre para guiar en la exactitud médica del proyecto. Gracias a la colaboración de estos expertos en el campo médico, hemos podido validar con exactitud técnica la especificación de las pruebas que realizaríamos. Ha sido muy importante la comunicación entre los expertos tecnológicos y los expertos del dominio, ya que el sistema debe adaptarse a las necesidades evaluativas de las pruebas que conocen los expertos del dominio, pero somos los expertos tecnológicos los que disponemos de los conocimientos para traducir estos procesos y que se hagan efectivos en un entorno virtual.

Mediante el desarrollo de este proyecto, deseamos resolver algunos de los problemas que se encuentra el personal médico a la hora de formarse, dado que tanto para practicar como para evaluarse, es necesario que los aprendices se encuentren físicamente en el centro de estudios y que los espacios y materiales de prácticas estén disponibles. Todos estos problemas se acentúan todavía más en los casos en los que el aprendiz está realizando los estudios a distancia o una estada en el extranjero (*Erasmus*), ya que entonces se hace realmente difícil realizar una evaluación práctica de su nivel de conocimiento.

Estos problemas han sido observados repetidas veces por los responsables de la docencia médica, que han buscado soporte tecnológico para intentar solucionarlos. Al estudiar el origen de los problemas, se llegó a la conclusión de que se podrían resolver si hubiera disponible algún sistema fiable en el que realizar los ejercicios prácticos de formación, desde cualquier ubicación y con un registro de los parámetros importantes para la evaluación de los aprendices (errores, tiempo, precisión, etc.). Este sistema debería ser fácil de utilizar y personalizable, de modo que los procedimientos puedan ser modificados y ampliados en caso de necesidad.

El sistema solicitado permitiría que los estudiantes pudieran practicar los procedimientos estudiados de modo instantáneo, sin necesidad de realizar la preparación de los escenarios ni disponer de las herramientas físicas. Al tratarse de una simulación virtual, el sistema también podría ser utilizado por los docentes para realizar seguimientos y evaluaciones a distancia, dado que las únicas necesidades serían que el estudiante tuviera acceso al sistema de simulación y una conexión estable con el evaluador.

2.3. Actores implicados

El sistema solicitado tendrá implicaciones importantes en el desarrollo de la formación del personal médico, las cuales podrán afectar a los diversos perfiles involucrados de modo distinto. A continuación realizaremos un análisis de los actores que se beneficiarán del proyecto:

- **Estudiantes de enfermería:** Proporcionaremos una nueva herramienta de aprendizaje a los estudiantes de auxiliar de enfermería, mediante la cual podrán practicar los procedimientos en un entorno virtual seguro. Esta posibilidad supondrá una mejora en la disponibilidad y el realismo del aprendizaje virtual, ya que se podrá realizar una simulación en primera persona de los procedimientos con solo disponer de unos cascos de realidad virtual y el espacio que estos necesitan para funcionar.
- **Personal docente de enfermería:** Aportaremos mayor variedad a las metodologías prácticas de aprendizaje, de modo que los docentes podrán decidir qué metodología es más conveniente aplicar en cada momento. También se mejorará la comodidad de los docentes, dado que podrán realizar evaluaciones simultáneas y a distancia de manera cómoda, sin necesidad de montar y recoger el material médico de las pruebas ni de vigilar el correcto uso de material peligroso.
- **Hospital Sant Joan de Déu:** La institución se está esforzando en aplicar metodologías modernas e innovadoras, así que gracias al proyecto podrá empezar a aplicar la enseñanza mediante realidad virtual. Además de ganar prestigio, mejorará la calidad y variedad de los métodos docentes aplicados actualmente, incrementando también la seguridad de todos los implicados en la enseñanza a la hora de manipular sustancias y herramientas peligrosas.

3. Actualidad de la tecnología aplicada

3.1. Descripción de la tecnología

Con el objetivo de mejorar el proceso educativo de los auxiliares de enfermería, aplicaremos diversas tecnologías modernas para realizar simulaciones realistas de escenarios. La principal característica de estas simulaciones será el uso de gafas de Realidad Virtual, de modo que el usuario realizará una experiencia en primera persona totalmente inmersiva: podrá ver y escuchar todo lo que sucede en el entorno virtual, sintiendo que realmente se encuentra en él. Las tecnologías aplicadas, entonces, estarán relacionadas con el ámbito de los *Gráficos por ordenador*, concretamente en su uso para diseñar entornos de Realidad Virtual inmersivos.

El dispositivo de realidad virtual utilizado permitirá que el usuario se mueva físicamente por un espacio reducido de modo seguro. Este espacio estará controlado en todo momento por el usuario utilizando los cascos, pues podrá visualizar la delimitación del espacio físico mediante un sistema de paredes virtuales llamado *Chaperone*[1]. Utilizando los mandos incluidos en el dispositivo de Realidad Virtual, se permite realizar un seguimiento constante del movimiento de las manos del usuario, las cuales son visualizadas en la escena y permiten interactuar con los objetos virtuales de manera simple y natural.

Los sistemas de Realidad Virtual comercializados actualmente suelen necesitar un ordenador con una tarjeta gráfica suficientemente potente para procesar las simulaciones, y ya incluyen un sistema de configuración sencilla del entorno, mediante la que se establece si la simulación será de pie o sentado además de la referencia del espacio real disponible. La localización del usuario en el entorno se deberá realizar mediante el posicionamiento de sensores en esquinas estratégicas, aunque hoy en día ya han aparecido versiones de cascos que incorporan tanto el sistema de sensores como el procesamiento gráfico en el propio dispositivo [26], de modo que el usuario se pueda olvidar de realizar todas las conexiones correctamente.

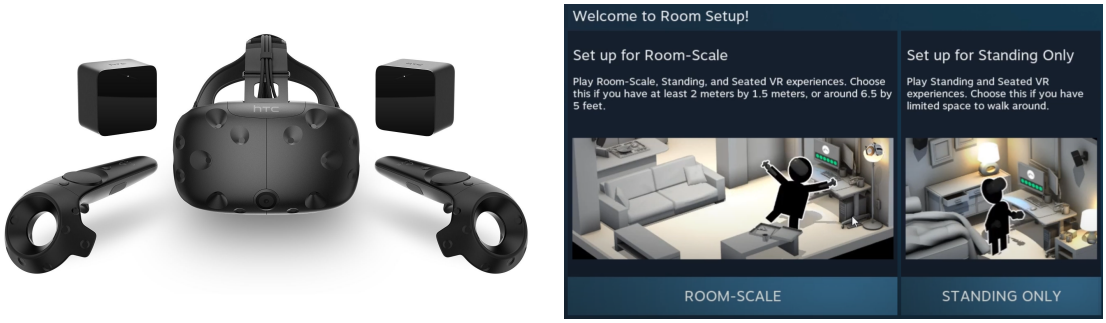


Figura 3: *Cascos de Realidad Virtual HTC VIVE, se configuran utilizando SteamVR. Fuentes: [4, 5]*

Estos dispositivos proporcionan una herramienta útil y fiable sobre la que establecer cualquier tipo de escenario virtual inmersivo, los cuales pueden ser diseñados utilizando motores gráficos y plataformas de desarrollo actuales. Estas plataformas ya disponen de las herramientas necesarias para poder utilizar cualquiera de estos dispositivos, de modo que la portabilidad entre ellos sea lo menos problemática posible. Gracias a la disposición de una tecnología sólida y fiable sobre la que trabajar, se puede focalizar todo el trabajo a realizar en el propio proyecto, sin necesidad de preocuparnos por modificaciones a nivel del *hardware* que se utilizará.

[1] Sistema utilizado por *Valve* para mostrar una cuadrícula en el escenario virtual que delimita la "zona segura" de juego del usuario. Otras compañías utilizan sistemas parecidos con un nombre distinto.

3.2. Proyectos parecidos

Dado el increíble avance en la tecnología de Realidad Virtual que ha sucedido estos últimos años, varios grupos de investigación han encontrado una oportunidad perfecta para aplicar sus conocimientos y recursos en dicha tecnología para el ámbito médico. Conociendo este dato, es necesario realizar un análisis de las opciones que están disponibles hoy en día en el mercado para solucionar la necesidad de utilizar un simulador para entrenar al personal médico. Mediante este análisis previo, podremos reflexionar si realmente es necesario realizar un proyecto desde cero y observar las características diferenciativas que tendría nuestro producto.

- **Oxford Medical Simulation:** En asociación con la *Universidad de Oxford*, se trata de un simulador de Realidad Virtual en el que se deberán realizar exámenes y pruebas a pacientes. Se puede utilizar material básico como fonendoscopios e inyecciones, además de poder consultar análisis y radiografías del paciente, consultar con él los síntomas y realizar un examen de dilatación ocular. Después de las pruebas se observa una lista de los errores realizados y cómo solucionarlos.

Figura 4: *Oxford Medical Simulation. Fuente: [6]*

- **UbiSim:** Proyecto multidisciplinar localizado en Canadá y Suiza, con clientes importantes como la *Cruz Roja francesa* y la *Universidad de Montréal*. El simulador permite, entre otras cosas, consultar el estado del paciente usando un fonendoscopio o un electrocardiograma, además de permitir realizarle transfusiones o cambiarle el suero fisiológico.

Figura 5: *UbiSim. Fuente: [7]*

- **VR Medical:** Proyecto asociado, entre otros, con la *Universidad de Wisconsin-Madison*. El simulador permite escoger diferentes roles para realizar cirugías y pruebas médicas al paciente, con interacción tanto directamente con el paciente como con herramientas y heridas. Se permite el acceso remoto al entrenamiento VR para poder seguir aprendiendo desde cualquier entorno.

Figura 6: *VR Medical. Fuente: [8]*

Al haber analizado algunos de los sistemas utilizados hoy en día para entrenar a personal médico con Realidad Virtual, podemos observar que estos sistemas utilizan sesiones fijas que solo pueden ser modificadas por los programadores responsables de su desarrollo. Como veremos más adelante, la necesidad de que el sistema sea personalizable es una de las principales razones por las que se ha solicitado hacer nuestro proyecto, dado que los protocolos y parámetros de evaluación varían considerablemente dependiendo de la región, el centro y el momento en el que se aplique. Es por eso que encontramos la necesidad de realizar un simulador propio que sea personalizable y escalable de manera rápida.

4. Características del proyecto

4.1. Objetivos

La petición del proyecto es el desarrollo de una aplicación de Realidad Virtual que permita el entrenamiento y la evaluación de personal de enfermería mediante la configuración de sesiones en un entorno virtual. El sistema permitirá definir y configurar sesiones de entrenamiento (como por ejemplo la preparación de la mesa quirúrgica, entrega de instrumental al cirujano durante la intervención, etc.), proporcionando una experiencia totalmente inmersiva mediante el uso de gafas y controladores de Realidad Virtual.

Una vez definido el principal problema al que deberemos enfrentarnos, podemos definir los requisitos clave que deberá tener el sistema a desarrollar:

1) **Multiplataforma:** Se requiere que la simulación se pueda realizar sin restricción del casco de realidad virtual que se desee utilizar.

2) **Personalizable:** Las pruebas a realizar deberán ser modificables al gusto del evaluador, de modo que se pueda alterar el orden y la cantidad de eventos que sucedan en el flujo de ejecución de la prueba.

3) **Sencillo:** El sistema debe ser autoexplicativo y fácil de utilizar por el usuario final, con un diseño sencillo que permita que lo utilice personal sin altos conocimientos informáticos.

4) **Evaluable:** Los valores numéricos de rendimiento del usuario que ha realizado la prueba deberán ser accesibles con facilidad, de modo que se pueda realizar una evaluación y comparación de la mejora de la técnica del técnico.

5) **Extensible:** Como los procedimientos médicos avanzan rápidamente (se doblan cada 6-8 años [27]), necesitamos un proyecto que sea fácilmente escalable. Para cumplir este objetivo, debería ser más sencillo añadir nuevos procedimientos al sistema que crear un sistema nuevo desde cero.

6) **Autónomo:** El sistema deberá contener todo el software necesario unificado, de modo que no se requieran instalaciones adicionales que podrían derivar en problemas de versión. De este modo, también aumentaremos la fiabilidad de ejecución del simulador en diferentes equipos.

4.2. Plataforma y ampliaciones

Para el desarrollo del proyecto hemos decidido utilizar como motor de juego *Unity*, ya que es un motor gráfico sencillo, multiplataforma, con el que tenemos experiencia y que dispone de variedad de herramientas de soporte al desarrollo para Realidad Virtual. Para conseguir el soporte de realidad virtual requerido, nosotros utilizaremos VRTK (acrónimo de *Virtual Reality Toolkit*) [28], que es un asset disponible en la tienda de Unity de manera gratuita que nos ayudará a disponer de las funcionalidades básicas que necesitaremos para el desarrollo de los distintos escenarios.

Para asegurar que las sesiones de nuestro simulador cumplan el objetivo de la personalización, utilizaremos como base el sistema de edición de nodos personalizable desarrollado por Sergi Tortosa en su Trabajo Final de Máster del cuadrimestre pasado [9], ya que nos ofrece un sistema de flujo modificable sobre el que desarrollar toda la lógica para configurar y realizar las pruebas requeridas por nuestro cliente. Podemos determinar que nuestro proyecto será una continuación y ampliación del trabajo realizado por Sergi, dado que se realizó una plataforma sobre la que poder desarrollar el sistema de simulación y buena parte de nuestros objetivos son los especificados como trabajo por hacer en la conclusión de su proyecto.

Es necesario detallar que el proyecto está compuesto por tres partes bien diferenciadas que están relacionadas entre ellas:

– **Menú principal:** Pantalla inicial de la aplicación, que permite añadir y eliminar sesiones al sistema. También es el encargado de enlazar las otras dos escenas del proyecto dependiendo si se quiere modificar la sesión o ejecutarla.

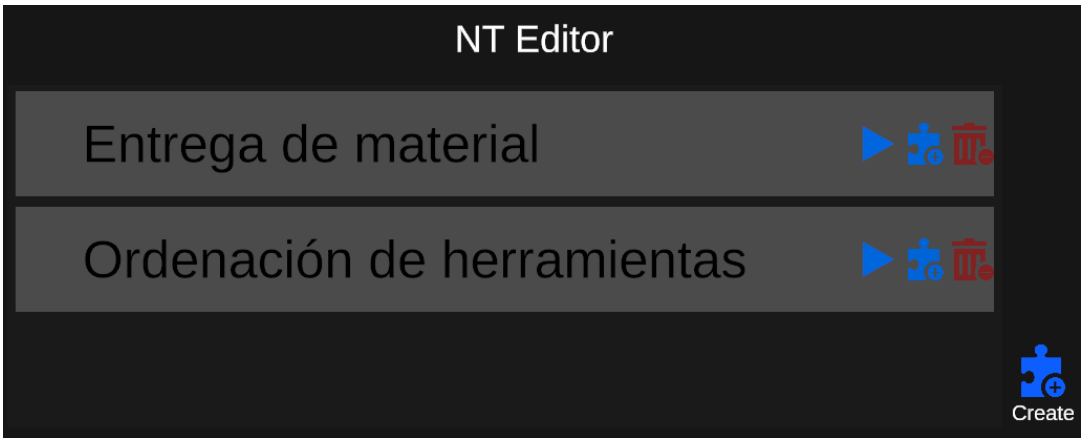


Figura 7: *Menú principal del proyecto*

– **Editor de sesiones:** Pantalla desde la que se pueden modificar los objetos de la sesión y toda su lógica, compuesta principalmente por una vista previa de la escena y un editor de los nodos para configurar la lógica.

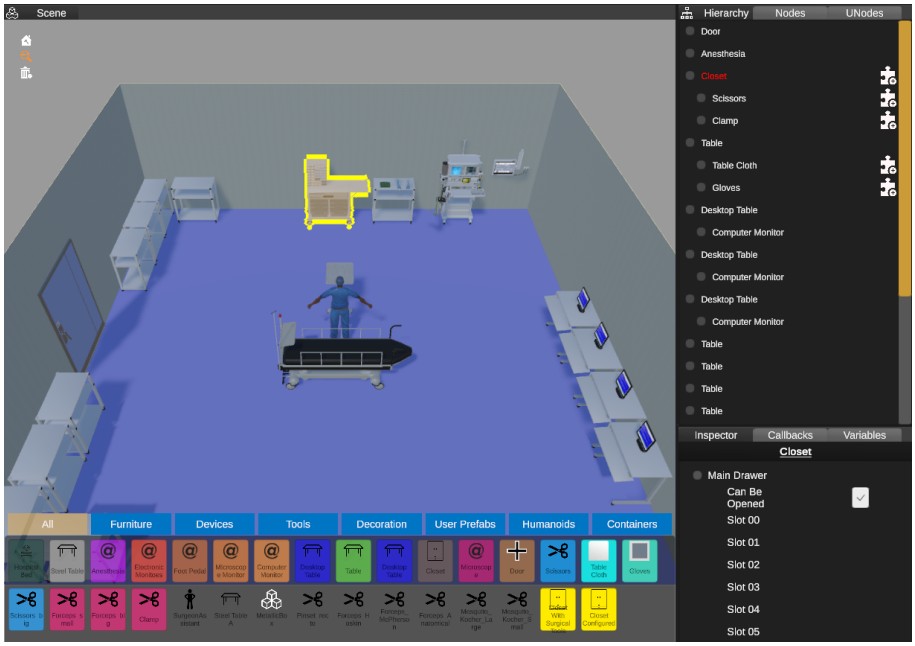


Figura 8: *Visualización del editor de escena*

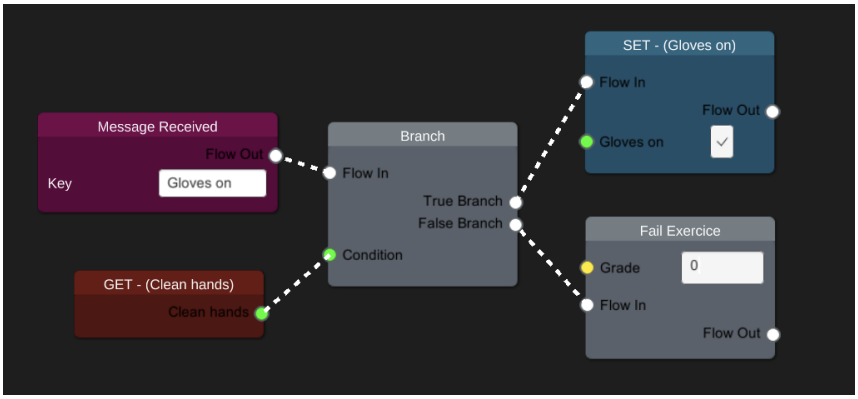


Figura 9: *Editor de nodos. Fuente: Documento [9]*

– **Simulador:** Escena en la que ejecutará la sesión, de modo que se deberán realizar las tareas correspondientes utilizando las gafas y controladores de realidad virtual.

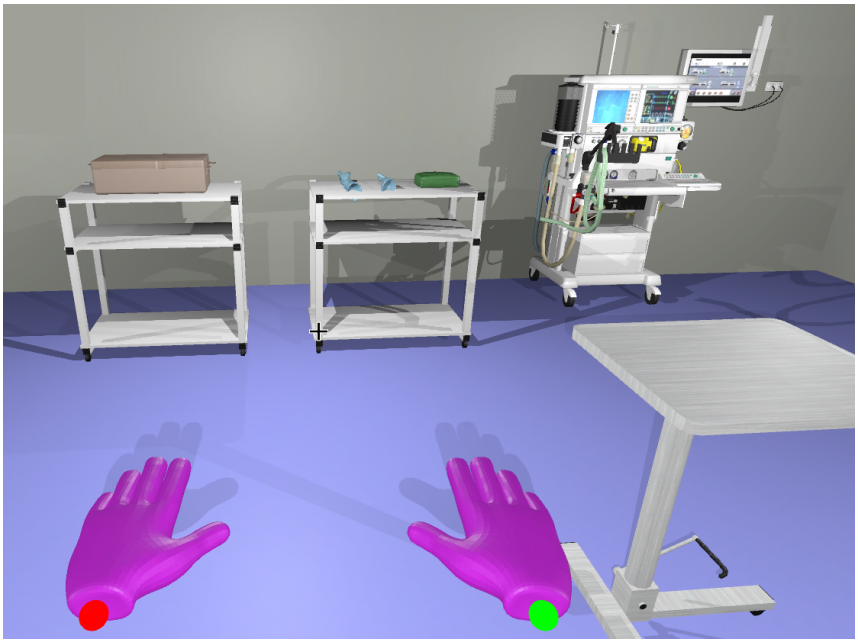


Figura 10: *Visualización de la escena desde la simulación*

Será necesaria una ampliación en la variedad de opciones disponibles en el sistema de edición de nodos personalizables, ya que la configuración que disponemos como base es demasiado sencilla. Entre las características que necesitaremos añadir tenemos:

- **Cuadros de texto:** Necesitamos que puedan aparecer mensajes de texto durante las pruebas, tanto a modo de información del estado como para informar al usuario de lo que se espera que haga a continuación. Al tratarse de una una aplicación de realidad virtual, su implementación será más compleja que su análogo *2D*, ya que deberá decidirse su emplazamiento en el escenario *3D* asegurando que se realice la interacción esperada con los demás objetos.
- **Reproducción de voz:** Como soporte al sistema de información al usuario, se requerirá de un sistema personalizable de síntesis de voz a partir de texto, de modo que el usuario no tenga que ir leyendo continuamente cuadros de texto, sino que pueda recibir la información también de forma oral, que es mucho más cómoda y natural en escenarios reales. Estos mensajes de voz se habrán establecido con antelación y deberán reproducirse junto a todo el sistema de sonido de la aplicación.

- **Interacciones:** Deberemos ampliar la cantidad de interacciones disponibles entre los diferentes objetos y con las personas virtuales, para que la experiencia sea más realista. Por ejemplo, deberemos permitir la interacción con el cirujano, haciendo que pueda pedir instrumental al estudiante para que este lo recoja y se lo entregue, todo ello en un entorno *3D* inmersivo que imite las interacciones en la realidad.
- **Plantillas de prueba:** Para aportar más comodidad a la configuración de las pruebas, necesitaremos que las sesiones básicas estén almacenadas en plantillas, de modo que se puedan añadir y quitar del flujo de ejecución de modo sencillo y ya se disponga de una base configurada. De este modo, podremos partir de un escenario con los objetos necesarios y la lógica correcta para la sesión deseada, para que el usuario pueda personalizar y ejecutar la simulación realizando la mínima cantidad de modificaciones posibles.
- **Guardado y Carga de sesiones:** Se necesita un sistema de guardado y carga de sesiones en el sistema, de modo que se puedan realizar configuraciones con anterioridad y se puedan cargar al instante para realizar las pruebas. Esta opción será muy útil para poder guardar las sesiones personalizadas y que se puedan copiar a otros dispositivos para utilizarlas.

4.3. Características de las sesiones

Hay distintas sesiones básicas que se han especificado de acuerdo a las necesidades aportadas por los expertos del dominio. En el editor de nodos se puede escoger entre utilizar estas sesiones como base o configurar tus propias sesiones desde cero. Entre las sesiones preconfiguradas encontramos:

- **Preparación de la mesa quirúrgica:** El usuario deberá manipular y ordenar correctamente el material que se utilizará en la cirugía antes de que ésta comience, respetando las normas de esterilidad.
- **Entrega de instrumental al cirujano:** El usuario deberá entregar de manera correcta el instrumental requerido por el cirujano durante la cirugía. El material será pedido tanto de manera oral como en un cuadro de texto, de modo que el usuario deberá identificarlo por el nombre.
- **Protocolo de vestimenta:** El usuario deberá vestirse para entrar a quirófano. Será muy importante que siga los pasos requeridos en el protocolo y que ningún objeto esterilizado entre en contacto con zonas no estériles.
- **Reacción ante emergencias:** El usuario deberá reaccionar correctamente ante eventos inesperados, siguiendo los protocolos apropiados para solucionar la situación.

4.4. Obstáculos y riesgos

Al determinar el alcance de nuestro proyecto, también es importante realizar una reflexión de aquellos factores que pueden perjudicar su correcta ejecución, de modo que se puedan prevenir en la medida de lo posible.

- **Falta de conocimientos:** Es importante ser conscientes que en algún punto del desarrollo del proyecto podemos detectar que nos faltan los conocimientos técnicos para resolver una tarea. En este caso será muy importante volver a planificar el desarrollo a corto plazo, de modo que haya tiempo para resolver esta falta técnica sin perjudicar excesivamente al proyecto.
- **Comunicación:** Dado que el proyecto es una colaboración entre distintas entidades, deberemos asegurar que haya un buen flujo de información entre ellas. Cualquier falta de comunicación podría atrasar los tiempos del proyecto, de modo que deberá detectarse y solucionarse lo más rápido posible.

- **Cambio de alcance:** Deberemos tener en cuenta que puede haber grandes cambios al revisar las tareas, de modo que se pueda necesitar ampliar una parte del desarrollo por petición del cliente. Estos cambios en el alcance de las tareas pueden evitarse con una buena comunicación y una planificación que tenga en cuenta que una parte del tiempo se puede derivar en solucionar problemas y cambios de última hora.
- **Falta de tiempo:** La falta de tiempo a la hora de terminar los desarrollos suele ser debido a una mala planificación. Para evitarlo deberemos realizar una planificación de tiempos pesimista, de modo que siempre tengamos algo de tiempo libre para reasignar en tareas que faltaría pulir.
- **Imposibilidad de acceso al hardware necesario**: Si bien este riesgo no fue contemplado al inicio del proyecto, se ha considerado conveniente la necesidad de añadirlo. En la planificación del proyecto se contó con la plena disponibilidad de los sistemas de realidad virtual del grupo *ViRVIG*, concretamente un sistema *HTC Vive* y un sistema *Oculus Quest*, con dos controladores cada uno. Lamentablemente, la situación de alarma actual causada por el *COVID-19* ha imposibilitado el acceso a este material en una etapa crítica del proyecto. Aunque el desarrollo ha podido completarse, las pruebas se han tenido que desarrollar con la dificultad añadida de tener que utilizar un modo simulación que imita la entrada *3D* de los controladores utilizando teclado y ratón. Esto obligará al planteamiento de una revisión del sistema de interacción utilizando los controladores físicos, que deberá ejecutarse una vez termine el estado de alarma que actualmente nos concierne.

4.5. Metodología

Dada la gran variedad de técnicas y herramientas de organización de proyectos, hemos valorado las características de nuestro proyecto para escoger el sistema de desarrollo que mejor se adapta a nuestras necesidades. Después de comparar las opciones de organización del desarrollo, hemos llegado a la conclusión de que la mejor opción es seguir una *Metodología Ágil* [29]. Utilizando esta metodología, podremos obtener rápidamente una primera versión usable del producto, y mejorarla mediante iteraciones de las diferentes fases de desarrollo. La mayor ventaja de esta metodología es que, al requerir de repetidas iteraciones en el producto, aseguraremos que el diseño y la estructura del proyecto sean fácilmente escalables, de modo que se pueda ampliar de modo sencillo en un futuro si fuera necesario.

La Metodología Ágil también nos permite dividir y organizar de manera mucho más fácil el proyecto, dado que las iteraciones se deberán realizar sobre las distintas partes a mejorar en el producto. Las iteraciones se realizan en *Sprints*, que es el tiempo esperado de desarrollo entre una versión estable y otra. Hemos determinado que el tiempo de *Sprint* que utilizaremos será de dos semanas, dado que nos permite empezar realizando una planificación de las tareas, disponer del tiempo suficiente para desarrollarlas y finalmente realizar una demostración y evaluación de los avances realizados. Después de realizar cada *Sprint*, se realizará una reunión de control en la que se analizará todo el trabajo hecho para redistribuir la carga de trabajo del siguiente desarrollo si fuera necesario; de este modo podemos adaptar la carga de trabajo a las necesidades del proyecto y las capacidades de los desarrolladores de manera cómoda.

Como herramienta de organización de proyectos, hemos decidido utilizar *Trello*. Esta herramienta en línea permite organizar proyectos mediante el uso de tableros y notas, de modo que podremos realizar fácilmente un seguimiento de las tareas a realizar con un sistema muy visual. A parte de ser muy útil para planificar el desarrollo, también nos permite realizar otras tareas más organizativas como hacer una lluvia de ideas o realizar valoraciones conjuntas del estado del proyecto en las reuniones.

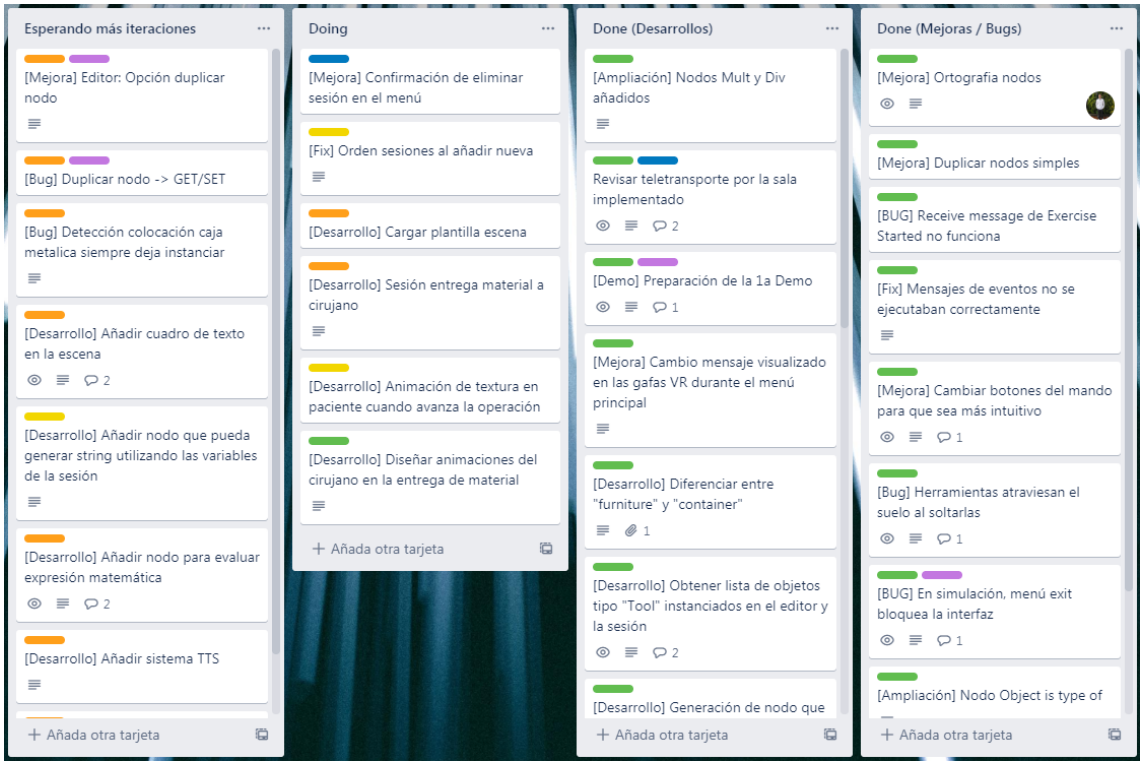


Figura 11: *Trello del proyecto en su fase final*

5. Planificación del desarrollo

5.1. Descripción inicial de las tareas

En esta sección realizaremos un estudio y explicación de las tareas que se ha estimado que deberán realizarse en el proyecto, antes de realizar su desarrollo y definiéndolas individualmente para poder agruparlas por temática.

- **Desarrollo del software**
  - **Modificaciones del editor**
    1) **Ampliación de los tipos de nodos del editor:** Para poder realizar las configuraciones necesarias para las pruebas en el editor de nodos, necesitaremos más variedad en los nodos básicos proporcionados por el sistema, de modo que se pueda añadir tanta complejidad al flujo como sea necesaria. Esta puede ser una tarea arbitrariamente compleja, ya que por ejemplo podríamos querer añadir un nodo que verifique si el instrumental quirúrgico está colocado sobre la mesa quirúrgica; esto implicaría diseñar toda la lógica del nodo además de su interfaz, su relación con otros objetos, el guardado y cargado de sus datos, etc. También hay que tener en cuenta que el comportamiento del nodo deberá ser el esperado tanto en el editor como a la hora de utilizarse durante la ejecución de la simulación.
    2) **Diseño de sesiones de ejemplo:** Una vez ampliados los nodos del editor, los utilizaremos para diseñar las diferentes sesiones que se requieren en el proyecto. Con esto obtendremos unas sesiones base que utilicen la lógica implementada para realizar la simulación.
    3) **Añadir uso de plantillas y guardado de modificaciones:** Se requiere que el editor permita cargar plantillas prediseñadas para cada prueba. De este modo, cuando queramos crear una nueva sesión en la aplicación, tendremos la opción de utilizar una sesión en blanco o partir de una sesión prediseñada para empezar con una base configurada y solo tener que realizar modificaciones. También será necesario permitir extraer y cargar las sesiones modificadas para moverlas entre dispositivos.
    4) **Adaptación de la interfaz del editor al usuario final:** Como nuestro usuario final tendrá un nivel básico de informática, deberemos adaptar los conceptos y el diseño de la interfaz actual del editor de nodos para que sea más intuitiva.

  - **Funcionalidades del simulador**
    5) **Enriquecer tipos de objeto:** La cantidad de objetos utilizada hasta el momento es demasiado simple, de modo que se ampliará la diversidad de objetos de decoración y herramientas interactuables para que los entornos sean más realistas y se pueda ajustar mejor la complejidad de las simulaciones. Los objetos añadidos se agruparán en categorías que estarán disponibles en el editor, de este modo será más sencillo diferenciar entre elementos decorativos, herramientas interactuables, compañeros, etc.
    6) **Implementar comportamiento de objetos:** Una vez añadidos los objetos correctamente, habrá que implementar el comportamiento de aquellos con los que el usuario pueda tener alguna interacción. La complejidad del desarrollo dependerá de las interacciones deseadas y la lógica del objeto, siendo más costoso implementar características que todavía no estuvieran contempladas en el sistema. Por ejemplo: añadir una decoración es sencillo porque no se deberá interactuar con ella, pero añadir un contenedor de objetos es complejo porque se deberá implementar tanto su comportamiento desde el editor como desde dentro de la simulación.

7) **Añadir sintetizador de voz:** Para la personalización del sistema se requiere poder introducir texto y que sea pronunciado durante la simulación, de modo que se pueda transmitir la información al usuario de una manera más natural. El texto a pronunciar deberá haberse introducido con antelación durante la configuración de la sesión. Este proceso será posible añadiendo un sistema de síntesis de voz (en inglés conocido como *text to speech*), proceso cuya complejidad será elevada porque incluye tanto modificaciones en la simulación, como en el editor y el sistema de sonido de la aplicación.

8) **Añadir visualización de texto:** Para poder describir las tareas a realizar, se requiere poder introducir texto personalizado en ciertos momentos de la simulación para que el usuario los pueda leer. Estos textos deberán poder ser introducidos tanto como mensaje de texto *3D* en la escena como información adicional en el lateral del visor. Esta tarea, por lo tanto, integra modificaciones en los nodos del editor, en los elementos *3D* de la escena y en la interfaz *2D* del visor.

9) **Animación de modelos (cirujano, enfermeras, paciente, ...):** Para dar más realismo y dinámica a las pruebas, necesitaremos implementar animaciones a las personas virtuales que rodeen al usuario durante las simulaciones. Es natural que los demás compañeros se muevan y hagan otras tareas, además de gesticular cuando nos quieran transmitir información. La parte más compleja de esta tarea será la coordinación de las animaciones con el momento en el que deberán ejecutarse, ya que se trata de una interacción que no estaba contemplada inicialmente en el proyecto y deberá incluirse en el comportamiento de los nodos.

10) **Mejora interacción en Realidad Virtual:** Se configurarán todos los parámetros de interacción desde el punto de vista del jugador, de modo que la experiencia sea más fluida y natural. Las modificaciones incluirán verificaciones en las animaciones de la mano al agarrar y soltar herramientas, la posición de las herramientas en la mano, la interacción con la interfaz del menú de la simulación, entre otras.

11) **Añadir sonido ambiente:** Para proporcionar una mejor ambientación al simulador, se añadirá música ambiente y sonidos a los distintos objetos y herramientas. La inclusión de un sistema de sonido en la aplicación hará que la experiencia sea mucho más inmersiva y natural, aportando más solidez a la simulación que hasta el momento solo era visual. En esta tarea hay un alto nivel de trabajo, dado que primero habrá que hacer una selección de los distintos sonidos y músicas que se quieren añadir al sistema, para después incorporarlos y hacer que se coordinen entre ellos de manera lógica.

12) **Puntuación de los resultados:** Una vez terminada cada simulación, necesitamos obtener los datos de tiempo y precisión del usuario en la prueba para poder valorar si se ha realizado correctamente o hay puntos a mejorar. Poder extraer valores numéricos de los apartados de la simulación será un buen método para comparar distintas ejecuciones, de modo que se puedan realizar estudios estadísticos de aprendizaje y dificultad con los resultados obtenidos. También se deberán poder extraer datos más visuales como imágenes de distintos elementos de la escena, para poder valorar su evolución. Este apartado requerirá tanto de configuración en la escena de la simulación como en los nodos del proyecto, de modo que los datos se extraigan de manera automatizada.

- **Gestión del proyecto**

13) **Revisión del proyecto:** Se realizarán reuniones con el director del proyecto cada una o dos semanas para realizar un análisis del sprint actual y revisar el siguiente. En estas reuniones se pondrá en común cualquier duda y se establecerán las prioridades en el siguiente desarrollo. También se incluye en este apartado cualquier reunión que esté relacionada con dudas en temas técnicos, tanto de funcionamiento de la aplicación como de estrategias y tecnologías para los desarrollos.

14) **Reunión con cliente:** Para comprobar que el desarrollo se adapta correctamente a las necesidades del proyecto, se realizará una reunión física con el cliente al menos una vez cada dos meses. En esta reunión se pondrán en común los objetivos realizados hasta el momento y la planificación hasta la siguiente reunión, de modo que se verifique la corrección de los desarrollos y se pueda detectar cualquier incongruencia lo antes posible.

15) **Documentación:** La documentación del proyecto deberá ampliarse cada semana con los avances realizados y el seguimiento del proceso de desarrollo. Realizar la documentación de modo progresivo disminuirá la carga final de esta tarea.

Una vez realizada la definición de las tareas, deberemos observar si algunas de ellas tienen correlación en el orden del desarrollo para tenerlo en cuenta durante la planificación. En nuestro proyecto podemos observar dos conjuntos de tareas que tienen dependencia entre ellas:

- *1) Ampliación de los tipos de nodos del editor* es prerrequisito de *2) Diseño de sesiones de ejemplo*, dado que las sesiones se diseñarán utilizando todos los nodos del editor, de modo que ciertos comportamientos no se podrán representar hasta que se implementen los nodos que los representan.
- *5) Enriquecer tipos de objeto* es prerrequisito de *6) Implementar comportamiento de objetos*, ya que no se podrá configurar el comportamiento de los objetos hasta que estos sean añadidos correctamente al proyecto.
- *11) Añadir sonido ambiente* es prerrequisito de *7) Añadir sintetizador de voz*, dado que se deberá haber configurado correctamente todo el sistema de sonido de la aplicación para poder incorporar la lectura de texto y que se coordine con los demás sonidos existentes.

5.2. Planificación inicial del tiempo

Para entender correctamente los datos temporales que se mostrarán junto a cada tarea, primero deberemos poner en contexto las horas de las que disponemos. Según el planteamiento inicial, el desarrollo del proyecto se realizará desde el día 1 de octubre de 2019 hasta el 10 de enero de 2020, que son **70 días laborables**; invertiremos 5 horas diarias de media, lo que se traduce en un total de **350 horas útiles**. La fecha de entrega final del proyecto es el día 15 de enero de 2020, y su lectura está por acordar, entre el 23 y el 29 de enero de 2020.

Nº	Tarea	Horas
-	**Modificaciones del editor**	**80**
1	Ampliación de los tipos de nodos del editor	30
2	Diseño de sesiones de ejemplo	30
3	Añadir uso de plantillas y guardado de modificaciones	10
4	Adaptación de la interfaz del editor al usuario final	10
-	**Funcionalidades del simulador**	**140**
5	Enriquecer tipos de objeto	30
6	Implementar comportamiento de objetos	30
7	Añadir sintetizador de voz	20
8	Añadir visualización de texto	10
9	Animación de modelos (cirujano, enfermeras, paciente, ...)	20
10	Mejora interacción en Realidad Virtual	10
11	Añadir sonido ambiente	20
12	Puntuación de los resultados	20
-	**Gestión del proyecto**	**80**
13	Revisión del proyecto	20
14	Reunión con cliente	10
15	Documentación	60
-	**TOTAL**	**330**

Tabla 1: *Tiempo inicial de las tareas*

Como podemos observar, el total de horas del proyecto muestra 20 horas menos de las planificadas originalmente. Estas horas sobrantes en la planificación, nos permitirán disponer de tiempo para administrar en imprevistos y posibles complicaciones en las tareas.

5.3. Recursos del proyecto

Para poder realizar el proyecto hemos utilizado diversos recursos, cada uno de ellos aportando alguna funcionalidad clave para que todo se desarrolle de manera organizada y correcta. El análisis económico de estos recursos se realiza en el apartado de *Presupuesto*, de modo que aquí simplemente se listarán y se detallará su función específica en el proyecto.

Recurso	Tipo	Descripción
Desarrollador	Humano	Responsable de la organización y ejecución del proyecto
Ordenador Windows	Material	Dispositivo para desarrollar el proyecto que tendrá todo el software necesario instalado
HTC Vive	Material	Dispositivo de realidad virtual en el que se ejecutarán las simulaciones
Unity	Software	Plataforma de desarrollo en la que se programará el proyecto
Visual Studio	Software	IDE utilizado para modificar código en Unity
C#	Software	Lenguaje de programación utilizado en Unity
GitHub	Software	Sistema de versiones en línea para almacenar los cambios en el desarrollo y disponer del proyecto desde cualquier lugar
Overleaf	Software	Recurso en línea para realizar la documentación utilizando LaTeX.
Trello	Software	Recurso en línea para organizar el proyecto y controlar su correcto desarrollo
Google Drive	Software	Recurso en línea para almacenar copias de seguridad de todas las partes del proyecto semanalmente. También permite escribir documentos *Office* en línea.

Tabla 2: *Descripción de los recursos utilizados*

5.4. Planificación inicial mediante Gantt

Una vez definidas las tareas a realizar y su coste de trabajo en horas, vamos a realizar una representación gráfica (mediante la esquematización Gantt) de las tareas planificadas a desarrollar en cada semana del proyecto, con las horas a aplicar en cada una de ellas:

Número Tarea	Nombre Tarea	Semanas															
		Octubre				Noviembre					Diciembre				Enero		
		1..6	7..13	14..20	21..27	28..3	4..10	11..17	18..24	25..1	2..8	9..15	16..22	23..29	30..5	6..12	13..19
	Modificaciones del editor																
1	Ampliación de los tipos de nodos	10	10	5	5												
2	Diseño de sesiones de ejemplo				5	5	5	5	5	5							
3	Añadir uso de plantillas y guardado de modificaciones										5	5					
4	Adaptación de la interfaz del editor al usuario final												5	5			
	Funcionalidades del simulador																
5	Enriquecer tipos de objeto	10	10	5	5												
6	Comportamiento de objetos			5	5	5	5	5	5								
7	Añadir sintetizador de voz									5	5	5	5				
8	Añadir visualización de texto					5	5										
9	Animación de modelos						5	5	5	5							
10	Mejora interacción en VR			5		5											
11	Añadir sonido ambiente										5	5	5	5			
12	Puntuación de los resultados							5	5	5	5						
	Gestión del proyecto																
13	Revisión del proyecto	1	2		2	1	1	2		2	1	2		2	2		2
14	Reunión con cliente			2					2				2			4	
15	Documentación	2	2	2	2	2	2	2	2	2	2	2	2	4	14	14	4
Horas de la semana		**23**	**24**	**24**	**24**	**23**	**23**	**24**	**24**	**24**	**23**	**19**	**19**	**16**	**16**	**18**	**6**

Figura 12: *Planificación inicial de las tareas por semana*

5.5. Gestión del riesgo: planes alternativos y obstáculos

A nivel organizativo, es muy importante prever situaciones que puedan poner en riesgo el proyecto con antelación, de modo que se puedan plantear medidas para evitarlas y tener en cuenta el impacto en caso de que sucedan.

- **Falta de conocimientos:** *Impacto bajo*. En caso de requerir el aprendizaje de nuevos conocimientos para seguir con alguna tarea, primero de todo habría que pausar el desarrollo de la tarea. Al utilizar una metodología *Agile*, no habría problema en dejarla para más tarde y seguir realizando otras tareas que no estén bloqueadas.
  - La solución con menos impacto sería aprender estos conocimientos de manera autónoma y fuera del tiempo del proyecto, ya que de este modo no habría que modificar la planificación y se podría seguir con la tarea una vez se adquieran los conocimientos.
  - Si se necesitara invertir tiempo del proyecto en el aprendizaje de estos conocimientos, se podría utilizar parte de las 20 horas sobrantes en la planificación inicial. Si este tiempo libre ya no estuviera disponible, habría que volver a planificar las semanas pendientes del proyecto para poder terminar el desarrollo satisfactoriamente.
- **Comunicación:** *Impacto medio*. Una buena comunicación entre los diferentes grupos relacionados con el proyecto es clave para su correcto desarrollo. La mala comunicación en cualquiera de las partes implicadas provocaría que las tareas se realicen de un modo no deseado, así que los plazos del proyecto se alargarían.
  - Para evitar esto, se necesita una fuente fiable de comunicación (en nuestro caso *correo electrónico* y *teléfono*) y que las reuniones con los implicados sean altamente explicativas; de modo que todos estén de acuerdo con qué y cómo se va a desarrollar en cada fase del proyecto.
  - Si se detectan incongruencias o dudas entre lo especificado y lo desarrollado, deberá comunicarse inmediatamente a los implicados para aclarar el problema. Si esto no fuera suficiente, deberá realizarse una reunión extra a las planificadas para lograr llegar a una solución con el mínimo impacto posible en el proyecto.
- **Cambio de alcance:** *Impacto alto*. La definición de las tareas puede verse modificada a lo largo del desarrollo, bien porque no se especificó correctamente o porque no se tuvieron en cuenta ciertas necesidades para su implementación.
  - El tiempo de duración de las tareas se ha intentado valorar de modo generoso, aunque aún puede haber algunos imprevistos. En caso que sucediera un cambio de alcance en alguna de las tareas, podremos utilizar las 20 horas sobrantes de la planificación inicial para intentar solucionar el problema. Si esto no fuera suficiente, necesitaríamos volver a planificar las semanas restantes de modo que todas las tareas se pudieran finalizar a tiempo.
- **Pérdida de datos:** *Impacto alto*. Para realizar una correcta prevención de riesgos, hay que tener en cuenta que los sistemas que utilizamos podrían fallar y que perdamos los avances en el proyecto.
  - Trabajaremos con la planificación, la documentación y el proyecto subidos a servidores en línea (*Google Drive*, *Overleaf* y *GitHub* respectivamente), de modo que si el equipo que estamos utilizando se apaga repentinamente sin guardar cambios, como máximo perderemos los cambios realizados durante el último día de trabajo (o desde la última vez que se guardara el proyecto en local).
  - En caso de que haya pérdida de datos en los servidores en línea, deberemos ser conscientes que el proyecto se desarrollará en varios ordenadores y que cada uno de ellos tendrá su copia local, de modo que simplemente deberíamos continuar con el desarrollo partiendo desde la versión más reciente.
  - Para evitar errores fatales, se realizará mínimo una copia de seguridad semanal en el disco duro del ordenador. Esta copia de seguridad se puede replicar en servidores de almacenamiento o dispositivos externos para que las probabilidades de fallo sean aún menores.

- **Falta de tiempo:** *Impacto alto*. El mayor riesgo de todo proyecto es que este no se pueda terminar a tiempo. Esta falta de tiempo puede ser debido a alguno de los riesgos anteriormente nombrados, u otros más bien fisiológicos como podría ser que el desarrollador se ponga enfermo o se encuentre indispuesto durante varios días.
    - En caso de una falta pequeña de tiempo, disponemos de 20 horas libres en la planificación inicial para poder administrarlas según necesidad en los imprevistos del proyecto.
    - En caso de que se necesitara más tiempo, deberemos ser conscientes que en la planificación solo se han tenido en cuenta los días laborables durante 5 horas cada uno. Se podría plantear como solución aumentar las horas a aplicar diariamente o incluso seguir avanzando en las tareas durante el fin de semana (dado que varias de ellas se pueden realizar sin necesidad de los cascos de realidad virtual).
    - Si el desarrollo del proyecto tuviera tanto atraso temporal que esta solución no fuera viable, deberemos tener en cuenta que, avisando con suficiente antelación, existe la posibilidad de realizar la entrega final del proyecto en abril. Este dato también podría ser útil en el caso de que una mayor inversión de tiempo aumente la calidad del proyecto o surjan requisitos indispensables que no se habían tenido en cuenta inicialmente.
- **Falta de material:** *Impacto alto*. Cualquier avería que no nos permita utilizar los sistemas necesarios, puede atrasar el correcto desarrollo de las tareas.
    - En caso de que el casco de realidad virtual se estropeara, disponemos de versiones más antiguas pero igualmente compatibles que se podrían utilizar hasta conseguir sustituirlo; de este modo podríamos evitar que el desarrollo se detuviera.
    - En caso de que el ordenador de trabajo se estropeara, disponemos de otros equipos igualmente capaces de realizar simulaciones en realidad virtual; simplemente deberíamos descargar los datos del proyecto en uno de ellos. En este caso se podrían perder los últimos avances realizados, así que actuaríamos de acuerdo con el punto *Pérdida de datos*.
    - Una casuística que no se había tenido en cuenta es que no se pudiera acceder a todo el hardware específico necesario por limitaciones de movilidad de la población, como ha sucedido en la fase final del proyecto debido a la pandemia global conocida como *COVID-19*. En este caso se han continuado los desarrollos con la máxima normalidad posible y a la espera de hacer una revisión del proyecto con el hardware una vez finalice el estado de alarma.

5.6. Cambios en la planificación final

El proyecto ha tenido diversas modificaciones de planificación respecto la visión inicial, de modo que las detallaremos y mostraremos la planificación final actualizada.

Me gustaría empezar explicando que a principios de diciembre se observó que el proyecto se iba desarrollando correctamente, pero a un ritmo algo más lento de lo esperado. También empezaron a aparecer dificultades con la personalización de ordenación de herramientas en el primer escenario que se estaba desarrollando, hecho que empezó a retrasar el ritmo de trabajo. Debido a estas causas y con el conocimiento desde antes de empezar el proyecto que esta opción era posible, se decidió posponer la entrega del proyecto de enero a finales de abril. Con estos tres meses de margen, se dispondría del tiempo suficiente para finalizar el proyecto y solucionar los problemas encontrados, de modo que se hizo una replanificación teniendo en cuenta el nuevo margen de tiempo.

Otra variación que se ha aplicado a la planificación ha sido no realizar la tarea número 11, que trataba de incorporar sonidos y música en la aplicación. Este cambio se produjo debido a que, después de analizar el estado del proyecto durante su fase final (que era cuando se planeaba hacer esta tarea), se valoró que era más prioritario e importante para la aplicación que las dos sesiones estuvieran correctamente configuradas e implementadas, reinvirtiendo el tiempo disponible en ello.

Nº	Tarea	Horas
-	**Modificaciones del editor**	**120**
1	Ampliación de los tipos de nodos del editor	80
2	Diseño de sesiones de ejemplo	20
3	Añadir uso de plantillas y guardado de modificaciones	10
4	Adaptación de la interfaz del editor al usuario final	10
-	**Funcionalidades del simulador**	**240**
5	Enriquecer tipos de objeto	50
6	Implementar comportamiento de objetos	80
7	Añadir sintetizador de voz	10
8	Añadir visualización de texto	30
9	Animación de modelos (cirujano, enfermeras, paciente, ...)	20
10	Mejora interacción en Realidad Virtual	10
11	Añadir sonido ambiente	0
12	Puntuación de los resultados	40
-	**Gestión del proyecto**	**160**
13	Revisión del proyecto	40
14	Reunión con cliente	10
15	Documentación	110
-	**TOTAL**	**520**

Tabla 3: *Tiempo final de las tareas*

Con el cambio de calendario, el proyecto finalmente se ha desarrollado desde el día 1 de octubre de 2019 hasta el 26 de abril, que son **142 días laborales**; se han invertido una media de 4 horas diarias, lo que se traduce en un total de unas **550 horas útiles**. Como podemos observar, el total de horas aplicadas en las tareas ha sido de 520 horas, de modo que hemos dispuesto de 30 horas adicionales de margen para que fueran aplicadas en las tareas y los desarrollos más urgentes.

5.7. Planificación final mediante Gantt

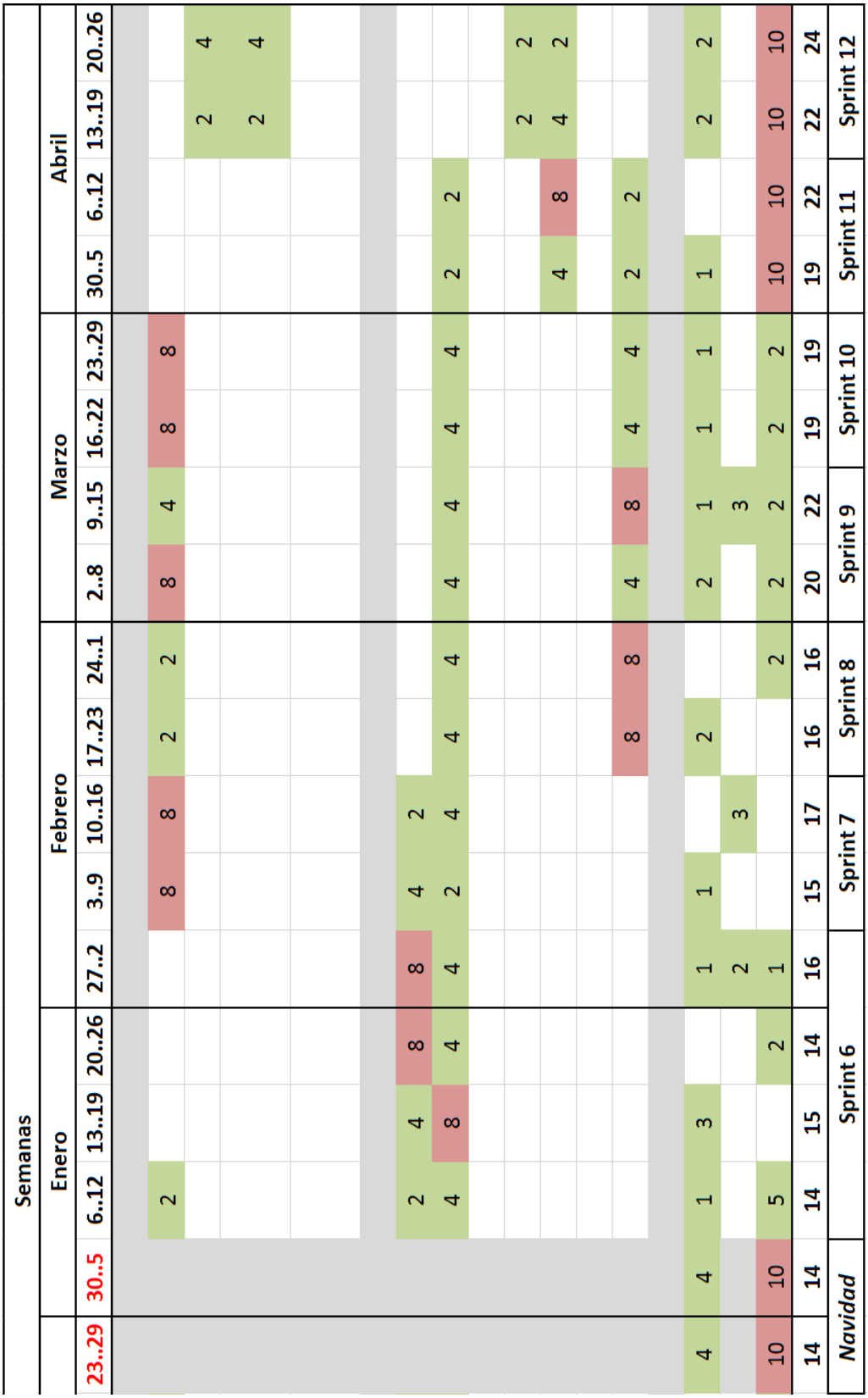


Figura 13: *Planificación final de las tareas por semana y sprint (I)*

Número Tarea	Nombre Tarea	Octubre				Noviembre					Diciembre		
		1..6	7..13	14..20	21..27	28..3	4..10	11..17	18..24	25..1	2..8	9..15	16..22
	Modificaciones del editor												
1	Ampliación de los tipos de nodos			5	5	5	5	2	2	2	2	2	2
2	Diseño de sesiones de ejemplo			5	5			2	2				
3	Añadir uso de plantillas y guardado de modificaciones												
4	Adaptación de la interfaz del editor al usuario final							2	4				
	Funcionalidades del simulador												
5	Enriquecer tipos de objeto			2	2					5	5	5	5
6	Comportamiento de objetos			2	2					5	5	5	5
7	Añadir sintetizador de voz							5	5				
8	Añadir visualización de texto					10	10	4	4				
9	Animación de modelos												
10	Mejora interacción en VR							3	2				
12	Puntuación de los resultados												
	Gestión del proyecto												
13	Revisión del proyecto	2	4		1		1		1	1	1	1	
14	Reunión con cliente							2					
15	Documentación	12	10			1				1			5
Horas de la semana		14	14	14	15	16	16	20	20	14	13	13	17
Sprint		***Organización***		**Sprint 1**		**Sprint 2**		**Sprint 3**		**Sprint 4**		**Sprint 5**	

Figura 14: *Planificación final de las tareas por semana y sprint (II)*

6. Presupuesto

6.1. Costes directos

Una vez definido el proyecto y sus tareas, es un buen momento para plantearnos su viabilidad económica. Hasta el momento hemos definido las horas de trabajo y los materiales que utilizaremos, pero no se ha tenido en cuenta el coste que todo ello conlleva.

Una de las partes más decisivas es el pago a los trabajadores, dado que la persona que desarrollará el proyecto será la pieza clave para que todo suceda correctamente. De acuerdo al convenio que proporciona nuestra universidad para trabajar en empresa, la remuneración mínima establecida es de 9€/hora [30]. Dado que los trabajadores del proyecto somos estudiantes con poca experiencia laboral, la remuneración de trabajo aportado por el convenio es una aproximación bastante buena.

El principal problema que nos encontramos es que la cantidad a pagar es bastante relativa, ya que dependiendo de la tarea específica que se esté realizando (desarrollador, jefe de proyecto, diseñador, etc.) y de la experiencia en el campo, la remuneración media a nivel laboral puede variar bastante. Para hacer un cálculo correcto de la remuneración justa en nuestro proyecto, analizaremos los distintos roles que se desarrollarán, su porcentaje de aplicación y la remuneración media del mercado. Los datos de remuneración se extraerán de la *Guía HAYS* publicada el 2018 [31], que nos muestra una media de los salarios en el mercado separados por ciudad, rol y años de experiencia. Escogeremos los datos de Barcelona y con menos de 2 años de experiencia laboral, ya que son los que más se adecúan a nuestro perfil. Para pasar la remuneración anual a la remuneración por hora, quitaremos la aportación a la seguridad social (35 %) y dividiremos el coste entre los 250 días laborales anuales con una media de 8h de trabajo diarias.

Rol	Porcentaje	Remuneración HAYS	Precio neto por hora
Jefe de proyecto	30 %	45,000 €	16.67
Programador	50 %	27,000 €	10
Diseñador escenarios	10 %	30,000 €	11.11
Tester	10 %	28,000 €	10.37

Tabla 4: *Remuneración media del mercado por rol informático (2018)*

Al realizar el cálculo de la media del precio por hora con los porcentajes mostrados en la tabla, obtenemos que la remuneración neta que debería recibir el desarrollador es de **12.15€/hora**. Como podemos ver, hay cierta diferencia en comparación con el importe que ofrece el convenio de la universidad, de modo que realizaremos el cálculo del coste del desarrollo con ambos importes. Una vez hecho el cálculo, utilizaremos el valor más elevado en los cálculos finales, ya que es el que nos permitirá tener mayor control del máximo coste económico del proyecto.

Nº	Actividad	Horas	Importe 1	Importe 2
1	Ampliación de los tipos de nodos	30	270	364.5
2	Diseño de sesiones de ejemplo	30	270	364.5
3	Añadir plantillas y guardado de modificaciones	10	90	121.5
4	Adaptación de la interfaz del editor al usuario final	10	90	121.5
5	Enriquecer tipos de objeto	30	270	364.5
6	Comportamiento de objetos	30	270	364.5
7	Añadir sintetizador de voz	20	180	243
8	Añadir visualización de texto	10	90	121.5
9	Animación de modelos	20	180	243
10	Mejora interacción en VR	10	90	121.5
11	Añadir sonido ambiente	20	180	243
12	Puntuación de los resultados	20	180	243
13	Revisión del proyecto	20	180	243
14	Reunión con cliente	10	90	121.5
15	Documentación	60	540	729
-	Horas extra	20	180	243
	CPA	**350**	**3,150 €**	**4,252.5 €**
	Seguridad Social (35 %)	**350**	**1,102.5 €**	**1,488.38 €**
	Total CPA	**350**	**4,252.5 €**	**5,740.88 €**

Tabla 5: *Presupuesto inicial por las actividades realizadas (coste directo)*

6.2. Costes indirectos

Además del coste de contratar nuestros trabajadores, también deberemos tener en cuenta todo el coste del material que utilizaremos, contando también con los recursos básicos de luz que gastaremos. Será necesario analizar cómo se amortizará el *Software* y *Hardware* utilizado en el proyecto, para que los recursos se aprovechen correctamente.

- **Amortización Hardware:** Los dispositivos físicos utilizados en el proyecto serán prestados por el grupo de investigación *VIRVIG*, que los adquirió tiempo atrás y los ha amortizado en varios proyectos. De todos modos, es importante tenerlos en cuenta en el presupuesto ya que son los dispositivos clave del desarrollo y queremos que el presupuesto muestre el coste de todo el material del proyecto, dado que podríamos no disponer de ellos o se podrían estropear. Los cálculos del coste se han realizado teniendo en cuenta una amortización de 4 años con un uso del hardware de 8 horas por día laboral.
- **Amortización Software:** Todos los programas utilizados en el proyecto disponen de una licencia gratuita para su uso no comercial y educativo. La herramienta más restrictiva en términos de licencia es Unity, pero no habrá problema dado nuestro proyecto no dispondrá de más de $100K de beneficio en los últimos 12 meses.
- **Gasto de luz:** También será necesario tener en cuenta la acumulación de gasto eléctrico que se producirá durante la ejecución del proyecto. Para justificar la exactitud del cálculo del gasto de luz, es necesario especificar cómo se ha realizado: 0.1€/kW * 0.7kW * 350 horas = 24.5€.

Coste	Base	Importe
Ordenador Windows	1,100	54.69
HTC Vive	500	24.86
Luz	-	24.5
Total CG	-	**104.05 €**

Tabla 6: *Presupuesto inicial por material (coste indirecto)*

6.3. Contingencia e imprevistos

Para disponer de un margen de seguridad económico (en caso de necesitarlo), a los costes directos e indirectos juntos se le añadirá un 15 % extra en relación a contingencias.

Coste	Importe
Total Costes: CPA + CG	5,844.93
Contingencia (15 %)	876.74
Total con Contingencia	**6,721.67 €**

Tabla 7: *Presupuesto inicial de costes con contingencia*

Finalmente, deberemos añadir a nuestro presupuesto una previsión de todos aquellos imprevistos que podrían afectar económicamente al proyecto, de modo que tengamos cierto margen de reacción si alguno de ellos sucediera. En nuestro caso hemos identificado las posibles averías a los sistemas que utilizaremos y la probabilidad de que la entrega del proyecto se deba alargar unos meses a fin de finalizar y pulir todos los aspectos del proyecto.

Imprevisto	Base	Importe
Avería ordenador (10 %)	1,100	110
Avería HTC Vive (10 %)	500	50
Extensión tiempo (20 %)	2,916	583.2
Total Imprevistos	-	**743.2 €**

Tabla 8: *Presupuesto inicial de imprevistos*

6.4. Coste inicial total

Llegados a este punto, ya hemos cuantificado los diferentes costes e imprevistos económicos que podemos controlar en nuestro proyecto antes de su realización. Una vez realizado este estudio, podremos concluir el coste según el cálculo inicial del proyecto:

Coste	Importe
CPA	5,740.88
CG	104.05
Contingencia	876.74
Imprevistos	743.2
Total Inicial	**7,464.87 €**

Tabla 9: *Presupuesto total inicial*

Con todos estos gastos juntos, podemos observar que el presupuesto calculado inicialmente en nuestro proyecto es de **7464.87€**, en el cual se incluyen el salario de nuestros trabajadores, el coste del material y el coste de los posibles imprevistos que puedan suceder.

Los cambios respecto esta valoración inicial deberán gestionarse con suficiente antelación durante la ejecución del proyecto, para asegurar que el proyecto se pueda permitir estas desviaciones. Todo cambio será cuantificados y razonado en el siguiente apartado del presupuesto.

6.5. Coste final

Debido a los cambios de planificación explicados en la *sección 5.6*, deberemos volver a realizar los cálculos del presupuesto teniendo en cuenta los cambios en la cantidad de horas totales del proyecto. De este modo, recalcularemos cada uno de los costes del proyecto para poder observar el coste real de la realización de este y ver el impacto económico de los cambios realizados.

Empezando por los costes directos, se ha utilizado la misma remuneración por hora ya calculada al inicio del presupuesto, pero modificando la cantidad de horas de cada tarea por las finales. De este modo, obtenemos el coste real de desarrollar las tareas al finalizar el proyecto.

Nº	Actividad	Horas	Importe 1	Importe 2
1	Ampliación de los tipos de nodos	80	720	972
1	Ampliación de los tipos de nodos	30	270	364.5
2	Diseño de sesiones de ejemplo	20	180	243
3	Añadir plantillas y guardado de modificaciones	10	90	121.5
4	Adaptación de la interfaz del editor al usuario final	10	90	121.5
5	Enriquecer tipos de objeto	50	450	607.5
6	Comportamiento de objetos	80	720	972
7	Añadir sintetizador de voz	10	90	121.5
8	Añadir visualización de texto	30	270	364.5
9	Animación de modelos	20	180	243
10	Mejora interacción en VR	10	90	121.5
11	Añadir sonido ambiente	0	0	0
12	Puntuación de los resultados	40	360	486
13	Revisión del proyecto	40	360	486
14	Reunión con cliente	10	90	121.5
15	Documentación	110	990	1,336.5
-	Horas extra	30	270	364.5
	CPA	**550**	**4,950 €**	**6,682.5 €**
	Seguridad Social (35 %)	**550**	**1,732.5 €**	**2,338.88 €**
	Total CPA	**550**	**6,682.5 €**	**9,021.38 €**

Tabla 10: *Presupuesto final por las actividades realizadas (coste directo)*

Para los costes indirectos se han utilizado los mismos valores base pero hemos actualizado la cantidad de horas del proyecto en la ponderación de cada una de sus líneas, actualizando los resultados al coste indirecto real. Este cambio se puede entender como una ampliación en el tiempo disponible para amortizar el *Hardware* utilizado en el proyecto, además de ser una contabilización de los recursos básicos que se requerirán a lo largo de todo el proyecto.

Coste	Base	Importe
Ordenador Windows	1,100	85.94
HTC Vive	500	39.06
Luz	-	38.5
Total CG	-	**163.5 €**

Tabla 11: *Presupuesto final por material (coste indirecto)*

Para el caso de la contingencia, como en el cálculo se utilizan como base los datos de costes directos e indirectos, se han recalculado actualizando estos a los nuevos valores calculados. Para el caso de los imprevistos, como estos no se han visto modificados porque sus datos no dependían de la planificación, se han utilizado los mismos que se habían calculado anteriormente.

Es importante seguir teniendo en cuenta tanto la contingencia como los imprevistos en el coste final del proyecto, ya que es un margen que puede ser necesario de utilizar en cualquier momento mientras el proyecto siga siendo desarrollado y modificado.

Coste	Importe
Total Costes: CPA + CG	9,184.88
Contingencia (15 %)	1,377.73
Total con Contingencia	**10,562.61 €**

Tabla 12: *Presupuesto final de costes con contingencia*

Finalmente solo nos queda juntar todos los gastos ya calculados para obtener el coste final del proyecto, que deberá ser compatible con el presupuesto disponible.

Coste	Importe
CPA	9,021.38
CG	163.5
Contingencia	1,377.73
Imprevistos	743.2
Total Final	**11,305.81 €**

Tabla 13: *Presupuesto total final*

Como podemos observar, el coste final con todos los gastos asociados al proyecto será de **11,305.81€**. Si comparamos con los **7,464.87€** del cálculo inicial podemos ver que la diferencia es de **3,840.94€**, aproximadamente un 50 % más de lo calculado inicialmente. Esta diferencia está dentro del rango esperado en el cambio de planificación, dado que el tiempo de desarrollo del proyecto se amplió a casi el doble de lo previsto inicialmente.

6.6. Control de gestión

Una vez establecido el coste del proyecto, deberemos analizar los posibles factores que nos marquen un incorrecto avance de su desarrollo. Realizando este estudio, podremos identificar y rectificar correctamente las posibles desviaciones del proyecto, garantizando así su correcta ejecución.

Los criterios que utilizaremos para cuantificar cada parte serán los siguientes:

- **Desviación del coste de desarrollo:** (Coste Estimado - Coste Real) * Horas Reales
- **Desviación del coste de recursos:** (Coste Estimado - Consumo Real) * Coste Real
- **Desviación en horas del desarrollo:** (Consumo Estimado - Consumo Real) * Coste Real
- **Desviación total en desarrollar las tareas:** (Coste Estimado Total - Coste Real Total)
- **Desviación total de recursos:** (Coste Estimado Total - Coste Real Total)

Estas posibles desviaciones se comprobarán periódicamente en las revisiones de cada *Sprint* para poder detectarlas cuanto antes. El coste económico que se pueda ocasionar se saldará con el porcentaje del presupuesto especificado en base a Contingencias.

En el caso de que el presupuesto para Contingencias e Imprevistos no se utilice, se utilizará la cantidad económica sobrante para el mantenimiento del proyecto posterior a su producción. En el caso que la entidad que ha proporcionado el capital lo requiera, este se devolverá con las condiciones previamente establecidas.

7. Informe de sostenibilidad

Antes de realizar cualquier proyecto, es importante analizar el impacto que este puede tener a corto y largo plazo, de modo que se observe si realmente es viable y proporcionará los beneficios deseados.

Los distintos ámbitos que afectan a la sostenibilidad son el ambiental, el económico y el social. Aunque los tres son muy importantes, hoy en día se está haciendo mucho hincapié en el ámbito ambiental, ya que suele ser el más olvidado por las grandes empresas pero el más decisivo para el correcto mantenimiento del planeta. Si bien tanto el ámbito económico como el social son los que tienen un impacto más directo en el proceso de venta y el beneficio del producto, no tendrá sentido que se intente comercializar si con ello estamos realizando grandes estragos en otros aspectos de nuestro entorno.

En nuestro proyecto hemos intentado conseguir un buen equilibrio entre los tres ámbitos, dando importancia a la sostenibilidad a la vez que resolvemos los problemas planteados por nuestro cliente. Aun así, hemos podido comprobar que es muy importante realizar este análisis durante todo el proceso de desarrollo del proyecto, dado que ciertos aspectos referentes al aprovechamiento de recursos no se habían tenido en cuenta al principio, pero finalmente han sido rectificados para desarrollar correctamente la sostenibilidad del proyecto.

7.1. Estudio de impacto ambiental

Nuestro planeta es un sistema equilibrado en el que la vida de millones de organismos ha podido prosperar, y está determinado por muchos factores, algunos regidos directamente por la naturaleza, y otros que el ser humano es capaz de controlar y modificar a su favor. Es por esta razón que, dada la cantidad de irresponsabilidades que está realizando el hombre hoy en día, es muy importante que se haga un estudio exhaustivo de las consecuencias que tendrán nuestras acciones en nuestro entorno, de modo que se minimicen las consecuencias negativas en la delicada balanza de nuestro ecosistema.

Un punto muy importante a tener en cuenta es realizar un uso ético de la energía eléctrica que consumirá nuestro proyecto. Si bien pueda parecer que nuestro equipo consumirá una cantidad pequeña de electricidad, es importante revisar que los sistemas se encuentren correctamente apagados cuando no se estén utilizando, y se haga un uso eficiente de ellos cuando se vayan a utilizar.

Otro punto importante que deberemos observar es el sistema de administración de residuos que utilizaremos. Cualquier componente utilizado en el proyecto será utilizado y aprovechado mientras dure su vida útil, de modo que se optimicen los recursos invertidos en su fabricación. Cuando un recurso no pueda seguir siendo funcional, nos aseguraremos de depositarlo en la zona de reciclaje oportuna, de modo que se pueda reaprovechar cualquier componente en buen estado y se haga una correcta gestión de residuos.

Si observamos la cantidad de material físico que utilizaremos, fácilmente llegaremos a la conclusión de que es muy poca cantidad: tan solo uno o dos ordenadores y unos cascos de realidad virtual. Aun así, es importante que hagamos un uso responsable de todos los recursos que tenemos a nuestro alcance y de su posterior reciclaje; si cada uno nos responsabilizamos del material que se encuentra a nuestra alcance, conseguiremos que los recursos se aprovechen al máximo de su vida útil, disminuyendo considerablemente su huella ecológica a nivel global.

7.2. Estudio de impacto económico

Otro factor importante a optimizar en el proyecto son los recursos económicos, ya que son los responsables de que el proyecto tenga todo el material necesario para realizarse, y a nivel empresarial puede marcar la diferencia entre realizar uno o múltiples proyectos con el mismo capital.

En nuestro proyecto, el estudio de los recursos económicos requeridos se ha realizado optimizando la cantidad de personas y material solicitado. Es por esta razón, que el proyecto tiene todas las tareas de desarrollo asignadas a una misma persona, dado que el estado y el tiempo inicial no requieren la intervención de más individuos. El material solicitado es el mínimo requerido para su correcto funcionamiento: se trata de un ordenador y unos cascos de realidad virtual como herramientas de trabajo.

En la actualidad podemos encontrar varios proyectos parecidos al nuestro, en los que se utiliza un casco de realidad virtual para la enseñanza y evaluación de procesos. Uno de los factores decisivos a nivel económico es la posibilidad multiplataforma de nuestro proyecto, ya que al utilizar la plataforma gratuita *Unity*, tenemos la posibilidad de exportar nuestro proyecto a cualquier plataforma de realidad virtual disponible actualmente. De este modo, la única restricción para utilizar nuestro proyecto es que el dispositivo tenga la potencia gráfica suficiente y se disponga de dos mandos para poder realizar la interacción.

Hoy en día ya podemos encontrar dispositivos de realidad virtual más económicos que los convencionales, que permiten realizar la simulación sin disponer de un ordenador con alta potencia gráfica [26]. El hecho de disponer de un dispositivo compacto, además de evitar problemas con cables y conexiones, permite ahorrar todo el coste de adquirir un ordenador capaz de ejecutar realidad virtual. Lógicamente, estos dispositivos disponen de una potencia gráfica más limitada que los que utilizan un ordenador especializado, pero para proyectos sencillos como el nuestro sería una alternativa más que suficiente.

7.3. Estudio de impacto social

Uno de los principales objetivos de nuestro proyecto es el de ayudar en el proceso de evaluación y enseñanza en el ámbito médico, mediante la implementación de nuevas tecnologías que aporten más comodidad a todo el proceso. Actualmente, el método de enseñanza que se aplica se basa en la realización de un aprendizaje teórico para después realizar su evaluación práctica. Los principales problemas al utilizar esta metodología surgen en la parte práctica, ya que el material médico suele ser valioso y frágil, y además hay muchos escenarios que son demasiado costosos de simular.

Mediante la simulación de escenarios en realidad virtual, conseguimos eliminar el riesgo de romper material, evitar depender de espacios adaptados y poder realizar cambios instantáneos de escenario. Además, al tratarse de un sistema portátil, ofrece la posibilidad de realizar pruebas con tan solo disponer de un dispositivo de realidad virtual, de modo que los aprendices puedan seguir practicando todo lo aprendido en primera persona y desde cualquier lugar.

Creemos que siempre es importante tener varias metodologías disponibles a la hora de aprender, de modo que se pueda escoger la opción que mejor se adapte a cada estudiante y situación. Este proyecto proporcionará una herramienta complementaria para el proceso actual de enseñanza médica, mediante la cual se podrán realizar sesiones prácticas y evaluaciones a distancia. Si bien la tecnología de realidad virtual aún tiene mucho más potencial por desarrollar, será muy importante que se vaya implementando a un ritmo lento pero seguro, de manera que cada vez se diseñen productos más complejos y más fiables.

Para la correcta utilización de los cascos de realidad virtual, será indispensable leer correctamente las instrucciones de uso y no llevarlos durante varias horas seguidas, para prevenir que el usuario pueda tener molestias en la vista o el cansancio disminuya el rendimiento durante la prueba.

La realización de este proyecto aportará beneficios a varios colectivos, pero también habría que tener en cuenta las aportaciones que realizará a nivel personal. Pienso que siempre es importante realizar proyectos que supongan un reto, ya que es el modo más sencillo de aprender y estar motivado para seguir mejorando. Este proyecto me permitirá entrar en contacto con el ámbito médico, que es fundamental en la vida de todos, ayudando a mejorar en el aprendizaje de sus profesionales a la vez que realizo tareas en las que tengo interés; siempre he creído que es esencial aportar mejoras en la formación actual, porque los estudiantes de hoy serán los responsables en la sociedad del futuro.

8. Desarrollo del proyecto

8.1. Análisis de requisitos

El principal objetivo de este proyecto es la realización de sesiones prácticas configurables de procesos médicos en un entorno de Realidad Virtual. Gracias al soporte y las reuniones realizadas con trabajadores del *Hospital Sant Joan de Déu*, se consiguieron aclarar diversos aspectos técnicos y procedimentales que no se habían tenido en cuenta y que serían clave para incrementar el realismo y la corrección de las simulaciones. En total se hicieron cuatro reuniones, las dos primeras para mostrar el avance técnico del proyecto y realizar demostraciones prácticas con los dispositivos de realidad virtual, y las otras dos en espacios del *Campus Docent Sant Joan de Déu* para aclarar aspectos médicos y grabar demostraciones de los ejercicios que se realizarían en simulaciones docentes reales. En estas reuniones también se determinó que las simulaciones que realizaríamos estarían basadas en los procedimientos necesarios por el instrumentista durante una cirugía de apendicitis, ya que se trata de una cirugía localizada, sencilla y globalmente conocida.

La explicación práctica de la cirugía nos mostró que el instrumentista tiene la responsabilidad de realizar varios procedimientos correctamente: inicialmente deberá proceder a colocarse la bata y los guantes estériles, después deberá recoger las herramientas del interior de una caja metálica estéril y situarlos en la mesa quirúrgica en el orden correcto. Una vez hecho esto, empezará la cirugía y se deberá desinfectar la piel del paciente y colocar el campo quirúrgico en la zona de la cirugía. Durante la cirugía, deberá proporcionar las herramientas al cirujano y ayudar en el uso de algunas de ellas (aguantar separadores, aspirar, etc.). Una vez finalizada la cirugía, se encarga de recoger el material, asegurar que no falte nada y desechar correctamente todo el material.

Una vez realizada la visión general de los distintos procedimientos que deberá realizar el instrumentista durante toda la cirugía, se decidió especificar las partes concretas que se desarrollarán, ampliables según el avance del proyecto. Estos procedimientos fueron los siguientes:

- **Ordenación herramientas en mesa quirúrgica:** Se observó que el procedimiento correcto es recoger las herramientas del interior de una caja metálica estéril. Posteriormente se deberán situar las herramientas correctamente ordenadas encima de la mesa quirúrgica, que deberá estar cubierta con una talla quirúrgica estéril. El conjunto de herramientas necesarias y su orden correcto varía dependiendo de la cirugía.
- **Entrega de material al cirujano:** Durante la cirugía se le proporcionará al cirujano las herramientas que solicite, entregándolas orientadas del modo que se utilizarán. En un escenario real el instrumentista puede realizar un seguimiento de la cirugía y predecir las herramientas que se solicitarán.

La ejecución de estos procedimientos por parte del aprendiz se deberá poder evaluar tanto en el momento de la ejecución como *a posteriori*. Todos los datos recogidos a partir de la simulación serán muy importantes para poder realizar una valoración de la correcta ejecución de la prueba.

8.2. Análisis del desarrollo

Una vez establecidos los requisitos funcionales, será necesario realizar un análisis a nivel técnico de las tareas que deberemos realizar. Este estudio nos dividirá cada procedimiento en los desarrollos esenciales que serán necesarios para cumplir con las necesidades que se han estipulado:

Ordenación herramientas en mesa quirúrgica

- El sistema deberá disponer de las herramientas necesarias para la cirugía, la mesa quirúrgica y la caja contenedora de herramientas estériles.
- Las herramientas se deberán poder agarrar y mover por el sistema virtual.

- La mesa quirúrgica se podrá cubrir con la talla quirúrgica, una vez hecho será correcto dejar las herramientas encima de su superficie.
- El editor deberá permitir establecer el orden correcto de las herramientas usando un sistema de nodos. El orden se deberá comprobar al finalizar la sesión y plasmar los resultados en un registro de la sesión.
- El editor deberá permitir que la caja contenedora sea emplazada encima de cualquier mesa y que se añadan herramientas en su interior.

Entrega de material al cirujano:

El cirujano solicitará las herramientas que va necesitando durante la intervención, de modo que se deberán identificar en la mesa quirúrgica por nombre. Las herramientas deberán entregarse correctamente orientadas y recogerse una vez ya no sean necesarias.

- El sistema deberá disponer de una mesa quirúrgica con las herramientas necesarias para la cirugía actual colocadas.
- El cirujano deberá solicitar las herramientas en un orden lógico, recogerlas para utilizarlas y después devolverlas.
- El editor deberá permitir establecer el orden concreto en el que el cirujano pedirá las herramientas.

Será necesario realizar un control en los resultados del ejercicio, permitiendo conocer datos esenciales para poder valorar la ejecución de la sesión, como pueden ser el tiempo de demora en la entrega de la herramienta y su orientación.

8.3. Implementación técnica

Con el fin de poder contrastar mejor el peso del trabajo realizado, haremos un pequeño repaso a las herramientas que nos proporciona Unity. Si bien los motores de juego proporcionan una base sólida sobre la que poder desarrollar, estos intentan disponer de las herramientas más genéricas posibles para establecer un entorno útil tanto en *2D* como en *3D*. Estas herramientas se componen por: importación de modelos, guardado de configuraciones y jerarquías de objetos, gestión de iluminación, audio, efectos especiales, animación, físicas, interfaces gráficas de usuario, interacción y lógica mediante scripting, entre otras[32]. Aunque se disponga de un punto de partida en todos estos aspectos relacionados con el desarrollo de aplicaciones gráficas, hay que ser consciente que cualquier tipo de modificación en la lógica de los componentes[2] y objetos por defecto, deberá realizarse de manera personalizada aplicando scripts de código C# a los objetos que requieran estas ampliaciones.

Para poder cumplir con los objetivos definidos en los requisitos de las sesiones, se han realizado distintos desarrollos y ampliaciones en el sistema de simulación. Estos desarrollos incluyen tanto aquellos que son primordiales para poder realizar las sesiones (y se definieron a partir de ellas), como los que se derivaron a raíz de necesidades tanto técnicas como conceptuales. Destacar que en el *Anexo* se dispondrá de varias explicaciones a nivel técnico sobre el funcionamiento del proyecto y la herramienta de simulación.

Es importante tener un control de los cambios realizados y el razonamiento de su necesidad práctica, ya que este tipo de información está interiorizada por los actores que gestionan el proyecto pero es totalmente opaca para todo aquel que desee analizar el proyecto o modificarlo.

[2] Los componentes son scripts prefabricados en Unity que se pueden asociar a GameObjects para aplicarles la lógica que implementen

8.3.1. Configuración del modelo de Mesa quirúrgica

El modelo de mesa quirúrgica que se utilizará en el sistema proviene de los *assets* de *Unity* que se utilizan en el proyecto, dado que estos cumplen con las características que buscamos. Nuestra responsabilidad, entonces, será realizar la configuración pertinente para que su comportamiento sea el esperado.

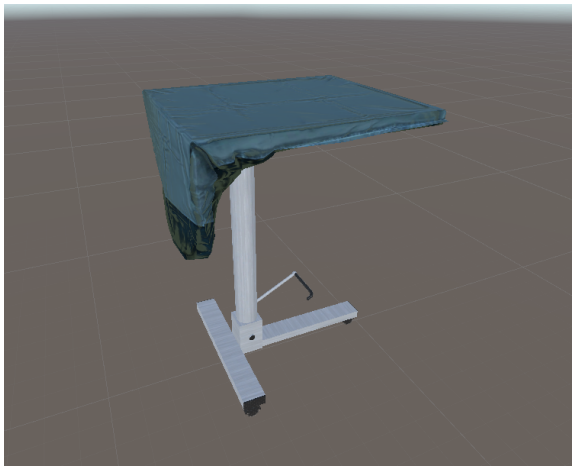
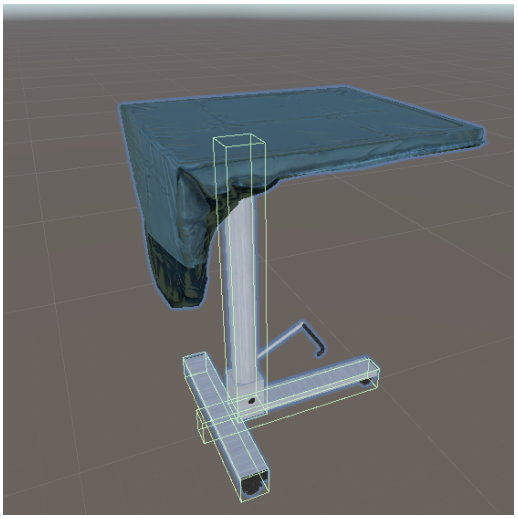


Figura 15: *Modelo utilizado para la mesa quirúrgica*

Empezando por el motor de físicas de *Unity*, se ha configurado el sistema de colisión de los elementos con la mesa. Para ello, se ha aplicado un sistema múltiple de *Colliders* [3] simples alrededor del tronco, la base y la superficie, de modo que se asemeje a la forma de la mesa pero tenga la mínima repercusión posible al rendimiento de la aplicación.



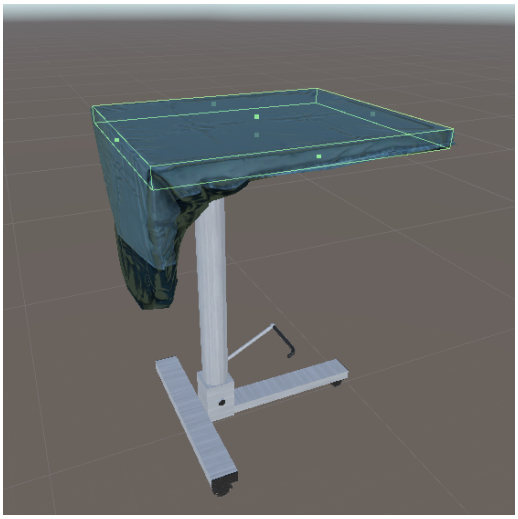


Figura 16: *Colliders configurados en la mesa quirúrgica*

[3] Componente con forma geométrica simple que permite definir el volumen del GameObject que lo contiene. Se puede utilizar junto a un Rigidbody para colisiones con físicas o como disparador de eventos

Para conocer de modo sencillo cuándo una herramienta está encima de la mesa y cuándo no, se ha aplicado un sistema basado en disparadores. Para conseguirlo, se ha situado un *Box Collider* de tipo *Trigger* con la misma forma que la superficie de la mesa, pero con un poco más de altura. De este modo, cuando la herramienta entra en contacto con el *Trigger*, se sitúa como hija de un *GameObject* [4] específico de la mesa, y cuando sale se la quita de la jerarquía de objetos de la mesa. Usando este método conseguimos tener controladas todas las herramientas que hayan sido situadas encima de la mesa en un solo *GameObject*, y al estar dentro de la jerarquía de la mesa nos aseguramos que, mientras estén encima suyo, cualquier movimiento de la mesa también se les aplicará, algo bastante útil si en un futuro se desea permitir mover la mesa por el escenario.

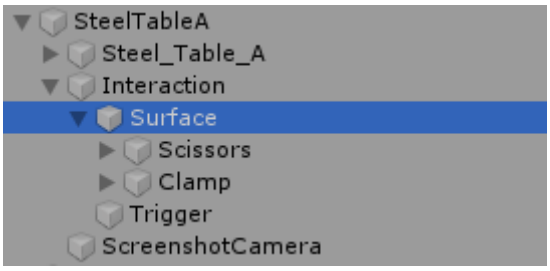


Figura 17: *Jerarquía interna de la mesa*

8.3.2. Configuración del orden correcto de las herramientas mediante nodos

Necesitamos establecer un modo de configurar el orden correcto de las herramientas en la mesa quirúrgica utilizando el sistema de nodos incluido en el proyecto. Antes de nada, realizamos un estudio sobre el tipo de comparaciones que mejor podía encajar en el escenario que se nos planteaba; estábamos buscando un sistema genérico que siguiera siendo válido al añadir o quitar herramientas del sistema, de modo que se descartó tener que establecer el orden de cada herramienta de la escena a mano.

La solución que encontramos a este dilema fue utilizar un sistema basado en reglas de precedencia por parejas de herramientas; es decir, establecer qué tipo de herramienta debe ir antes de qué otro tipo de herramienta del sistema. Se comprobó que, al utilizar este tipo de reglas, se puede definir el orden correcto de las herramientas en los escenarios planteados, de modo que marcando por incorrectas solo aquellas reglas que no se cumplan podemos generar escenarios fácilmente ampliables una vez establecidas las reglas correctas. Si por ejemplo quisiéramos determinar que el orden correcto sea tener primero un bisturí, luego unas tijeras y finalmente unas pinzas, se debería configurar el nodo estableciendo las reglas: bisturí debe ir colocado antes que tijeras; tijeras debe ir colocado antes que pinzas.

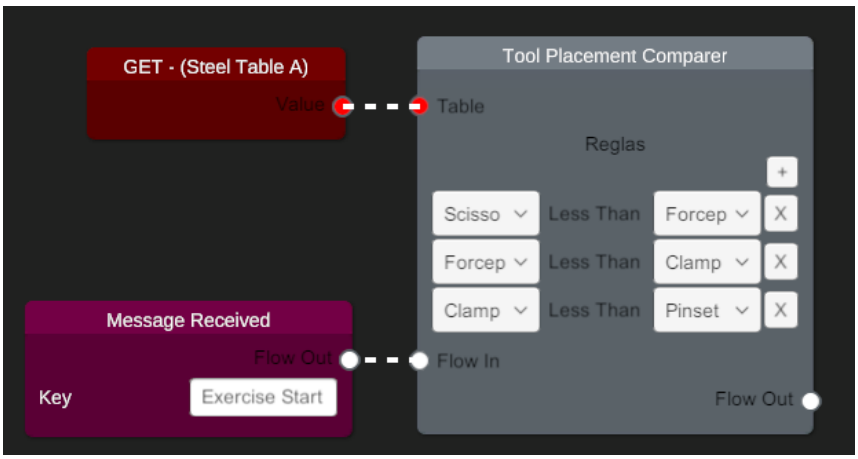


Figura 18: *Ejemplo de uso del nodo comparador de reglas*

[4] Clase básica utilizada por los objetos en las escenas de Unity, que permite que se le añadan propiedades, componentes y scripts, además de poder anidar otros GameObjects como hijos suyos en la jerarquía de la escena

El problema, ahora, estaba en plasmar esta idea mediante el uso de nodos, ya que el sistema actual no contemplaba el soporte de listas de elementos, y hasta el momento los nodos solo se habían compuesto de campos de texto, *checks* y seleccionables. Para ello, se aprovechó una de las características de *xNode* llamada *dynamic ports* (también referido como *instance ports*), que permite instanciar entradas y salidas del nodo de manera dinámica en tiempo de ejecución, utilizando como identificador un campo de texto.

En nuestro caso utilizaremos este campo de texto para almacenar datos que nos facilite la identificación del tipo de elemento y la localización del registro, separándolos los datos con una almohadilla; de modo que el resultado sería: "#List#NombreVariable#TextoElemento#id". De esta manera, cuando se están generando los nodos (clase *UGUIBasicNode*), podemos controlar que nos estamos refiriendo a elementos de una lista mirando si el nombre empieza por "#List", sabemos que proviene de la variable "NombreVariable" del nodo, que se quiere mostrar el texto "TextoElemento" y que tiene "id" como identificador único (generado por *Unity*).

Vista la importancia de este identificador de texto, hemos decidido almacenar los datos de las listas en un *Diccionario* con este texto como clave de registro. El diccionario deberá almacenar parejas de elementos de la enumeración de las herramientas, pero como en la práctica vimos que las estructuras de datos de C# no se terminaban de ajustar a nuestras necesidades (tanto *KeyValuePair<TKey,TValue>* como *Tuple<T1,T2>* son estructuras no mutables), tuvimos que implementar una clase sencilla *MutableTuple<T1,T2>*, cuyos elementos *Item1* e *Item2* eran accesibles y modificables sin complicación.

Las principales ampliaciones a nivel del código base de los nodos se hizo en la clase *UGUIBasicNode*, ya que es la encargada de instanciar todos los elementos que componen un nodo. Se añadieron dos tipos de comportamiento además del que ya estaba definido: el primero para instanciar el elemento de lista (dado por el diccionario) y el segundo para instanciar los elementos que incluiría la lista (identificados por ser dinámicos y su nombre empezar por "#List").

A nivel de interfaz, se generaron el prefab[5] de la lista y el prefab de sus elementos. El prefab de la lista está compuesto por el título, la lista de elementos y el botón de añadir; mientras que el prefab de los elementos de la lista está compuesto por dos campos de propiedad (reutilizando los que utilizan los nodos básicos), con un campo de texto entremedio de ellos y un botón de eliminar registro al final.

Figura 19: *Prefab para la lista y sus elementos, utilizados en el nodo de reglas*

Los botones de añadir y quitar registro tienen la responsabilidad de llamar a las funciones definidas en el propio nodo, que se encargan de añadir/eliminar la instancia en el nodo y el dato en el diccionario, respectivamente. El uso del diccionario también nos ayuda mucho en el hecho de eliminar registros, ya que si en una lista eliminas un elemento intermedio, todos los índices posteriores varían, pero como en el diccionario cada elemento tiene su identificador único, no existe impacto entre ellos al añadir o eliminar registros.

[5] Tipo de objeto de Unity que permite guardar un GameObject con su configuración y la de todos los que estén como hijos suyos en la jerarquía de la escena

Finalmente, necesitamos modificar las funciones encargadas de leer y modificar los datos entre la pantalla y el código, ya que estas usan la *Reflexión* de C# para navegar por los tipos de las variables utilizadas hasta llegar al dato buscado. La navegación se realiza estableciendo una "ruta" para encontrar el dato, pero hasta el momento solo estaba contemplada la navegación mediante atributos de la clase. Hemos tenido que ampliar esta funcionalidad para que también se pueda realizar una navegación por propiedades de la clase y permitir el acceso por clave a elementos de un diccionario (modificaciones realizadas en las funciones *GetValueOf* y *SetValueOf* de la clase *ReflectionUtilities*). De este modo, por ejemplo, ahora podemos acceder al primer elemento de la tupla con clave "#List#id1" del diccionario reglas realizando una lectura por Reflexión con el parámetro: "reglas/[#List#id1]/Item1".

8.3.3. Algoritmo de comprobación de las reglas en la mesa

Una vez establecido el sistema por reglas de precedencia que marcarán cómo deberán estar ordenadas las herramientas, deberá realizarse la comprobación de manera automatizada cuando se ejecute el nodo.

Para realizar esta comprobación, empezaremos dividiendo el *Collider* de la superficie de la mesa en la cantidad de filas que se haya configurado (N). Lo llamamos dividir porque se ocupará el mismo espacio que la superficie, pero realmente estaremos generando N nuevos *Colliders* con el mismo tamaño.

Figura 20: *División de la mesa en filas para comprobar las reglas*

Acto seguido, obtenemos la lista de herramientas situadas encima de la mesa y procedemos a situar cada herramienta en una de las N filas. Esta clasificación se realiza mirando encima de qué división se encuentra el centro de coordenadas del *Collider* de cada herramienta, de modo que se determine inequívocamente que cada herramienta estará situada solo en una de las filas. El proceso de clasificación termina realizando una ordenación de cada fila utilizando como referencia el eje paralelo a la longitud de las filas.

Finalmente se selecciona el tipo de cada herramienta para compararlas entre ellas, comprobando qué reglas no se cumplen y cuántas veces sucede. Esta comprobación se realiza por cada regla en cada una de las filas, de modo que los resultados serán independientes entre filas.

Los resultados finales obtenidos se guardan en el fichero de registro de la sesión, de modo que se podrán utilizar para una evaluación posterior y contrastar junto a las imágenes de los cambios en las herramientas de la mesa.

8.3.4. Recolección de resultados (evaluación)

Una de las partes más importantes del proyecto es desarrollar un sistema capaz de recoger resultados de las pruebas realizadas, de modo que las ejecuciones se puedan analizar en profundidad más adelante. Para poder cumplir con esta necesidad, se han implementado algunos sistemas que son capaces de registrar sucesos y datos a lo largo de las sesiones.

En primer lugar, se ha implementado un sistema de registro de sucesos (*log*) que permite guardar los datos de cada ejecución de la sesión en un fichero de texto. Este sistema está compuesto por una cabecera con el nombre de la sesión y la fecha y hora de inicio; seguido por todos los registros de eventos que se han almacenado, cada uno de ellos con el instante de tiempo en el que sucedió (relativo al inicio de la sesión).

Para que el proceso de implementar mensajes de registros sea más sencillo para los desarrolladores, se ha establecido la clase que lo implementa como *Singleton*, permitiendo que solo se instancie una vez y facilitando el control de la concurrencia de sus ficheros. De este modo, se evita la incomodidad de tener que inyectar el sistema logger en cada una las clases en las que se necesite utilizar, simplificando su funcionalidad a proporcionar directamente los datos a registrar a la instancia habilitada en la simulación.

Hasta el momento se han configurado eventos de prueba que almacenen tanto registros básicos como históricos de resultados, para poder verificar que el sistema sea fiable y válido. Entre ellos podemos encontrar:

- Cada vez que se abre o se cierra el menú de pausa
- Si una herramienta ha entrado en contacto con el suelo
- Resultados de la comprobación de reglas en la mesa quirúrgica
- Resultados del proceso de entrega de material al cirujano

Un ejemplo del fichero de resultados generado durante la ejecución de la sesión podría ser el siguiente:

```
////////////////////////////////////////////////////////////////////////////
/// Nombre sesión: NiceDecoration      Fecha de ejecución: 20/04/2020 18:46:35 ///
////////////////////////////////////////////////////////////////////////////

0:00:02,957 - Se ha cerrado el menú principal
0:00:10,402 - Se ha abierto el menú principal
0:00:11,294 - Se ha cerrado el menú principal
0:00:32,421 - La herramienta Scissors ha entrado en contacto con el suelo.
0:00:52,841 - La herramienta Scissors ha dejado de tocar el suelo.
0:03:34,402 - Se ha abierto el menú principal

*********************************************************
Comprobación de reglas en mesa quirúrgica
Time: 0:01:56,201
*********************************************************
Todas las reglas se han comprobado sin errores.
*********************************************************
* FIN * Comprobación de reglas en mesa quirúrgica
*********************************************************
```

Paralelamente, se ha implementado un sistema de captura de imágenes de la sesión que realiza fotografías de la mesa quirúrgica cada vez que se deja o se recoge una herramienta, desde una vista superior. Mediante la recogida de estas imágenes, podemos realizar un seguimiento de la evolución de la mesa a lo largo de la sesión y comprobar visualmente cuál era el estado final. Estos datos pueden ser de mucho valor en la evaluación posterior de la sesión ya que permitiría realizar una evaluación manual que complementaría la evaluación automática realizada por el sistema.

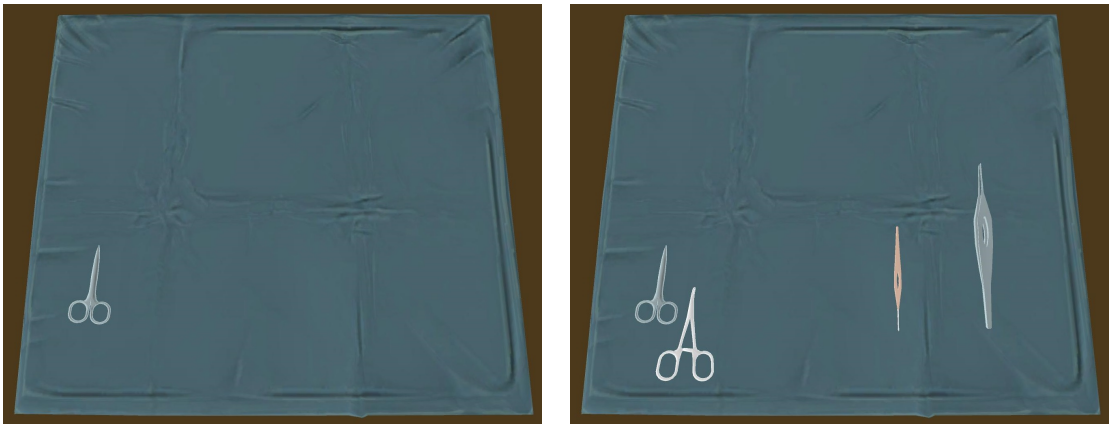


Figura 21: *Capturas de pantalla del proceso de ordenación de la mesa*

8.3.5. Almacenamiento de las herramientas

Las herramientas deberán ser recogidas del interior de una caja metálica estéril para situarlas en la mesa quirúrgica, ya que es la metodología que se usa en escenarios reales.

El modelo de la caja metálica se ha generado utilizando *Blender*, haciendo un modelo de la caja y otro de la tapa. Para generarlos se han utilizado formas básicas como cubos y cilindros, ya que juntándolos y utilizando restas de volumen hemos conseguido las formas que buscábamos. Se han aplicado texturas procedentes de imágenes de cajas metálicas reales para aumentar la calidad y el realismo del modelo.

Una vez modeladas la caja y su tapa, las hemos añadido a nuestro proyecto de *Unity* configurándolas como un *Prefab* para que formen la caja completa. Se les ha añadido los scripts básicos de interacción, además de permitir abrir la caja de forma articulada con el script *VRTK Rotator*.

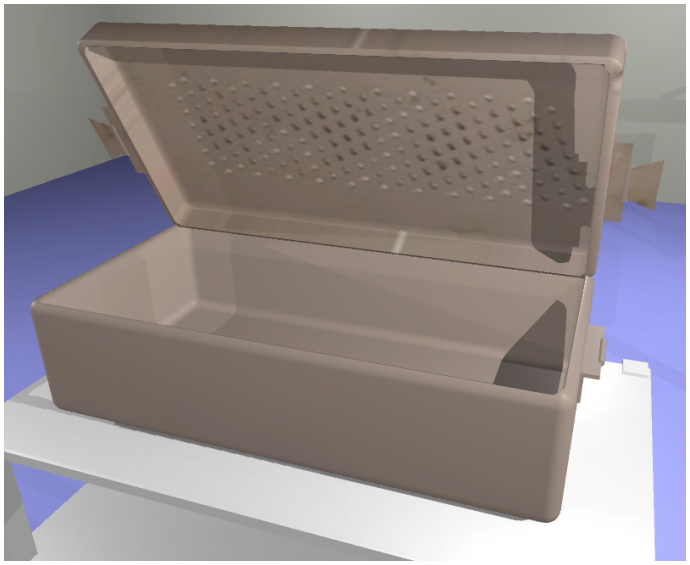


Figura 22: *Caja metálica con la tapa semiabierta*

Para conseguir que las herramientas se puedan añadir desde el editor, se han situado varios *spawnpoints* en el interior de la caja, que son puntos de coordenadas utilizados en el *script* de configuración como referencia al instanciar las herramientas. Redefiniendo las funciones específicas *CanHoldItem* y *HoldItem* utilizadas por el editor podemos, respectivamente, mostrar visualmente en el editor si caben más herramientas y definir cómo se instancian, permitiendo realizar una lógica personalizada a las necesidades de cada contenedor.

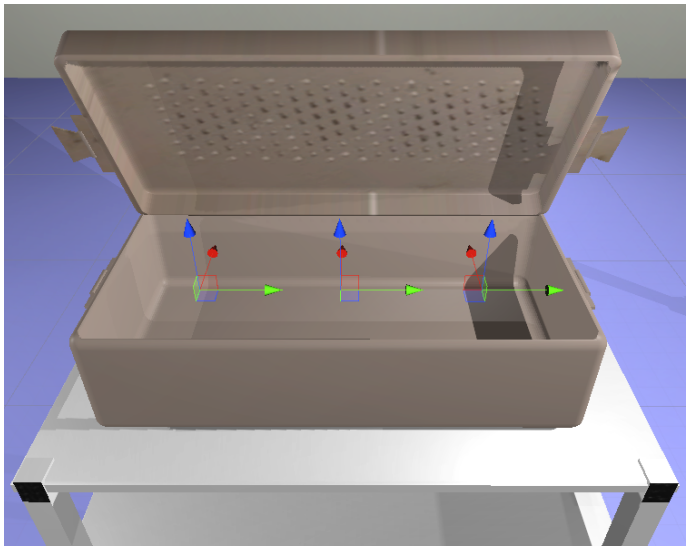


Figura 23: *Muestra de los spawnpoints de la caja metálica*

8.3.6. Solicitud de material por el cirujano

El modelo de cirujano que se utilizará para generar el prefab de la sesión es el que se encuentra en los *assets* de pago del proyecto, al cual se le han añadido los componentes y scripts que permitirán la interacción con el usuario. El componente básico para poder entregar y recoger herramientas es *VRTK Snap Drop Zone*, que se ha situado en la mano derecha del cirujano y cuando está habilitado permite que las herramientas se suelten y se recojan de esa posición. El script de configuración que se ha añadido al objeto del cirujano contiene toda la lógica que deberá realizar el objeto en la escena, incluida la implementación de los parámetros que se utilizarán para gestionar el flujo de las animaciones.

Partiendo de este modelo como base, se han implementado diversas animaciones para utilizarlas en el proceso de solicitud de las herramientas. Estas animaciones incluyen todos los movimientos que necesitará el cirujano en el proceso de solicitar herramientas: se empezará por un movimiento básico (para dar vida al personaje) y al solicitar herramientas se realizará una animación con el brazo derecho para recogerla. Al situar una herramienta en la mano, se comprueba si es la que se quería para que en caso negativo se proteste y el usuario tenga que entregar una diferente. En caso positivo el cirujano realizará una animación de confirmación y trabajará en el paciente; cuando termine devolverá la herramienta, que se deberá recoger y esperar a que se solicite alguna otra.

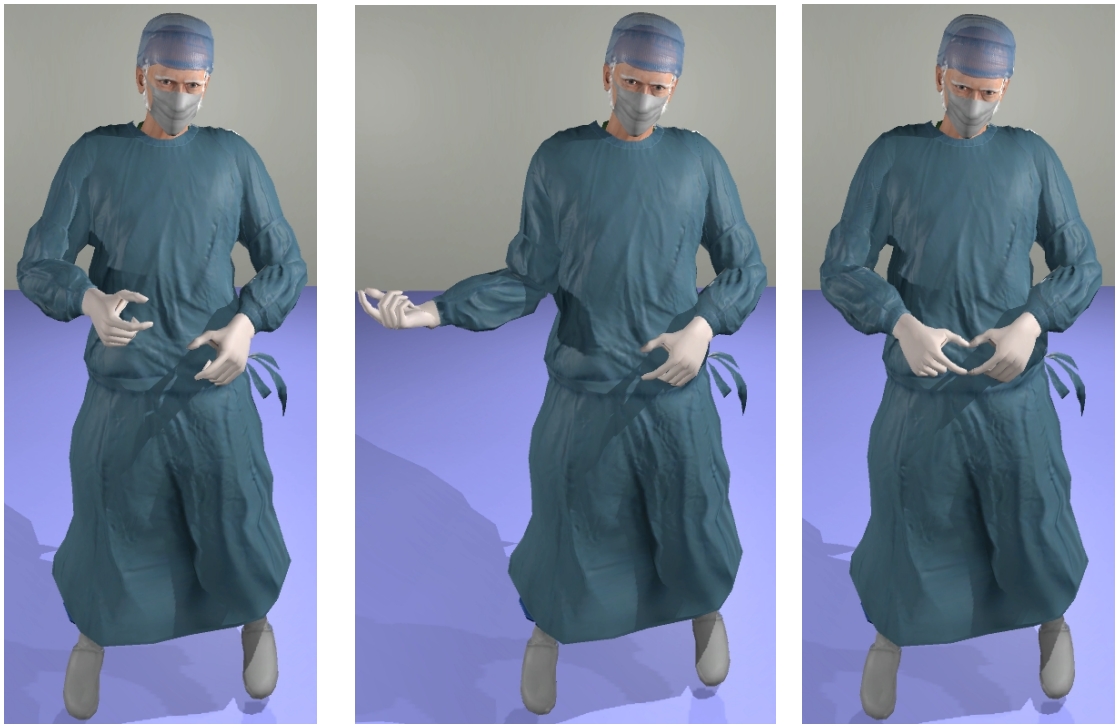


Figura 24: *Animación utilizada por el cirujano para pedir herramientas*

El nodo implementado permite gestionar cuando se desea que el cirujano empiece a solicitar las herramientas. Al empezar esta acción, el cirujano pedirá cada uno de los tipos de herramientas que se encuentren en la sesión en un orden aleatorio, de modo que se utilicen todas y el usuario las tenga que reconocer y entregar lo más rápido posible. Este nodo ha sido diseñado a modo de ejemplo inicial, pero se planea ampliarlo en la futura revisión de la interacción para que también permita especificar el orden concreto en el que se quieren solicitar las herramientas.

8.3.7. Modificación de la textura en el modelo del paciente

Para dar más realismo durante la simulación de la intervención, el modelo del paciente tendrá una talla quirúrgica que dejará el abdomen a la vista, para que a través de ella se pueda visualizar el avance de la operación.

Para realizar esta tarea, se modificará la textura en el interior de la talla para que muestre imágenes de diferentes momentos de la intervención a medida que el cirujano vaya pidiendo herramientas.

Cabe destacar que las imágenes utilizadas no son de cirugías reales, sino que se trata de una simulación realizada con material biológico y sangre falsa en el centro docente *Sant Joan de Déu*, de una manera parecida a como se podría realizar en una simulación con estudiantes de enfermería.

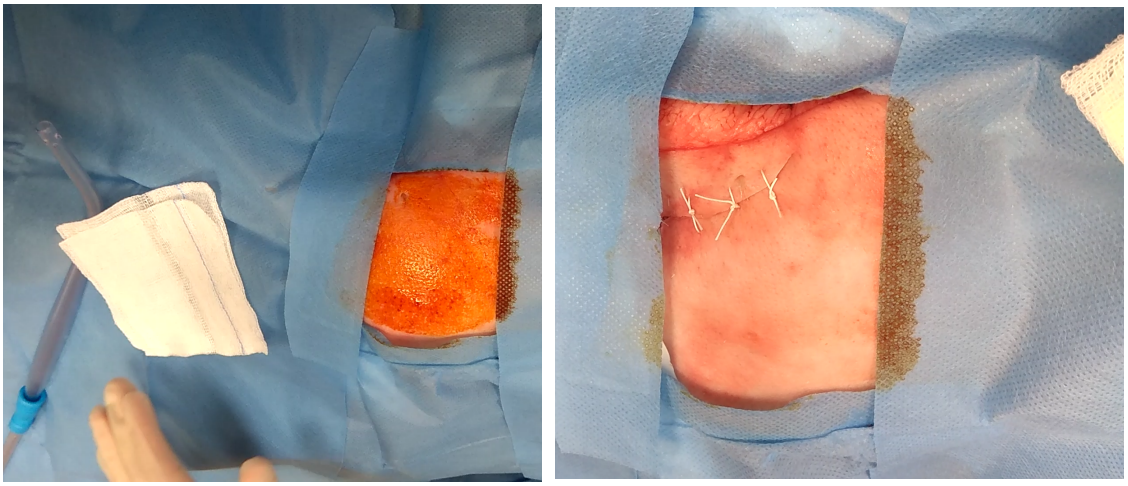


Figura 25: *Simulación de cirugía con material biológico*

8.3.8. Nodos adicionales

Varios de los desarrollos secundarios que se han realizado, tienen nodos asociadas que permiten configurarlos y coordinarlos con el flujo de la simulación.

- **Ampliación nodos de aritmética:** Se han incluido en el editor nodos para realizar aritmética básica (*Sum, Mult, Div*). Como estos nodos podían no aportar todos los cálculos necesarios, también se ha añadido el nodo *MathExpression*, que acepta aritmética más compleja (*pow, sqrt, sin, cos, ...*) y el uso de las variables numéricas del editor, de modo que toda la lógica numérica se compacte en un solo nodo. Para hacer posible esta interpretación matemática, se ha utilizado la implementación en C# de la clase *ExpressionEvaluator*[33], configurando el nodo para que le proporcione los valores de las variables numéricas a la hora de realizar los cálculos.

- **Mensajes de texto en el lateral del visor:** Con el objetivo de que el sistema permitiera aportar una guía de las acciones a realizar por los usuarios a lo largo de la simulación, se incorporó en la parte superior derecha de la pantalla un campo de texto. El mensaje de texto se puede modificar introduciendo el texto requerido en el nodo *DisplayHint*, haciéndose invisible cuando el mensaje es un texto vacío.

- **Bocadillo de texto en la escena:** Se solicitó poder mostrar mensajes de texto en la escena, de modo que la solución más sencilla encontrada fue mostrar un bocadillo de texto alrededor del objeto deseado. Para poder realizar esta tarea, se implementó el nodo *DisplayMessage*, que permite establecer el objeto de la escena alrededor del cual se mostrará el mensaje, el texto del mensaje y el evento que aplicará la modificación. Para ocultar el bocadillo de texto, simplemente hay que ejecutar el nodo con el mensaje de texto vacío. La lógica de este nodo tiene más complejidad de la que aparenta, dado que el bocadillo de texto deberá mostrarse al lado del objeto establecido que puede ser de cualquier tamaño, y deberá evitar la colisión con paredes y otros elementos que dificulten su lectura.

Figura 26: *Bocadillo de texto durante una simulación*

8.3.9. Otros desarrollos derivados

En este apartado detallaremos los desarrollos que también se han realizado en el proyecto pero que tienen un menor impacto en la configuración de las sesiones:

- **Añadir sintetizador de voz:** Para complementar los sistemas de comunicación por mensajes de texto, se requería incluir en el proyecto un sistema de síntesis de voz a partir de texto (conocido como *Text-to-Speech (TTS)*). Para conseguirlo, se estuvieron investigando las diferentes implementaciones disponibles para escoger la alternativa que mejor equilibrio calidad-precio aportara al proyecto (dado que la mayoría de sistemas gratuitos generaban sonidos de muy mala calidad). Después de diversas pruebas, se observó que la opción que mejores resultados aportaba era la implementación que usaba el sistema de síntesis de voz de *Microsoft* [34], dado que mostraba una calidad elevada en cuanto a sonido y pronunciación, a la vez que permitía un uso gratuito con las únicas limitaciones de solo poder realizar 1 petición de síntesis simultánea y un máximo de entre 0.5 y 5 millones de caracteres gratis al mes (dependiendo de la opción escogida: Neuronal o Estándar).
- **Duplicación de nodos:** Se ha habilitado en el editor de nodos la opción de duplicar tanto nodos sueltos como una selección de nodos; esta acción se encuentra accesible desde el menú que se despliega al realizar click derecho en el editor. Aunque la implementación actual no mantiene las conexiones existentes entre los nodos, aportará mucha más comodidad al usuario que modifique la sesión cuando este desee aplicar una misma lógica de nodos varias veces en el flujo.

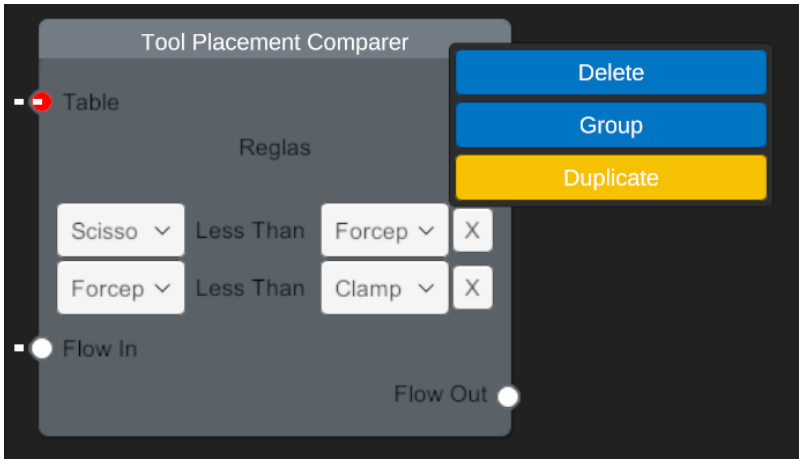


Figura 27: *Opción de duplicar nodo en el menú contextual*

- **Modificación interacciones:** Al realizar pruebas con el simulador había dificultades para entender los controles configurados, de modo que algunas asignaciones se modificaron para que fueran más intuitivas para los usuarios. Los cambios realizados fueron que el rayo desde el controlador se mostrara al contacto con el *trackpad*, que la confirmación de teletransporte por la escena y selección en el menú de pausa se hiciera presionando el botón del propio *trackpad* y que la recogida/agarre de objetos se realizase con el grip.

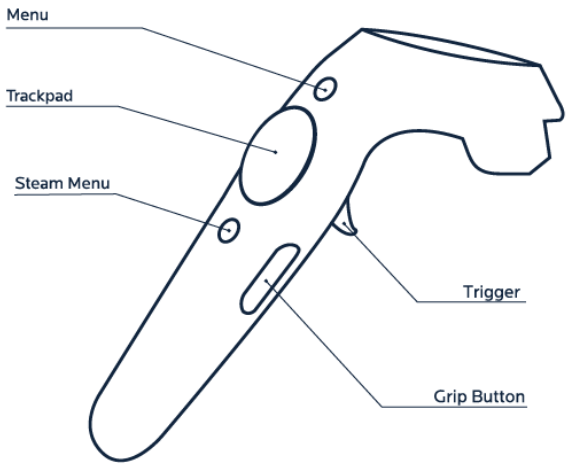


Figura 28: *Distribución de botones en el controlador de Realidad Virtual utilizado*

- **Colisión de las herramientas con el suelo:** En las primeras pruebas del sistema se comprobó que las colisiones con los objetos de tipo herramienta no se detectaban correctamente, haciendo que objetos básicos como el suelo fueran atravesados por ellas. Después de realizar pruebas, se detectó que el problema provenía de la configuración del sistema de físicas en el objeto, de modo que se modificó el componente *Rigidbody* [6] de todas las herramientas para que el cálculo de la colisión se hiciera correctamente (opción "Continuous Speculative").
- Se ha añadido una diferenciación entre las superficies sobre las que colocar objetos (*surface*) y los contenedores en los que solo se les pueden colocar herramientas (*container*).
- Se han añadido diversas decoraciones y herramientas al sistema para su utilización, entre ellas algunas que se obtuvieron modelándolas a través de imágenes realizadas a material real del *Hospital Sant Joan de Déu*.

[6] Componente que permite que a un GameObject se le aplique la lógica del motor de físicas de la aplicación

Figura 29: *Contenedores de material médico modelados a partir de imágenes*

- A todos los personajes incluidos en la escena, se les ha configurado una animación por defecto (*Idle*) para aportar más naturalidad a la sesión.
- Se ha realizado una adaptación de la interfaz del menú principal. Entre los cambios se ha forzado la ordenación de las sesiones del sistema por fecha de creación, se ha añadido la opción de poder crear sesiones a partir de las plantillas de sesión disponibles y se ha añadido una ventana de confirmación a la hora de eliminar sesiones.

8.4. Dificultades encontradas

Si realizamos un análisis completo a cualquier proyecto en el hayamos trabajado, será imposible encontrar alguno en el que no hayan surgido problemas, bloqueos o replanificaciones. Esto tampoco tiene que ser observado como algo negativo, ya que es imposible conseguir controlar todas las opciones y variables desde el principio del proyecto, así que deberemos aprovechar la capacidad que tenemos como seres humanos de resolver y adaptarnos a los problemas que se nos presentan.

Este planteamiento también se ha aplicado a nuestro proyecto, que ha presentado dificultades y problemas a lo largo de todo el proceso de desarrollo, aunque por suerte se han podido solucionar con mayor o menor impacto en la planificación. A continuación se realizará una síntesis de las dificultades encontradas a lo largo del proyecto que no hayan sido explicadas en otros puntos:

- Ha habido ciertos conocimientos avanzados de *Unity* de los que tenía poco conocimiento pero han sido necesarios para el correcto desarrollo de las distintas partes del proyecto, de modo que se tuvo que emplear tiempo de investigación y pruebas para aprender su funcionamiento y poder utilizarlos. Algunos de estos conocimientos han sido conceptos muy concretos de Unity, como las diferentes metodologías de colisión utilizadas por el sistema de físicas; mientras que otros han estado relacionados con VRTK o el propio lenguaje C#.
- El proyecto base desde el cual se ha partido y que debía modificarse, disponía de documentación orientada al usuario final pero con poca explicación técnica, de modo que para entender todas las funcionalidades que se debían modificar se ha tenido que realizar constantemente análisis mediante la lectura del código. Este tipo de análisis es más costoso y lento, de modo que ha afectado negativamente a la planificación del proyecto.

- Uno de los problemas a la hora de analizar el proyecto ha estado que el funcionamiento del código puede estar condicionado a parámetros y valores configurados desde el propio inspector de Unity, de modo que entramos en un proceso de análisis por la jerarquía de objetos y componentes para encontrar todos los objetos que utilizan el script deseado, intentando descubrir el sentido que tiene su utilización y los parámetros configurados en la lógica de la aplicación.
- Ciertas lógicas de los assets utilizados están ocultas en la implementación de los paquetes. Por ejemplo, costó varias horas descubrir que el sistema de agarre de *VRTK* guarda en qué posición de la jerarquía de la escena de Unity se encontraba el objeto, y lo restaura de manera forzada cuando este se suelta. Esta lógica iba en contra de la que se había implementado para la mesa quirúrgica, haciendo pensar que había un problema en su funcionamiento cuando en realidad el problema surgía de un elemento ajeno.
- La implementación personalizada del nodo dinámico de reglas supuso el mayor foco de problemas y bloqueos de todo el proyecto, absorbiendo mucho más tiempo para su desarrollo del esperado inicialmente. Esta es la razón por la que el desarrollo del propio nodo y su lógica se alargó durante varios sprints, sucediendo el mayor de los bloqueos durante la época de navidad. Como el uso de listas y elementos compuestos no estaba contemplado en el sistema, se tuvo que realizar una implementación desde cero y modificar varias clases que eran críticas para el correcto funcionamiento de todos los nodos.

9. Conclusiones

Para poder concluir correctamente un proyecto, se debe realizar un análisis del trabajo realizado durante los últimos meses para poder documentarlo, dejando constancia de los objetivos cumplidos y las decisiones tomadas; pero también se deberán tener en cuenta las posibles proyecciones y ampliaciones que se puede requerir del proyecto en un futuro, incluyendo tareas de mantenimiento.

El principal objetivo del proyecto era la realización de sesiones de entrenamiento en un entorno de Realidad Virtual, cuyo cumplimiento se hace implícito al haber realizado la explicación de las sesiones que hemos implementado al sistema. Estas sesiones cumplen con la lógica que se definió en la definición de los dos tipos de sesiones que se establecieron como objetivo principal del proyecto.

Otra necesidad clave era la evaluación de las pruebas, satisfecha gracias a los sistemas de recogida de resultados implementados. Los valores numéricos de las sesiones que se quieran recoger, pueden ser obtenidos realizando un desarrollo que los plasme en el fichero de sucesos y resultados. También se ha demostrado la flexibilidad del sistema en cuanto a posibilidades de recogida de datos, dando la opción de tener un histórico de imágenes con los cambios realizados en la mesa quirúrgica.

La posibilidad de personalización de las sesiones viene dada por la posibilidad de representar su lógica mediante el uso de nodos. El nodo más destacado en cuanto a personalización de sesiones es el comparador de reglas de precedencia, que permite configurar la combinación correcta de herramientas en la fila de una mesa quirúrgica. De todos modos, como el sistema permite añadir tantos nodos como sea necesario siempre y cuando se implemente su lógica a nivel de código, cualquier procedimiento que el usuario final desee aplicar a las sesiones, podrá aplicarse siempre que se pueda representar mediante un flujo de nodos y todos los nodos necesarios estén implementados.

Trabajo a realizar en los próximos meses

La imprevisible pandemia que ha sucedido a principios del 2020, conocida como *COVID-19*, ha obligado a la población a quedarse encerrada en sus viviendas, para así intentar minimizar el riesgo de contagio y el bloqueo del sistema sanitario nacional. Estas medidas sin precedentes han provocado que el proyecto se finalice sin poder efectuar comprobaciones y retoques en el sistema de interacción con las gafas y los controladores de Realidad Virtual físicos; proceso esencial que estaba planificado para realizarse durante la fase final del proyecto, dado que era el momento en el que se configurarían las sesiones e interacciones para demostrar todos los desarrollos realizados hasta ese punto.

Para adaptarnos a este imprevisto ajeno al proyecto, nos hemos visto obligados a posponer diversas tareas hasta el momento en el que finalice el estado de alarma que limita la movilidad y el acceso al material prestado por el centro de investigación VIRVIG. Será esencial que estas tareas se realicen para poder dar por concluidas las modificaciones y ampliaciones al sistema que se establecieron en la planificación del proyecto y validar el producto final.

- Ampliar personalización en la sesión de entrega de material al cirujano, de modo que se permita escoger entre usar el método actual de elección aleatoria de las herramientas o que el usuario establezca a mano el orden en el que el cirujano las irá pidiendo, mediante el uso del sistema de nodos.
- Mientras el usuario que realice la simulación tenga el menú de pausa abierto, se deberán paralizar todas las animaciones e interacciones por el escenario.
- Mejora en el sistema de interacción con los mandos, de modo que las animaciones en las manos sean las correctas y las herramientas se sitúen en la mano de un modo parecido a como se cogerían en la realidad.

- Realizar una refactorización de implementación realizada en la caja metálica y la comprobación de reglas, situando la lógica en una clase de la que hereden las que tienen la implementación actualmente. De este modo conseguiremos que el código sea reusable y se pueda utilizar en otras implementaciones que apliquen sistemas parecidos en objetos contenedores o nodos comprobadores.
- Probar la correcta interacción con todos los elementos incluidos en las últimas implementaciones utilizando los controladores y las gafas de VR.
- Soporte y solución a otros errores no contemplados que se detecten al realizar simulaciones en el sistema con el hardware específico.

Futuras ampliaciones del sistema

Para terminar con la visión de futuro del proyecto, es necesario tener en cuenta todas aquellas tareas que podrían aportar más valor al sistema de simulación, pero que no se han podido incluir en la planificación de esta segunda fase del proyecto.

- Implementar un sistema de sonido para la aplicación, de modo que se pueda poner música ambiente de fondo y se reproduzcan sonidos al interactuar con los objetos y el entorno. Esta tarea también podría incluir la implementación del sistema de síntesis de voz que se estuvo investigando en este proyecto, para que la información de la sesión también pueda ser transmitida oralmente.
- Añadir un nodo que permita la introducción de pequeños fragmentos de código. Esta característica podría dar más opciones de personalización al usuario, permitiendo la modificación de comportamientos simples con la ejecución de scripts interpretados.
- Implementar un sistema de guardado y carga de sesiones desde fuera de la aplicación, de modo que se puedan extraer sesiones para importarlas en diferentes dispositivos, o se permitan guardar en algún almacenamiento en línea.
- Desarrollar una adaptación del sistema de edición de sesiones para que pueda ser utilizado en dispositivos móviles, dado que el sistema actual solo permite realizar modificaciones desde un ordenador.
- Realizar mejoras en el editor, por ejemplo: poder deshacer los últimos cambios realizados (*Ctrl+Z* en editores), permitir cambiar el tamaño de los nodos desde el editor y permitir editar las conexiones de los puertos. También debería realizarse una adaptación de la interfaz para que pueda ser utilizada por usuarios con pocos conocimientos de informática.

Llegados a este punto, es nuestra responsabilidad realizar una reflexión general del proyecto teniendo en cuenta los objetivos iniciales establecidos y valorando su grado de satisfacción. La realización de este proyecto me ha permitido efectuar un aprendizaje intensivo de las distintas partes que componen la gestión de proyectos. Dado el hecho de que era mi primera experiencia estando al mando de todas las decisiones de un proyecto de este calibre, he podido aprender la importancia de una correcta gestión y organización del trabajo pendiente, incluida la gestión de plazos de entrega y los métodos de actuación en caso de tener un bloqueo en algún desarrollo.

El hecho de haber tenido que cargar con la responsabilidad de realizar todas las implementaciones del proyecto por mi cuenta, ha permitido que comprenda la importancia de disponer de varias personas para realizar los desarrollos. Si bien he tenido disponible el soporte tanto de mi director como del compañero que implementó el sistema inicialmente, todas las decisiones finales las he tenido que tomar por mi cuenta, sintiendo constantemente el deber de tomar la mejor resolución posible y echando en falta el contraste de opiniones entre compañeros de equipo.

De todos modos, siento que estas dificultades me han ayudado a ser más autónomo y resolutivo, permitiéndome aprender a gestionar mejor mi tiempo y mis niveles de estrés. Después de mucho esfuerzo y la inversión de muchas horas, he conseguido desarrollar en *Unity* el sistema de realidad virtual requerido, así que me siento muy orgulloso con los resultados obtenidos y haber conseguido realizar el reto que se me había planteado al inicio del proyecto.

Agradecimientos

Por último, me gustaría dejar reflejada en esta parte final de la documentación una reflexión más a nivel personal, remarcando el hecho de que este proyecto significa la finalización de una de las etapas más importantes y complicadas de mi vida, sobre todo a nivel académico.

Me gustaría empezar dando las gracias especialmente a mis padres, por el soporte y los ánimos que me han estado dando durante todo mi recorrido académico; a mi hermano, por ayudarme a desconectar cada semana y aportar un lado más artístico al diseño de la documentación y la presentación; a mi hermana, por enseñarme a ser valiente y no rendirme nunca. A todos mis amigos, que me han ayudado a evadirme de mis preocupaciones durante largos ratos; incluidos todos los que me han acompañado durante estos años en la universidad, especialmente Albert, Isa y Borja.

Finalmente, agradecer todo el apoyo que he recibido por parte de mi tutor del TFG, Carlos Andújar, que ha estado realizando un control semanal de mis avances y me ha ayudado a ver las cosas mucho más claras con cualquier duda que he tenido. También agradecer a Sergi Tortosa todo el soporte técnico que me ha dado para aclarar dudas sobre su sistema; a Esther Insa y Dori Andújar del *Hospital Sant Joan de Déu*, por aportarnos todo el conocimiento médico y permitirnos visitar las instalaciones; a Marta Fairen, por acompañarnos en las reuniones médicas; a los compañeros del VIRVIG, por prestarme el material y acompañarme en más de una comida; a mis compañeros de Everis, por hacerme mejorar a nivel personal y laboral.

Anexo

A. Documentación explicativa de la herramienta

A.1. Descarga y configuración del proyecto

El proyecto se encuentra subido en *GitHub* [35, 36], con el código disponible públicamente. Para obtenerlo se puede realizar un *Clone* del repositorio para modificarlo en local, o bien realizar un *Fork* en la cuenta de *GitHub* personal para poder hacer modificaciones en el proyecto. Destacar que al realizar un *Fork*, después se podrá solicitar al proyecto original que se actualice con los cambios realizados en nuestros *Commits* (lo que es conocido como *Pull Request*). En caso de necesitar más información sobre la utilización de *GitHub*, consultar la información en su página web oficial [37].

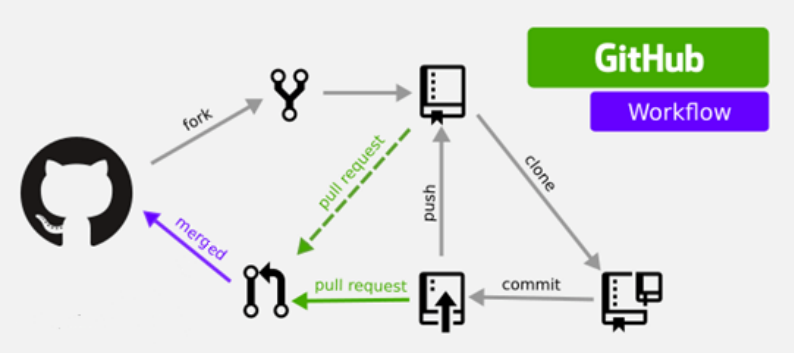


Figura 30: *Ejemplo de GitHub Workflow*

Es necesario aclarar que la versión de *xNode* que se utiliza en el proyecto es una versión concreta con algunas modificaciones en las clases, y está configurado en el repositorio GIT para que se obtenga como submódulo del proyecto. Esta versión se cogió como base en la expansión de funcionalidades que se hizo en la primera aproximación del proyecto, realizada por el compañero *Sergi Tortosa*.

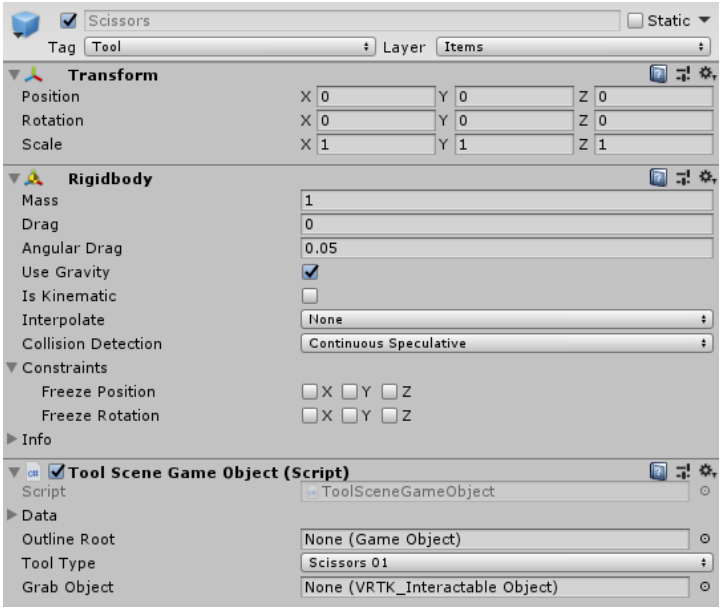
Para cargar los modelos de los objetos utilizados, es necesario importar en el proyecto de *Unity* los *Assets* de pago que se obtuvieron para poder realizar el proyecto. En caso de no disponer de estos Assets, se pueden sustituir los modelos utilizados por otros que tenga disponibles el usuario, o bien obtener los mismos o algunos parecidos en la tienda oficial de *Unity*.

A.2. Añadir nuevas herramientas

En los assets obtenidos para el proyecto, podemos obtener diversas herramientas en la carpeta *TestProps/OR_Room/Medical/Props*.

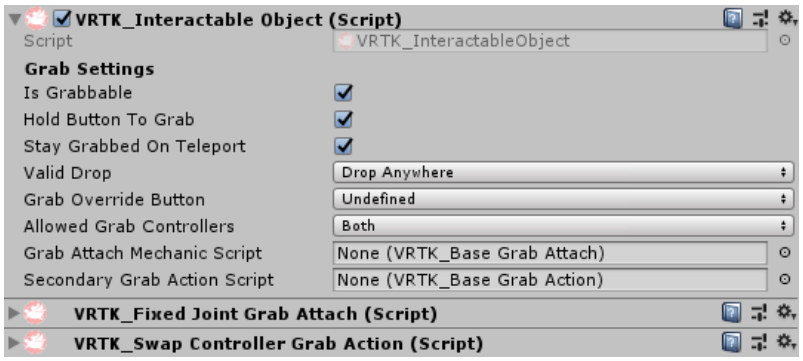
Para añadir una nueva herramienta al sistema, primero deberemos importar el modelo a nuestro proyecto Unity. Una vez realizado esto, deberemos abrir una escena temporalmente para realizar las configuraciones que generen un *Prefab* usable.

1. Crearemos un nuevo *Empty Gameobject* y le añadiremos el componente *Rigidbody* (marcando la opción de *Collision Detection* como “Continuous Speculative”). También añadiremos el script *ToolSceneGameObject*, el cual deberemos editar para añadir el nuevo *Tool Type* con el nombre de la herramienta en el enumerable “Tools” del script, para finalmente poder seleccionar la nueva opción en el Inspector de Unity.



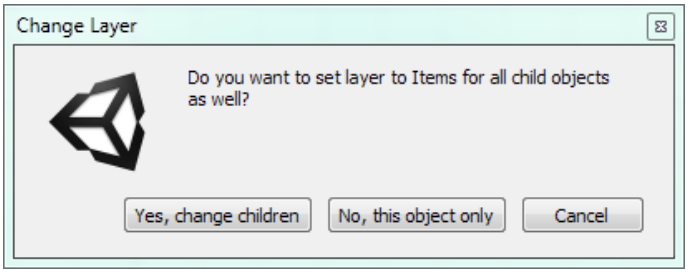

```
public enum Tools
{
    Scissors01,
    Forceps01,
    Clamp01,
}
```

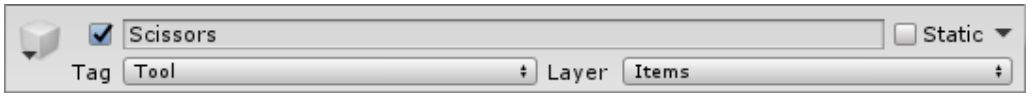
2. A continuación añadir los componentes necesarios para la interacción de VRTK: *VRTK Interactable Object* (marcar “Is Grabbable”), *VRTK Fixed Joint Grab* y *VRTK SwapControllerGrabAction*.



3. Deberemos arrastrar el modelo de la herramienta que queremos añadir al *GameObject* que hemos estado configurando, de modo que se sitúe como hijo. En caso que no disponga de un *Collider*, deberemos añadirlo y configurarlo de modo que se ajuste a su forma (es preferible usar colliders simples como *BoxCollider*).

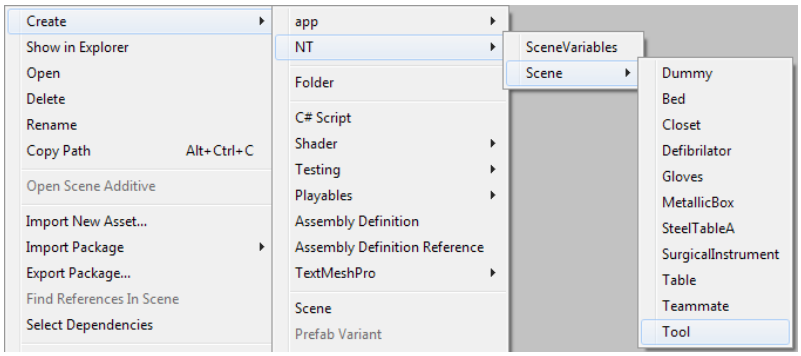
4. Deberemos seleccionar el *GameObject* padre, y en el inspector cambiar su tag a “Tool” y su layer a “Items”. En este último caso confirmamos que tanto a él como a sus hijos se les modifique el layer.



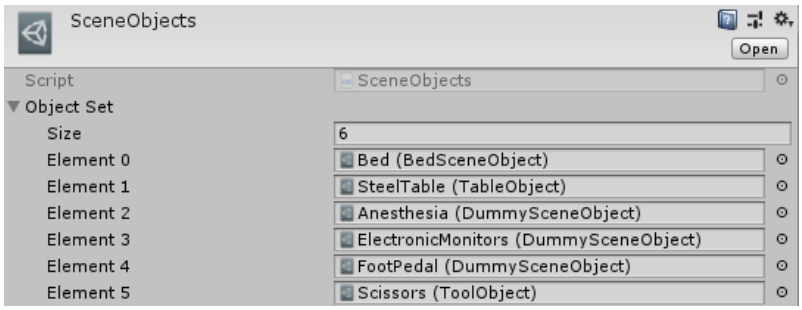


5. Ahora guardaremos la configuración en un *Prefab*, de modo que situaremos nuestro explorador de proyectos en la carpeta *Prefabs/SceneObjects/Tools* y arrastraremos el *GameObject* padre para crear el *Prefab* en el directorio. Deberemos editar el *Prefab* para asegurarnos que tanto el *Gameobject* padre como el hijo tienen la Posición y Rotación del Transform todo a 0, para evitar problemas con las interacciones.

6. En el mismo directorio, crearemos el script de configuración de herramientas con: click derecho ->Create >NT >Scene >Tool. Configurar los parámetros de acuerdo al comportamiento esperado



7. Para que la nueva herramienta esté disponible en el editor, deberemos ir una carpeta atrás (*Prefabs/SceneObjects*) y seleccionar el fichero *SceneObjects*. En el inspector desplegaremos “Object Set”, añadiremos 1 al *size* y seleccionaremos en la última posición el *ToolObject* recién creado.



A.3. Explicación de los scripts principales del proyecto

Scripts principales de gestión (*Managers*):

- **Session Manager:** Se encarga de la gestión de la sesión, incluyendo el guardado de la configuración y su carga al realizar la simulación. Por sus características se determina como Singleton, de modo que solo se puede instanciar una vez, y la instancia se puede encontrar en *SessionManager.Instance*.
- **Exercise Manager:** Se encarga de gestiones durante la simulación del ejercicio, como abrir el menú de pausa y empezar o terminar la sesión.
- **Main Menu Manager:** Gestiona las funcionalidades que se pueden realizar en el menú principal de la aplicación.

Scripts principales de los nodos (xNodeExtension):

- **Runtime Graph:** Es el encargado de tener la referencia a los GameObjects que se utilizarán para instanciar visualmente a los nodos.
- **UGUI Base Node:** Contiene la lógica básica que tendrán todos los nodos como base, incluidas funcionalidades como la instanciación de sus puertos al cargar y la configuración de sus propiedades.
- **UGUI Port:** Contiene la lógica para crear y utilizar puertos en los nodos, además de funcionalidades como el comportamiento al arrastrar y soltar el ratón encima suyo, la coloración dependiendo del tipo de dato o la tipología de puerto (input/output).
- **GUIProperty:** Contiene la lógica para escoger el tipo de propiedad que hay que mostrar en el puerto del nodo (checkbox, campo de texto o enumeración) y realizar su correcta configuración.

A.4. Estructura de las escenas (Menú, Editor, Ejercicio)

Main

Escena inicial, en la que se muestra el menú principal de la aplicación. Se permite crear nuevas sesiones, modificar las actuales y ejecutar el ejercicio. El principal objetivo de esta escena es que se utilice como el nexo de unión de las diferentes partes que forman la aplicación.

EditorScene

Escena del editor de sesión, desde donde se pueden modificar los objetos existentes en la sesión y el comportamiento definido por los nodos. Esta es la escena con la jerarquía más compleja del proyecto, dado que se coordinan todos los elementos que componen el editor junto a sus scripts y configuraciones.

Exercise

Escena desde donde se ejecuta la sesión. Contiene toda la lógica del sistema de Realidad Virtual y los componentes del toolkit que se utiliza (VRTK).

A.5. Otras referencias

Para consultar el funcionamiento del sistema de edición en sí, y sobre todo la parte de utilización y generación de nodos y puertos, se puede consultar la memoria que realizó Sergi Tortosa explicando todo el proceso de implementación del sistema que realizó en su Proyecto Final de Master [9].

Referencias

[1] Alejandro Polanco. Tecnología Obsoleta - Sensorama. https://alpoma.net/tecob/?p=1141. Accedido: 14/04/2019.

[2] Lisando Pardo. NeoTeo - Sega VR. https://www.neoteo.com/sitem/sega-vr/. Accedido: 14/04/2019.

[3] Kent Bye. 50 Years of VR: The Super Cockpit. https://www.roadtovr.com/50-years-vr-tom-furness-super-cockpit-virtual-retinal-display-hit-lab-virtual-world-society/ Accedido: 14/04/2019.

[4] HTC Vive. https://www.vive.com/eu/. Accedido: 14/04/2019.

[5] SteamVR configuration for HTC Vive. https://support.steampowered.com/steamvr/HTC_Vive/. Accedido: 14/04/2019.

[6] Oxford Medical Simulation. https://oxfordmedicalsimulation.com/. Accedido: 14/04/2019.

[7] UbiSim VR. https://www.ubisimvr.com/. Accedido: 14/04/2019.

[8] VR Medical - Arch Virtual. https://archvirtual.com/medical/. Accedido: 14/04/2019.

[9] Sergi Tortosa. A language for designing nursing training sessions. *Facultat d'Informàtica de Barcelona - UPC*, 2019.

[10] Thomas D Parsons and Albert A Rizzo. Affective outcomes of virtual reality exposure therapy for anxiety and specific phobias: A meta-analysis. *Journal of behavior therapy and experimental psychiatry*, 39(3):250–261, 2008.

[11] David Jack, Rares Boian, Alma S Merians, Marilyn Tremaine, Grigore C Burdea, Sergei V Adamovich, Michael Recce, and Howard Poizner. Virtual reality-enhanced stroke rehabilitation. *IEEE transactions on neural systems and rehabilitation engineering*, 9(3):308–318, 2001.

[12] Zahira Merchant, Ernest T Goetz, Lauren Cifuentes, Wendy Keeney-Kennicutt, and Trina J Davis. Effectiveness of virtual reality-based instruction on students'learning outcomes in k-12 and higher education: A meta-analysis. *Computers & Education*, 70:29–40, 2014.

[13] Morton L Heilig. Sensorama simulator, August 28 1962. US Patent 3,050,870.

[14] Huong Q Dinh, Neff Walker, Larry F Hodges, Chang Song, and Akira Kobayashi. Evaluating the importance of multi-sensory input on memory and the sense of presence in virtual environments. In *Proceedings IEEE Virtual Reality (Cat. No. 99CB36316)*, pages 222–228. IEEE, 1999.

[15] David Kahaner. Japanese activities in virtual reality. *IEEE Computer Graphics and Applications*, 14(1):75–78, 1994.

[16] Paul Williams and JS Perry Hobson. Virtual reality and tourism: fact or fantasy? *Tourism Management*, 16(6):423–427, 1995.

[17] Tomasz Mazuryk and Michael Gervautz. Virtual reality-history, applications, technology and future. *Vienna University of Technology*, 1996.

[18] John M Rolfe and Ken J Staples. *Flight simulation*. Cambridge University Press, 1988.

[19] Thomas A Furness III. The super cockpit and its human factors challenges. In *Proceedings of the human factors society annual meeting*, volume 30, pages 48–52. SAGE Publications Sage CA: Los Angeles, CA, 1986.

[20] Robert T Hays, John W Jacobs, Carolyn Prince, and Eduardo Salas. Flight simulator training effectiveness: A meta-analysis. *Military psychology*, 4(2):63–74, 1992.

[21] Peter H Cosman, Patrick C Cregan, Christopher J Martin, and John A Cartmill. Virtual reality simulators: current status in acquisition and assessment of surgical skills. *ANZ journal of surgery*, 72(1):30–34, 2002.

[22] Guiseppe Riva. Applications of virtual environments in medicine. *Methods of information in medicine*, 42(05):524–534, 2003.

[23] Teodor P Grantcharov, Viggo B Kristiansen, Jørgen Bendix, Linda Bardram, Jacob Rosenberg, and Peter Funch-Jensen. Randomized clinical trial of virtual reality simulation for laparoscopic skills training. *British journal of surgery*, 91(2):146–150, 2004.

[24] Randolph H Steadman, Wendy C Coates, Yue Ming Huang, Rima Matevosian, Baxter R Larmon, Lynne McCullough, and Danit Ariel. Simulation-based training is superior to problem-based learning for the acquisition of critical assessment and management skills. *Critical care medicine*, 34(1):151–157, 2006.

[25] Richard M Satava. Virtual reality surgical simulator. *Surgical endoscopy*, 7(3):203–205, 1993.

[26] Oculus. Quest All-In-One VR. `https://www.oculus.com/quest/`. Accedido: 05/10/2019.

[27] Fabrizia Mantovani, Gianluca Castelnuovo, Andrea Gaggioli, and Giuseppe Riva. Virtual reality training for health-care professionals. *CyberPsychology & Behavior*, 6(4):389–395, 2003.

[28] TheStoneFox. VRTK GitHub Repository. `https://github.com/ExtendRealityLtd/VRTK`. Accedido: 23/09/2019.

[29] Robert C Martin. *Agile software development: principles, patterns, and practices*. Prentice Hall, 2002.

[30] UPC. Prácticas en empresa - Facultad de Informática de Barcelona. `https://www.fib.upc.edu/es/empresa/practicas-en-empresa`. Accedido: 01/10/2019.

[31] HAYS. Informe Sectores y Salarios (2018). `https://guiasalarial.hays.es/descargar/sectores-y-salarios`. Accedido: 10/10/2019.

[32] What is a game engine - Unity. `https://unity3d.com/what-is-a-game-engine`. Accedido: 14/04/2019.

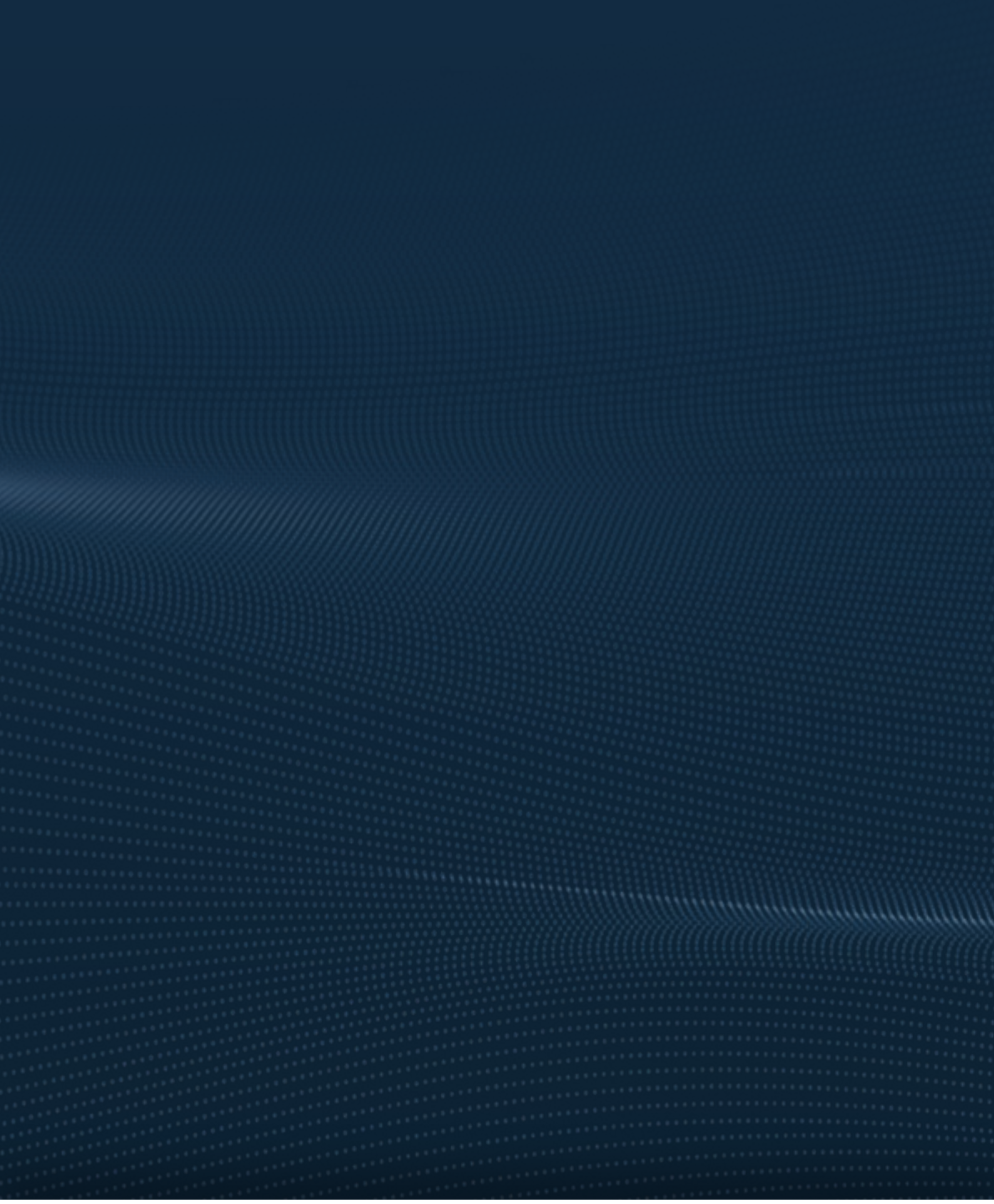
[33] Coding Seb. Expression Evaluator. `https://github.com/codingseb/ExpressionEvaluator`. Accedido: 05/11/2019.

[34] ActiveNick. Unity Text-to-Speech with Microsoft Cognitive Services. `https://github.com/ActiveNick/Unity-Text-to-Speech`. Accedido: 20/11/2019.

[35] Jordi Nieto. VR Entrenamiento Enfermería. `https://github.com/Bermellet/vr-entrenamiento-enfermeria`. Accedido: 14/04/2019.

[36] Sergi Tortosa. Nursing Training. `https://github.com/sigr3s/NursingTraining`. Accedido: 14/04/2019.

[37] GitHub. Ayuda de GitHub. `https://help.github.com/es`. Accedido: 14/04/2019.



TRABAJO FINAL DE GRADO

Sistema VR configurable para entrenamiento en enfermería

UPC