Formación de Instructores de Simuladores de Vuelo basada en Escenarios Virtuales de Entrenamiento

Alfredo Hernández Burgos

FACULTAD DE INFORMÁTICA
DEPARTAMENTO DE INGENIERÍA DEL SOFTWARE
E INTELIGENCIA ARTIFICIAL
UNIVERSIDAD COMPLUTENSE DE MADRID

Trabajo Fin Grado en Ingeniería Informática

Madrid, 26 de junio de 2020

Director: Prof. Dr. Federico Peinado Gil

Agradecimientos

En primer lugar agradecer a Federico Peinado Gil, director de este proyecto por haber confiado en mi y por indicarme el camino a seguir cuando parecía que el proyecto no tenia claro el rumbo a seguir.

Muchas gracias también a Rubén Dapica, gracias a él se planteo este proyecto y era el que nos indicaba los errores en algunas de las partes mas técnicas en cuanto a terminología aeronáutica.

A mis padres por aguantarme y apoyarme, por no dejar que me rindiese cuando las cosas se ponían difíciles tanto en el proyecto como en la carrera en general y por preocuparse cuando tomaba un camino equivocado.

A mi hermana y mi cuñado por decirme las cosas claras cuando hacia falta, por preocuparse y apoyarme como podían y por darme a esas dos fantásticas sobrinas que aliviaban todo el estrés cuando aparecían por casa.

A mis amigos, porque me daban feedback lo mejor que podían cuando les enseñaba los progresos que hacia, aunque en algunos casos no entendían las cosas de las que hablaba.

A los compañeros de Narratech, que veían los progresos que hacíamos en las diferentes reuniones que tuvimos con ellos y nos daban su opinión para poder mejorar en nuestros proyectos.

A todos aquellos que contestaron el cuestionario, que dieron una opinión profesional del proyecto realizado.

GRACIAS

Índice general

Resumen

A pesar de todas las mejoras tecnológicas que existen en el sector aeronáutico, la mayoría de los accidentes se deben a factores humanos. (Kharoufah, Murray, Baxter, y Wild, 2018). Por eso estos últimos años, la Organización Internacional de Aviación Civil [1] (ICAO) viene impulsando un paradigma que integre todo tipo de competencias en el rendimiento de la tripulación, tanto técnicas como no técnicas. Se busca que el entrenamiento en cada sesión sea específico para las necesidades de formación de cada piloto, todo en base a los datos obtenidos tanto en operaciones reales como de entrenamiento. This alternative paradigm is called Evidence-Based Training (EBT) (ICAO, 2013).

Una parte fundamental en los escenarios de entrenamiento es el uso de simulador de vuelo. En estos simuladores al piloto se le somete a situaciones específicamente diseñadas para poner a prueba sus competencias. El diseño de estos escenarios suele corresponder al instructor, que muchas veces disponen de poco tiempo para elaborar nuevo material y reutilizan el viejo, y a pesar de ser un elemento tan crítico en la formación de pilotaje de aeronaves, lo cierto es que este trabajo de los instructores no está regulado ni vigilado convenientemente.

Nuestra propuesta consiste en estudiar la posibilidad de crear un aplicación informática educativa para instructores vuelo, de tal forma que estos se puedan enfrentar a situaciones similares a las que ocurren en el mundo real pero sin riesgos ni costes reales, de manera que puedan mejorar sus habilidades en un entorno controlado. Esta aplicación estaría enfocada al campo de la aviación, pudiéndose utilizar en docencia o formación del sector aeroespacial para mejorar los conocimientos o habilidades de los instructores o candidato a instructores de pilotos.

[1] `https://www.icao.int`

Como resultado de este estudio proponemos el diseño de este "simulador de instructores de vuelo", de manera que los usuarios se puedan poner en el papel de un instructor y puedan seleccionar a un piloto ficticio para entrenarlo en las competencias que necesite mejorar este, utilizando para ello un escenario de entrenamiento que también podrá seleccionar e incluso configurar. Posteriormente dicho usuario deberá evaluar la actuación que ha tenido el piloto en el escenario propuesto y acabara recibiendo un informe sobre su labor como instructor y evaluador de pilotos.

Para llevar a cabo este proyecto se investigaron diferentes simuladores de vuelo, para conocer que experiencia ofrecían y que información mostraban, pues aunque no se trataba de desarrollar un simulador de vuelo el sistema propuesto era similar y se fundamentaba sobre los mismos conocimientos (eventos que pueden ocurrir durante el vuelo, competencias de los pilotos que deben entrenarse y son susceptibles de evaluación, etc.). Lo primero que se hizo a continuación fue hacer bocetos y esquemas en papel, para inmediatamente después trabajar con Balsamiq en las primeras maquetas cuya facilidad de uso pudiese probarse. Una vez que los conceptos estaban claros y no había grandes incógnitas, el siguiente paso fue empezar a trabajar directamente en un prototipo funcional en Unreal Engine donde se añadieron características que inicialmente no estaban planteadas. Este proceso iterativo de mejora continuó hasta llegar al diseño final que hemos querido poner a prueba para verificar la utilidad de la aplicación y considerar posibles modificaciones, puliendo detalles según el criterio de evaluadores de la herramienta expertos en el sector aeroespacial.

Palabras clave

Simulador de vuelo, Aeronáutica, Aviación, Aprendizaje, Escenarios de entrenamiento, Instructores, Competencias, Evaluación de Pilotos

Capítulo 1

Introducción

1.1. Estructura de la memoria

Los capítulos que de los que esta compuesto este trabajo son los siguientes:

1. En el capítulo 1: Introducción, se relata el planteamiento inicial del proyecto, las razones de elegir un proyecto que se apoya en el ámbito aeroespacial, las razones de decidir hacer un simulador y como el enfoque inicial del simulador se acabo modificando dando lugar un tipo diferente de simulador.

2. En el capítulo 2: Estado de la técnica, se hablará de la historia de la aviación para situar el campo en el vamos a movernos durante todo el proyecto, acabando con la historia de los simuladores de vuelo para especificar aun más en el ámbito. Después se comentaran algunos de los tipos de software destinados a educación pues al final el proyecto es un simulador que no deja de ser un tipo de software destinado a la educación. Y se hablara sobre algunos de los simuladores existentes, así como de el conocimiento aeroespacial sobre entrenamientos en los que se basa el proyecto.

3. En el capítulo 3: Objetivos y especificación, en este capítulo se mencionarán los objetivos iniciales del proyecto y su evolución durante el desarrollo. Para acabar se definirán y explicarán cuales fueron finalmente los objetivos de este proyecto. Por otro lado se mencionarán y explicarán los requisitos del software de esta aplicación, tanto los funcionales como los no funcionales.

4. En el capítulo 4: Metodología y herramientas, se explicarán las razones que dieron lugar a la elección de la metodología utilizada para este proyecto, también se mencionaran las herramientas utilizadas para el desarrollo del proyecto y como se utilizaron en el mismo.

5. En el capítulo 5: Análisis, diseño e implementación, recorreremos todas las clases, estructuras y menús creados, explicando su función, las variables o campos que contienen y las funciones creadas en su interior explicando para que se utilizan.

6. En el capítulo 6: Pruebas y resultados, se explicarán los motivos de realizar una prueba de concepto y veremos y analizaremos las respuestas de los especialistas al formulario que rellenaron después de ver el vídeo de la prueba de concepto.

7. En el capítulo 7: Conclusiones, después de analizar las respuestas obtenidas, en este capitulo veremos las conclusiones a las que llegamos y se explicará el posible futuro de este proyecto, viendo como podría desarrollarse.

1.2. Planteamiento inicial

Para el planteamiento este proyecto, se pensó que en muchos casos estos proyectos están enfocados únicamente en el campo de la informática, cada uno enfocado en un área o campo concretos dentro de esta, pero solo teniendo en cuenta la informática.

Por eso para realizar este proyecto se pensó que podría ser un cambio recomendable apoyarse en otro campo diferente del de la informática, de forma que utilizando los conocimiento adquiridos sobre informática a lo largo de la carrera se podría ayudar en otro campo de conocimiento diferente.

Pensando que sería recomendable seleccionar un campo que ya tuviese alguna relación con la informática, pues intentar unir dos campos que no tiene nada que ver al menos hasta el momento puede ser una tarea difícil de afrontar y a la que hay que dedicarle un tiempo del que no se disponía.

Por eso después de pensar en que otra disciplina de conocimiento se podría apoyar este proyecto, se decidió que sería buena idea utilizar la aeronáutica.

La informática y la aeronáutica ya están bastante relacionadas, tanto por la existencia de los sistemas montados dentro de las diferentes aeronaves, como por el desarrollo de los simuladores de vuelo, y fue precisamente en los simuladores en los que se enfocaría finalmente este proyecto.

Una vez decidido esto, se empezó a pensar que tipo de simulador se desarrollaría pues ya hay muchos simuladores en el mercado, y tomando como referencia el artículo *Evaluation of a Similarity Function for Generation of Flight Simulator Training Scenarios using Case-Based Reasoning* (Dapica y Peinado, s.f.), pensamos que era buena idea plantear un simulador de vuelo basado en SBT [1].

Aun teniendo claro este enfoque, al ir avanzando en el proyecto, este se fue modificando y evolucionando. Uno de los mayores cambios que sufrió fue el cambio de enfoque de ser un simulador de vuelo a ser un simulador para instructores de vuelo.

Las razones de esto fueron, primero que debido a la cantidad de simuladores existentes era complicado hacer algo novedoso o que profundizase en algún aspecto concreto de los simuladores, mejorando así los simuladores de vuelo existentes actualmente.

[1] Scenario-based training

Por otro lado no se encontraron simuladores pensados para instructores de vuelo por lo tanto era una faceta de los simuladores que se podría explotar sin tener competencia.

Y otra de las razones fue que sería una buena idea poder ayudar a los instructores de vuelo a mejorar sus capacidades a la hora de instruir y evaluar pilotos, porque aparte de poder reciclar sus conocimientos mediante cursos y certificaciones, un instructor de vuelo solo puede utilizar simuladores de vuelo, pero de esta forma solo puede entrenar sus capacidades de vuelo como piloto, pero no las capacidades didácticas para impartir o evaluar el conocimiento de los pilotos.

De cara a la evaluación que haría el instructor al piloto, actualmente se está empezando a instauran un modelo tanto de aprendizaje (Ortega, 2017) como de gestión (Sánchez Ramírez, 2014) basado en competencias, tanto en el ámbito aeroespacial como en otras áreas por lo que se pensó que sería bueno tomar esta idea para el proyecto, de tal forma que el instructor diseñase los escenarios en función de las competencias que el piloto necesitaría entrenar.

Una vez que se planteo que sería un simulador para instructores de vuelo, y que tomaría el enfoque de la competencias para evaluar al piloto, no sufrió cambios muy importantes, pero se añadieron algunas funcionalidades y se fue modificando el diseño de los diferentes menús.

Aunque se modificaron algunas características del proyecto, la base siempre fue el entrenamiento basado en escenarios en el finalmente sería el instructor el que modificaría los diferentes eventos que forman el escenario, para adecuarlo a las necesidades del piloto. De esta forma en función de las competencias que tenga el piloto y que necesite entrenar, el instructor debería elegir los eventos en consecuencia para formar un escenario adecuado.

1.3. Asignaturas

Las asignaturas que resultaron mas útiles para realizar este proyecto en concreto fueron Fundamentos de la programación, Estructuras de datos e información, Desarrollo de sistemas interactivos, el curso de Unreal Engine y Practicas en empresas, a continuación se indicará en que aspectos fueron relevantes dichas asignaturas.

- **Fundamentos de la programación**: A pesar de utilizar los *blueprint* y nodos proporcionados por Unreal Engine, la forma y las reglas a la hora de programar son las mismas que si programamos con código tal y como se nos enseña en las diferentes asignaturas de programación y concretamente en fundamentos de la programación.

- **Estructura de datos e información**: Esta asignatura ha sido especialmente útil dada la cantidad de estructuras que hay que crear para poder representar la información de pilotos, eventos, escenarios ..., así como identificar que clases eran mas útiles para representar esta información. Otro aspecto que ha sido muy útil de esta asignatura es la capacidad de ver el consumo de espacio de según que estructuras y formas de trasmitir esta información de formas mas eficientes.

- **Desarrollo de sistemas interactivos**: Esta asignatura ha sido útil sobre todo porque gracias a ella se tenían conocimiento sobre las diferentes metodologías, como hacer los prototipos, sacar cuales eran los requisitos..., en definitiva como afrontar el proyecto en si.

- **Practicas en empresas**: En mi caso concreto estuve trabajando con Unreal Engine y uno de los trabajos en los que pase mas tiempo trabajando es en la creación de interfaces para videojuegos, por lo que aprendí el manejo de los *widget* en Unreal Engine.

- **Curso Unreal Engine**: Aunque no es una asignatura, fue conocimiento impartido en la facultad y que permitió tener una base de conocimiento para poder utilizar este motor gráfico.

También aunque no acabo siendo relevante, fueron útiles la asignatura de Bases de datos, Redes y Ampliación de sistemas operativos y redes en el momento en que se planteo la creación de una base de datos que había que conectar con Unreal Engine, pero finalmente se abandono esta idea dado que no parecía correcto que se necesitase conectarte a la red para poder acceder a los datos que se requerían, por lo que finalmente se tomó otro enfoque.

Capítulo 2

Estado de la técnica

Para poder hablar apropiadamente del estado de la técnica en herramientas de entrenamiento de pilotaje de aeronaves, y poder determinar así los productos más relacionados con el tipo de simulador que queremos desarrollar en este proyecto,, hay que empezar conocimiento primero la historia general de la aviación y también con un poco más de detalle la historia de los simuladores de vuelo.

2.1. Historia

Desde 1890, cuando Clément Ader construye su avión autopropulsado, hasta nuestros días, en los que es común tomar un avión para viajar a un país que está al otro lado del océano[1], la historia de la aviación ha progresado a pasos agigantados.

Si profundizamos un poco en la historia, aunque contemos la creación de Ader como el primer avión de la historia, antes de él ya existían los planeadores (no reconocidos, eso sí, como aviones propulsados), que podríamos considerar como precursores del avión y que hacen su primera aparición en los modelos de Leonardo Da Vinci.

[1]Al menos así era antes del COVID-19 y esperemos que lo siga siendo cuando el lector tenga acceso a estas paginas.

Después de Ader también encontramos hitos importantes y quizás mas conocidos en el mundo de la aviación como es el vuelo de los hermanos Wright en 1903, o el considerado como primer vuelo en un avión autopropulsado, realizado por Alberto Santos Dumont en 1906.

La aeronáutica, debido a su importancia tanto por el poder militar que supone como por la información que permitía obtener sobre las fuerzas enemigas, se desarrolló mucho durante los conflictos bélicos, llegando a evolucionar en este campo hasta los modernos aviones de combate no tripulados con los que contamos hoy día.

Pero si nos distanciamos un poco de los aspectos más belicistas de la aviación, los avances producidos durante aquellos conflictos han propiciado que hoy tengamos aviones de uso civil que son comúnmente utilizados por la población, y que son considerados como uno de los medios de transporte más cómodos y seguros que existe.

Si hablamos de uso civil el transporte de viajeros no es su única función pues también son muy importantes a la hora de distribuir mercancías, permitiendo la importación y exportación que de otra forma sería imposible o se tendría que hacer por medio de medios no tan rápidos ni eficaces.

Al hablar de la historia de la aviación no podemos olvidarnos de los helicópteros, que comenzaron su andadura formalmente con los trabajos de Heinrich Focke, un ingeniero alemán que voló en 1936 con el que se considera el primer helicóptero. Antes de Heinrich Focke ya habían aparecido "helicópteros", en forma de prototipo al menos, como el de Berliner o el famoso autogiro del español Juan de la Cierva, aunque muchos otros también experimentaron con aparatos que podrían considerarse en cierta medida precursores del helicóptero actual, llegando incluso hasta el ya mencionado Da Vinci, quien diseño un artefacto volador que contaba con un rotor. Los helicópteros, aunque no se empleen para largos viajes, tienen otras ventajas como por ejemplo su mayor movilidad y maniobrabilidad, muy superior a la de un avión convencional, razón por la cual son muy utilizados por servicios de rescate, fuerzas policiales, servicios médicos, etc. al ser el vehículo aéreo ideal para acceder a ciertos edificios y a terrenos menos accesibles.

Afortunadamente, el uso de la informática y electrónica con la que cuentan los aviones en la mayoría de los casos resulta de mucha ayuda, facilitando a los pilotos información del avión con precisión y eficacia. Otro aspecto que ha mejorado sustancialmente es la comunicación con el exterior, permitiendo a los controladores aéreos saber la posición y los problemas ocurridos en todo momento durante el vuelo.

Estos aspectos junto con otros muchos hacen que nos demos cuenta de cómo la informática y la electrónica se ha acabado convirtiendo en un gran aliado para los pilotos.

De todas formas, existen otros modos en los que la informática ayuda al mundo de la aviación, como cuando la tecnología no está integrada en la aeronave si no que es una forma de ayudar a los pilotos cuando están en tierra, este es el caso particular de la tecnología utilizada en los simuladores de vuelo. Estos son capaces de replicar situaciones y comportamientos que pueden aparecer en la aeronave durante un vuelo, ayudando así en el aprendizaje del piloto a reaccionar ante situaciones que puedan ocurrir durante el vuelo, pero siempre de una forma controlada y en la que no existe peligro.

Aunque actualmente los simuladores de vuelo están muy avanzados tecnológicamente hablando, no siempre fue así y para poder demostrarlo nos tenemos que remontar a 1910 cuando decidieron montar un barril en un marco o cuando montaron el *Sander Teacher* en un brazo articulado de tal forma que el piloto pudiese simular los movimientos del avión sin necesidad de volar realmente.

Pero fue en 1940 cuando aparecieron los primeros simuladores electrónicos, que utilizaban computadoras para realizar las ecuaciones de vuelo. Y fue finalmente en 1948 cuando *Curtiss-Wright* desarrollo el primer simulador de vuelo que fue utilizado por una aerolínea. A partir de aquí se fueron añadiendo diferente elementos para mejorar la experiencia hasta que en 1954 *General Precision Inc.* desarrolla un simulador con movimiento.

A partir de aquí se siguió desarrollando, añadiendo funcionalidad, mejorando sus gráficos, y utilizando mejores computadoras, y todo esto hasta llegar a nuestros días, donde un simulador ya no es meramente algo con el objetivo de enseñar a los pilotos, si no que se ha terminado convirtiéndose también en el entretenimiento de muchas personas que pueden acceder a la sensación de volar desde la comodidad de su hogar.

La importancia de estos simuladores hoy en día es tal que contamos con una gran variedad de ellos, desde simuladores profesionales homologados por los organismos responsables [2], los cuales permiten contabilizar las horas de vuelo simulado como horas de vuelo real para el piloto, hasta simuladores a los que puede acceder cualquier persona e instalarlo en el ordenador domestico para uso recreativo.

Respecto a las empresas que dedican su tiempo al desarrollo de simuladores de vuelo, podemos encontrar varios perfiles que las definen.

[2]como la FAA, Administración Federal de Aviación de los Estados Unidos de América, o las Direcciones Aeronáuticas Civiles de los diferentes países.

Primero tenemos que ver que hay distintos tipos de simuladores de vuelo como por ejemplo los simuladores profesionales, estos simuladores están homologados y son muy fieles a la realidad llegando al punto de recrear físicamente las cabinas en algunos casos de manera que también pueden reproducir parcialmente los movimientos ocurridos en el simulador. Respecto a este tipo de simuladores podemos ver que muchos de ellos están hechos por las mismas empresas que se dedican al diseño y fabricación de aeronaves, como Boeing [3] o Airbus [4], junto con empresas tecnológicas como Indra [5] o escuelas de vuelo como CAE [6] y Panamedia [7]. Posiblemente una de las razones de que esto ocurra es que es más fácil simular un entorno y unas condiciones de las que tienes completo conocimiento, dado que eres el fabricante de la propia aeronave y tienes los conocimientos requeridos para ser piloto.

Por otro lado, también hay empresas que se dedican en exclusiva al desarrollo de simuladores de vuelo, como ejemplo podemos encontrar a Simloc [8], una empresa española que entre sus proyectos actuales cuenta con la integración de un simulador de vuelo en una cabina real.

Pero también podemos encontrar un tipo de simuladores de vuelo destinado a actividades mas recreativas, las empresas encargadas de este tipo de simuladores son generalmente desarrolladoras de videojuegos, como *Microsoft* o *Laminar Research*. Por esto podemos suponer que sus simuladores están destinados a entretener mas que a entrenar una serie de competencias propias del piloto. Nadie está capacitado para pilotar sin obtener las certificaciones correspondientes, aunque dado el realismo conseguido estos sistemas sí que permiten obtener conocimientos sobre aviación.

[3] `https://www.boeing.es/`
[4] `https://www.airbus.com/`
[5] `https://www.indracompany.com/es/trafico-aereo-0`
[6] `https://www.cae.com/civil-aviation/locations/cae-madrid/`
[7] `https://panamedia.org/`
[8] `https://simloc.aero/simuladores-de-vuelo/airbus-a320-easa-faa`

Respecto a los simuladores de uso lúdico no ahondaremos mucho en tema pues normalmente el usuario es un jugador que le gusta poder tener la sensación de pilotar estando en la comodidad de su casa. Las motivaciones podrían ser diversas pero lo que tenemos claro es que generalmente no representa un uso profesional.

Pero respecto a los simuladores profesionales, sí que podemos ver varios tipos de usuarios que hacen uso de los simuladores directa o indirectamente. Los usuarios que utilizan estos simuladores de forma indirecta son aquellos pertenecientes a las compañías aéreas. Estas compañías requieren del uso de simuladores para entrenar o poner a prueba las capacidades de sus pilotos, de tal forma que una compañía podría mandar a un piloto a que entrene con ciertos escenarios específicos en un simulador. Los usuarios directos serían los instructores de vuelo y los propios pilotos.

Los pilotos acuden a las instalaciones donde pueden usar estos simuladores para mejorar sus competencias y para ponerse a prueba. Pueden ir por su cuenta, pero en muchos casos como ya comentábamos, son las compañías aéreas y las regulaciones legales las que los obligan a ir para comprobar que sus capacidades no han mermado, o en caso de reincorporarse, para asegurarse de que las capacidades del piloto siguen intactas como para volver a pilotar.

Los otros usuarios que interactúan de forma directa y en los que ponemos el foco de este proyecto son los instructores de vuelo. Los instructores configuran los escenarios recomendados por las compañías para entrenar a sus pilotos y evalúan el rendimiento de estos durante toda la sesión de simulación. Los instructores tratan de ayudar a los pilotos a mejorar sus competencias y mandan los resultados de su evaluación a las compañías.

2.2. Software destinado a la educación

Como lo que pretendemos desarrollar es un simulador destinado a los instructores de vuelo de forma que estos mejoren a la hora de realizar sus funciones enseñando y evaluando a los pilotos, es conveniente hablar de los diferentes tipos de software que hay destinados a la enseñanza, y que nos ayudan a crecer en diferentes áreas.

Algunos de los tipos que podemos encontrar y que podrían ser mas relevantes en el ámbito en el que estamos trabajando son los siguientes.

2.2.1. Programas de simulación

El mas relacionado con este proyecto, son programas que nos hacen ponernos en la piel del objeto de la simulación, en nuestro caso un instructor de vuelo.

Es una forma de realizar las actividades a las que nos enfrentaremos en un futuro en el mundo real, pero en un entorno seguro en el que no hay repercusiones ni para nosotros ni para terceras personas.

Es un tipo de programas que suele ser efectivo porque simula escenarios fieles a lo que nos vamos a enfrentar, de forma que aprendemos mediante una experiencia realista que puede ser evaluada por el propio simulador o por terceros.

2.2.2. Tutoriales

Es otro tipo de programas en los que generalmente se nos aporta la información de como tenemos que realizar una cierta actividad, explicando detalladamente los pasos a seguir.

En muchos casos en el tutorial se nos pedirá que después de la explicación, realicemos lo que nos acaban de explicar.

Es un método muy eficaz sobre todo en las primeras fases del aprendizaje, porque mezcla un método didáctico con una practica guiada, mostrándonos así algo que tendremos que realizar en un futuro pero sin ningún tipo de guía

2.2.3. Programas de resolución de problemas

Son programas en los que nos plantean un problemas y tenemos que ser capaces de llegar a una solución. En nuestro caso por ejemplo, se nos podría plantear que tenemos que instruir a un piloto para que mejore en una competencia concreta y nosotros como instructor tendríamos que ver que explicarle o que escenario seleccionar o crear, para que de esta forma mejore en esa competencia

2.3. Simuladores

Como se ha explicado anteriormente los simuladores son un tipo de software destinado a la educación y a continuación presentaremos algunos de los simuladores de vuelo que existen, tanto profesionales como no profesionales, explicando algunas de sus características, para poder identificar así algunas de las similitudes que comparten con nuestro simulador.

2.3.1. Microsoft Flight Simulator

cc 

Figura 2.1: Imágenes del nuevo Flight Simulator

Esta es una larga saga de simuladores de vuelo [9], la primera aparición como tal es en 1982 cuando se licencia una versión para PC de IBM a *Microsoft* dando lugar a *Microsoft Flight Simulator 1.0*, con el paso de los años se han desarrollado varios simuladores de esta misma marca, apareciendo en algunos casos versiones estándar y profesionales cuya diferencia era la cantidad de contenido incluido, este desarrollo continuo de simuladores llega hasta la mas actual aparición del *Microsoft Flight Simulator 2020*.

Este es un simulador de vuelo de uso recreativo, no pretende impartir conocimientos ni capacitar realmente a alguien para convertirse en piloto, su única función es proporcionar entretenimiento al usuario. A pesar de ser el entretenimiento su enfoque principal y no estar enfocado a impartir conocimientos realmente,este juego si ha sido utilizado en la educación aeroespacial (JUMPER y BAUGHN, 2012) para enseñar por ejemplo factores humanos en la aviación (F., Moroney, y Moroney, 1991). Cualquier persona puede acceder a el pagando su precio y jugando en su ordenador.

Tiene un modo de entrenamiento, pero que como hemos comentado anteriormente no pretende capacitar a un piloto, es meramente un tutorial para que el jugador aprenda las nociones básicas para poder controlar la aeronave.

También podemos encontrar un modo de juego en el que se nos plantearan retos que tendremos que superar. Son misiones concretas en las que tendremos que ir de un punto A a un punto B con unas determinadas condiciones meteorológicas.

Y tendremos también un modo libre en el que configuraremos nosotros ciertas condiciones meteorológicas y podremos volar libremente sin tener que cumplir ningún tipo de misión o propósito, simplemente disfrutar del vuelo y las vistas.

[9] `https://www.flightsimulator.com/`

2.3.2. F/A-18 Korea Gold

Este simulador de vuelo también es de uso recreativo. En este caso nos pone en la piel de un piloto de las fuerzas aéreas y nos propondrá misiones desde realizar un despegue o aterrizaje en un portaaviones hasta neutralizar objetivos utilizando las armas de que dispone la aeronave.

Aunque se pueda cuestionar el realismo debido a los gráficos, planteaba la realización de varias acciones en las maniobras que lo dotaban de un realismo del que otros juegos de este estilo carecían, algunas de estas acciones a las que nos referimos es la solicitud de una pista de despegue o aterrizaje a la torre de control o cierta comunicación predefinida con torre de control u otras aeronaves.

2.3.3. X-Plane

Esta es otra saga de simuladores de vuelo [10], esta concretamente es uno de los competidores directos de *Microsoft Flight Simulator*. Hizo su primera aparición en 1993 y se han desarrollado varias versiones hasta la mas actual lanzada en 2019.

Aunque es un simulador de vuelo de uso recreativo, la Administración Federal de Aviación de los Estados Unidos de América ha autorizado su uso profesional para el entrenamiento de pilotos, siempre y cuando se use con un hardware específico.

El propósito de este simulador es intentar conseguir es la experiencia más realista posible, recreando el planeta tierra para poder volar libremente en el en una amplia gama de aeronaves. Otra de las opciones que permiten observar este realismo es la posibilidad de conectarnos ha internet a redes que permiten un servicio de control de trafico aéreo.

También cuenta con un editor que permite desde la edición de escenarios para volar hasta la edición y creación de aeronaves, además de perfiles alares.

[10] `https://www.x-plane.com/`

2.3.4. Simuladores profesionales

Figura 2.2: Simulador Profesional Airbus A320

Sin hablar de uno especifico porque en muchos casos la única diferencia es la aeronave que pretenden simular, si que comentaremos las diferentes características que tienen.

Estos simuladores profesionales van desde simuladores simples para entrenamiento básico de un piloto, hasta simuladores que son capaces de recrear lo movimientos simulados para preparar a la tripulación y capacitarlos para las diferentes circunstancias que se pueden dar durante cualquiera de las diferentes fases de vuelo, pero en un entorno seguro que no pone en riesgo la seguridad de los usuarios.

Estos simuladores son capaces de recrear averías en el avión, circunstancias medioambientales o problemas en los aeropuertos como por ejemplo el estado de las pistas, estos serán eventos que se le plantearan a los pilotos y que deberán ser capaces de solventar de forma satisfactoria.

Pero para ser capaces de capacitar a un piloto real y tener algún tipo de validez, estos simuladores tienen que estar homologados y certificados por ciertas instituciones gubernamentales como por ejemplo la Administración Federal de Aviación de los Estados Unidos de América o las Direcciones de Aeronáutica Civil de los diferentes países. Estos organismos los evaluaran y certificaran según su categoría.

2.3.5. Otros

Aunque otro tipo de simuladores, como los deportivos, se apartan bastante de la temática general del proyecto, es necesario hacer una mención especial sobre alguno, como por ejemplo Football Manager (SportsInteractive, 2019).

Las razones de hablar sobre este juego es el enfoque que tiene, en el que en vez de ponernos en la piel del jugador nos ponemos en la piel del mánager del equipo y tendremos que gestionar a un equipo de fútbol. En nuestro caso nos pondremos en la piel del instructor de vuelo que al igual que el mánager, tendrá que gestionar y crear los escenarios en los que entrenará el piloto.

Por este enfoque diferente y dado que no hemos encontrado simuladores que nos pongan en la piel de un instructor de vuelo, era necesario mencionar este videojuego.

Otro tipo de simuladores diferentes de los deportivos pero que también nos aleja del que pueda parecer el actor principal, son los simuladores de gestión. Este tipo de juegos se pusieron muy de moda hace algunos años y en ellos nos ponemos en la piel de por ejemplo el jefe de una fabrica que tiene que gestionar los diferentes pedidos y la contratación como en Little Big Workshop (MirageGameStudios, 2019), o juegos en los que no solo tenemos que gestionar si no construir por ejemplo un zoo (Planet Zoo) (FrontierDevelopments, 2019) o un parque de atracciones (Planet Coaster) (FrontierDevelopments, 2016).

2.4. Conceptos relevantes

Para poder realizar correctamente este proyecto hay que conocer algunos conceptos y metodologías en las que se apoya, y a continuación se explicaran los mas importantes.

2.4.1. *Scenario-based training* (SBT)

El entrenamiento basado en escenarios es un tipo de metodología que busca la simulación de escenarios realistas como medio de capacitación para el usuario. De forma que el usuario sea capaz de practicar y entrenar los comportamiento y capacidades para las que este destinada la simulación.

Aunque el enfoque actual de este proyecto es el ámbito aeroespacial, el SBT no es exclusivo de este ámbito y se usa en otros campos donde es importante mantener un alto grado de habilidades tanto cognitivas como motoras, y en los que las acciones puedan repercutir en la salud o seguridad de terceras personas, los campos mas relevantes en los que se hace uso del SBT son el médico (Salas y cols., 2008) y el militar(Salas, Priest, Wilson, y Burke, 2006).

2.4.2. Evidence Based Training (EBT)

Entrenamiento basado en evidencias, esta es una metodología que consiste en la evaluación de las competencias claves de un piloto, siendo capaz de desarrollarlas para preparar al piloto para situaciones peligrosas durante el vuelo.

Esta metodología empieza a desarrollarse bajo la iniciativa de Capacitación y Calificación IATA (ITQI) lanzada en 2007, y es finalmente en 2013 cuando la OACI respalda la metodología EBT momento en que se publica la "Guía de implementación de capacitación basada en evidencia"(IATA, 2013).

Y es en esta guía donde después de evaluar las situaciones peligrosas y estudiar las respuestas de los pilotos, se identifican entre otras cosas cuales son las competencias más importantes que necesita un piloto para responder en caso de suceder alguna de estas situaciones.

Las competencias son las siguientes:

- **Application of Procedures (APK)**:Identifica y aplica procedimientos de acuerdo con las instrucciones operativas publicadas y las regulaciones aplicables utilizando el conocimiento apropiado.
- **Communication (COM)**:Demuestra comunicaciones efectivas orales, no verbales y escritas en situaciones normales y no normales.
- **Aircraft Flight Path Management, automation (FPA)**:Controla la ruta de vuelo de la aeronave a través de la automatización incluido el uso apropiado de los sistemas de gestión de vuelo y la orientación.
- **Aircraft Flight Path Management, manual control(FPM)**:Controla la ruta de vuelo de la aeronave a través del vuelo manual, incluido el uso apropiado de los sistemas de gestión de vuelo y los sistemas de guía de vuelo.
- **Leadership and Teamwork (LTW)**: Demuestra liderazgo efectivo y trabajo en equipo.
- **Problem Solving and Decision Making (PSD)**: Identifica con precisión los riesgos y resuelve los problemas. Utiliza los procesos de toma de decisiones adecuados.
- **Situation Awareness (SAW)**: Percibe y comprende toda la información relevante disponible y anticipa lo que podría suceder que pueda afectar la operación.
- **Workload Management (WLM)**: Administra los recursos disponibles de manera eficiente para priorizar y realizar tareas de manera oportuna en todas las circunstancias

Estas son las competencias identificadas por IATA como relevantes a la hora de entrenar a un piloto utilizando una metodología EBT.

Capítulo 3

Objetivos y especificación

Inicialmente el objetivo de este proyecto fue crear un simulador de vuelo que se basase en SBT, [1] y este hecho no se modifico durante la realización del proyecto pues seguiríamos basándonos en SBT, pero no estaría destinado al piloto, sería el instructor el que tendría que crear dichos escenarios y sería evaluado por ello, por lo tanto en vez de un simulador de vuelo sería mas correcto llamarlo simulador de instructores de vuelo.

Conforme se avanzo en el desarrollo del proyecto y se empezaron a tener mas conocimientos sobre simuladores de vuelo profesionales, se fueron modificando los objetivos poco a poco, se añadieron algunas funcionalidades y otras fueron modificadas. Para empezar, el aula virtual se convirtió en un elemento secundario, pues ya existen en la industria soluciones similares mucho más avanzadas como por ejemplo recreaciones en realidad virtual de las cabinas de los aviones. Por otro lado el objetivo principal aunque seguía basándose en la creación de escenarios, se centro mas en la elección de los diferentes eventos que formaban el escenario. Esta representación del escenario que consiste en una serie de eventos durante la simulación y un contexto operativo se denomina EBAT [2].

[1] Scenario-based training

[2] event-based approach to training

Después de esto se podía definir el propósito de el sistema propuesto, que ya no era tanto un simulador de vuelo sino mas bien un simulador de instructores de vuelo. Al poner el foco en los instructores, pero manteniendo la idea de la formación y la representación de eventos en vuelo para configurar escenarios de entrenamiento, etc., lo que se pretendía era que el usuario "jugase" a entrenar pilotos ficticios mediante la creación o modificación de escenarios que contendrían estos eventos y que estarían además diseñados para entrenar competencias específicas de pilotaje de aeronaves. Como el que crea los escenarios es el instructor, este es también el que puede ser evaluado en función de sus elecciones, según sea o no bueno formado de pilotos, lo que deberá demostrar con la creación de escenarios y con la propia evaluación que haga de los pilotos en formación.

3.1. Objetivos

Los objetivos principales de desarrollo del sistema informático propuesto son los siguientes:

- Crear las clases necesarias para guardar la información imprescindible que necesitamos (Pilotos, Escenarios, Eventos, Meteorología ...).
- Diseñar una interfaz que sea lo mas clara o intuitiva posible, de forma que el instructor no se distraiga con los menús y se pueda centrar en las partes más importantes como la creación de los escenarios.
- Desarrollar un subsistema por el que el usuario pueda seleccionar y modificar escenarios para poder entrenar competencias específicas de los pilotos ficticios.
- Desarrollar otro subsistema mediante el cual un instructor pueda evaluar el desempeño de un piloto ficticio durante la simulación de un escenario escogido.

- Diseñar una manera de evaluar la toma de decisiones de un usuario/instructor a la hora de seleccionar eventos y escenarios para asignárselos como programas de entrenamiento a los pilotos.

- Desarrollar un sistema de musica ambiental que suene mientras navegamos por los diferentes menús.

3.2. Especificación de requisitos software

A continuación procederemos a explicar cuales son los diferentes requisitos software, así como las diferentes partes del software y la interacción de los diferentes actores con estas.

Veremos en primer lugar una lista de requisitos funcionales, luego otra de no funcionales y finalmente procederemos a explicar las especificaciones que deben cumplir los diferentes módulos de nuestro software.

3.2.1. Requisitos funcionales

- **Creación del escenario**: Consiste en la creación de un escenario mediante la elección de diferentes eventos para que el piloto pueda reaccionar a ellos y entrenar en la competencias que carece.

- **Diferentes escenarios**: Creación de escenarios preconstruidos entre los que el instructor podrá elegir para la simulación de un piloto especifico.

- **Posibilidad de personalización**: Dar al instructor la posibilidad de elegir las condiciones y eventos de la simulación en vez de elegir entre una serie de casos predefinidos.

- **Añadir un piloto**: Dar la capacidad al instructor de poder añadir un nuevo piloto.

- **Selección del piloto**: La capacidad de poder elegir entre una serie de pilotos cada uno con un nivel diferente de competencias que tendrán que ser entrenadas.

- **Posibilidad de modificación de pilotos**: Dar al instructor la posibilidad de elegir un piloto y modificar su información en caso de errores.
- **Selección de *trigger event***: El instructor podrá elegir una serie de eventos que podrán ser lanzados durante la evaluación.
- **Evaluación de los pilotos**: Los instructores tendrán que evaluar las competencias de los pilotos durante la simulación.
- **Menús sencillos e intuitivos**: Intentar que sean fáciles de usar sin que hagan falta grandes explicaciones para configurar la simulación.
- **Creación de un aula virtual**: Consiste en una representación de la cabina de una aeronave para poder simular el comportamiento de los diferentes medidores e indicadores a las acciones del piloto. De esta forma el alumno puede acostumbrarse a los controles de la aeronave en un entorno seguro y tranquilo.
- **Elección en el Aula Virtual**: Dar la posibilidad de elegir entre el panel de control de las diferentes aeronaves representadas en nuestro caso un Airbus A220 y un Airbus H145

3.2.2. Requisitos no funcionales

- **Música**: La creación de un sistema de musica de fondo que además pueda ser activada y desactivada a voluntad.
- **Estética**: Aunque no sea muy importante dado que se centra mas en un conjunto de menús, mejorar las texturas y modelos que se muestran en el mapa.
- **Realismo**: De cara a la simulación que sea lo mas realista posible.
- **Ampliación Futura**: Crear este simulador de tal forma que sea posible y de la forma mas cómoda posible la ampliación del simulador, por ejemplo ampliando el numero de casos predefinidos, o ampliando el tipo de aeronaves disponibles.

3.2.3. Simulador de instructores

Para el simulador de instructores lo primero que se tendrá que hacer es crear una clase que guarde todas las variables necesarias para crear y contener la información de los pilotos y crear un menú que los contenga y permita seleccionarlos.

A continuación tiene que ser creado un menú que nos permita seleccionar si queremos crear un escenario desde cero o seleccionar un escenario predefinido, para esto tienen que existir clases que contengan la información necesaria tanto para crear los eventos como para saber que eventos forman un escenario.

Por supuesto teniendo esto en cuenta hay que crear los diferentes menús que nos permitan la selección de estos escenarios y su creación mediante una elección de eventos.

Con la misma información de los eventos, se debe crear un menú que permita la selección de *trigger event* que son los diferentes eventos que podrán ser lanzados por el instructor durante la evaluación.

A continuación hay que crear una pantalla donde se nos muestre la "simulación" y un instructor pueda calificar las capacidades mostradas por el piloto y lanzar los *trigger event* de los que se hablaba antes.

Y por último se tiene que crear un sistema que evalúe las elecciones tomadas por el instructor en función de si ha tomado las elecciones recomendadas por el simulador o no.

El usuario por otro lado lo que tendrá que hacer es, primero seleccionar un piloto, después elegir si quiere hacer desde cero un escenario o seleccionar uno ya creado previamente para entrenar al piloto, una vez creado este escenario deberá elegir los *trigger event* y por último evaluar la actuación del piloto y recibir un informe con sus elecciones, las cuales podrá explicar, si lo desea.

3.2.4. Aula virtual

Para la creación del aula virtual lo primero que se tendría que crear es un sistema de menús que permita simular las cabinas de las dos aeronaves que planteamos en las aulas virtuales, que son un avión, en concreto el Airbus A220 y un helicóptero que sera el Airbus H145.

Por último se tendrá que crear un panel que muestre el interior de la aeronave, mas concretamente la cabina con los mandos de la aeronave, para que el instructor pueda empezar con sus explicaciones.

En este caso el usuario principal y el que interactúa con la aplicación es el instructor que elegirá cual de las aeronaves se simulara en el aula y empezara a dar explicaciones apoyándose en la cabina mostrada en pantalla.

Por otro lado el piloto también se podría considerar un usuario aunque en este caso lo único que hace es visualizar la cabina mostrada mientras que el instructor da su explicación sobre los mandos de la aeronave.

3.2.5. Menús

Para crear los menús de la simulación recurriremos a los *widget* de Unreal Engine Motion Graphics (UMG) que es la herramienta que proporciona dicho motor gráfico para crearlos.

Como finalmente el simulador solo esta orientado para un usuario lo utilice activamente, sera el instructor el que tendrá que interactuar con los diferentes menús para conseguir el objetivo que pretenda conseguir.

3.2.6. Prueba de concepto

Para poder mostrar la utilidad de este simulador, y que pudiese ser evaluado por algunos especialistas, se realizó una prueba de concepto. Para ello se grabó un vídeo en el que se mostraba el simulador, aunque se limitaron algunas de la funciones del simulador y se mostraron algunos aspectos de la simulación aunque un no estaban acabados al nivel que deberían estar para considerarlos completamente satisfactorios.

Junto ha esta prueba de concepto se redactó un cuestionario que los especialistas que visualizaban el vídeo de la prueba de concepto debían de rellenar, la función de todo esto es que pudiesen dar su opinión sobre lo que habían visto de la aplicación y de esta forma poder ver si lo encontraban de utilidad o no y que fallos habían observado, de forma que de seguir trabajando en el simulador se pudiesen mejorar estos aspectos y conseguir un simulador completamente acabado a un mayor nivel de detalle que el que se consigue en este proyecto.

Capítulo 4

Metodología y herramientas

Para la realización de este proyecto, había que plantearse qué tipo de metodología podría ser la más útil, la que mejor se adecuase a este proyecto que teníamos que realizar. Para la realización del proyecto había que analizar las características del mismo y sopesar las opciones de las que se disponía.

Era un proyecto con una fecha clara de finalización y aunque había una idea inicial clara y bien definida, era muy básica y no había manera de saber a ciencia cierta en que podría acabar convirtiéndose el proyecto, por estas razones las opciones que se barajaron para el proyecto fueron utilizar alguna de las distintas metodologías ágiles que existen como por ejemplo Scrum, una metodología en cascada con distintas iteraciones o una metodología iterativa incremental.

Todas ellas permiten reaccionar relativamente rápido a los problemas o cambios que puedan surgir, dado que se hacen revisiones o bien en cada iteración o bien el los distintos sprint en el caso de las metodologías ágiles.

4.1. Metodología

Al final la metodología que fue elegida para el desarrollo del proyecto fue una metodología iterativa incremental. Hubo varias razones por las que finalmente fue esta metodología elegida, pero la más decisiva a la hora de tomar esta decisión era el hecho de que, aunque se tenía claro cuál sería la base del proyecto no sabía como se le daría forma, no se tenia claro en que se podría convertirse finalmente o si surgirían algunos problemas que obligases a modificar el rumbo del proyecto .

Por lo tanto, era bastante razonable utilizar esta metodología en la que se empezaría con una iteración con esa base que se tenía clara sobre el proyecto y después iteración tras iteración se le daría forma hasta llegar a la finalización del proyecto.

Como se decidió utilizar esta metodología había que atender a las diferentes etapas y componentes que la caracterizan es decir :

- **Etapa de inicialización**

  Se podría decir que es la iteración 0, marca la base de lo que será el proyecto y será sobre la que iremos construyendo todo.

- **Etapa de iteración**

  En esta etapa realizaran todas diferentes iteraciones, y en cada iteración se construirá un prototipo y lo analizaremos para ver que hay que modificar, añadir o quitar de cara a la siguiente iteración.

- **Lista de Control**

  No es una etapa pero es un componente importante a la hora de utilizar esta metodología, será modificado en cada iteración cuando analicemos los prototipos y nos marcaran el camino a seguir en futuras iteraciones.

Teniendo claros estos elementos de la metodología, se decidió que esta era la metodología que daría mejores resultados a la hora de utilizarla en este proyecto.

Y con esto claro y atendiendo a las etapas necesarias, estaba claro que habría dos iteraciones que serían obligatorias, lo que podríamos decir que era una iteración en papel y otra iteración utilizando ya el motor gráfico, aunque hubo mas iteraciones, se podría decir que estas dos iteraciones fueron las que marcaron dos bloques dentro del proyecto.

Estos bloques estaban muy diferenciados y marcaban un gran salto en el progreso del proyecto. Los bloques eran los siguientes:

- **Prototipado en papel**

  Aunque lo hemos llamado prototipado en papel la mayoría de iteraciones que pertenecen a este bloque fueron realizadas con *mockups*, aunque si hubo un prototipado inicial en papel, que fue la etapa inicial que fue utilizada como base.

  En este bloque podremos ver el crecimiento del simulador desde la base que teníamos clara, hasta lo que se convertiría finalmente.

  Este bloque consta de cuatro iteraciones a través de las cuales se fue moldeando el proyecto, desde el inicio del simulador hasta que parecía razonable que era necesario dar el siguiente paso que sería el segundo bloque.

- **Prototipado con el motor gráfico**

  En este bloque todo lo que se hizo fue ya con Unreal Engine 4 y se podría decir que lo que se hizo en este bloque fue dar forma al diseño, modificar algunas de las funcionalidades para que fueran coherentes con la interacción del instructor y dar los últimos detalles. De forma que se pudieran observar y utilizar las funciones que en el prototipado en papel eran imposibles.

  En este bloque se hicieron cuatro iteraciones, tres de ellas en las que se modificaron bastantes cosas hasta llegar ha grabar una prueba de concepto y una última en la que se arreglaron algunos aspectos incorrectos de la prueba de concepto anterior y que una vez arreglados se volvió a grabar.

Algunos aspectos de mejora planteados en la primera prueba de concepto no pudieron se realizados debido a la complicación de implementarlos y el tiempo de que se disponía.

4.1.1. Flujo de trabajo: Distintas iteraciones del Proyecto

Aunque ya han sido contadas brevemente antes algunas de las iteraciones que se realizaron, a continuación se detallará cual fue el flujo de trabajo y las iteraciones que se realizaron durante el proyecto.

Etapa inicial

Para esta etapa se analizaron que casos de usos eran necesarios en nuestro simulador y en base a eso se sacaron los requisitos funcionales del simulador.

En ese momento los casos de uso eran:

Por parte del instructor que era uno de los usuarios de este simulador la configuración de un aula virtual, seleccionar una simulación, configurar los parámetros de la simulación o elegir una simulación preconstruida.

Por parte del piloto por otro lado tendría que ejecutar el aula virtual y ejecutar la simulación.

En base a esto se sacaron los requisitos funcionales y no funcionales. Los funcionales relacionados directamente con los casos de uso y los no funcionales relativos a los parámetros configurables de los escenarios simulados.

Con todo esto en mente se realizó el prototipado en papel, en el que sería plasmado todo lo comentado anteriormente en bocetos y esquemas dando una primera imagen de lo que podría ser el simulador.

La lista de control que se obtuvo al finalizar esta etapa era:

- Pasar el prototipo a *mockup*

Etapa iterativa

Después de esta iteración cero se empezaron con las diferentes iteraciones, después de cada iteración se añadía o modificaba alguna funcionalidad que parecía que se tendría que añadir para acercarse a lo que se tenia en mente como objetivo final o bien porque había que modificar ligeramente el rumbo del proyecto dándole otro enfoque.

4.1.1.1. 1ºIteración

Esta primera iteración consistió en pasar la iteración cero a *mockups*, esto facilitaba el mandárselo a gente y poder hacer pruebas sin la necesidad de estar presencialmente, lo cual demostró ser muy útil.

Lista de control

- Poner un sistema de pilotos.
- Añadir la posibilidad de crear pilotos en ese sistema o seleccionarlos para que sean el objeto de la simulación.
- Añadir la posibilidad eventos ocultos que el instructor pueda lanzar o ejecutar a voluntad.
- Modificar la forma en que se muestra y se modifica la información, eliminar las imágenes en estos menús no son representativas de nada y podría confundir al instructor.

4.1.1.2. 2º Iteración

En esta segunda iteración se añadieron algunas funciones planteadas en la lista de control de la iteración anterior, estas funciones se mantendrían hasta el final del proyecto, aunque algunas sufrirían algunas modificaciones.

Estas funciones estaban relacionadas con la incorporación de los pilotos, aunque antes ya se los asumía como un usuario de la aplicación, ahora se veía la necesidad de tener guardada su información. Por lo tanto las funciones añadidas fueron el poder crear el perfil de un pilotos y el acceder a una lista donde poder verlos a todos y elegir así con cual trabajaríamos en su simulación. Como también se contemplo en la iteración anterior se añadió una función que permitiría al instructor la capacidad de seleccionar eventos que activaría durante la simulación del piloto, modificando su experiencia de forma repentina.

Lista de control

- Añadir la posibilidad de modificar la información contenida de los pilotos.
- Poder mostrar todos los eventos disponibles de forma que sea mas fácil para el instructor poder ver cual es el mas adecuado.
- Crear una pantalla en la que el instructor pueda evaluar las acciones realizadas por el piloto.

4.1.1.3. 3ºIteracion

Esta iteración se hizo pensando en que sería el último paso de prototipado con *mockups* y después se realizarían las iteraciones en Unreal Engine 4.

Al igual que con el resto de iteraciones se añadieron nuevas funcionalidades y se modificaron otras existentes atendiendo a lo contemplado en la lista de control. De este modo fue añadida la capacidad de poder modificar la información de los pilotos, de forma que ahora se podría añadir un piloto desde cero, modificar un piloto ya existente o seleccionarlo para la simulación.

También se añadió la posibilidad de que tanto en los eventos contenidos en los escenarios, como en los eventos que serían lanzados por el instructor se mostrasen todos los posibles y que se marcara los seleccionados. Y por último se añadió una de las partes mas importantes y que aunque se había planteado aun no se había añadido, es función era la capacidad para que el instructor pudiese evaluar al piloto en base a las acciones realizadas en el escenario que el mismo había creado o seleccionado.

Lista de control

- Pasar todo lo realizado a Unreal Engine

4.1.1.4. 4ºIteracion

Esta iteración consistió mayoritariamente en pasar la iteración anterior a Unreal Engine 4.

Se añadió la posibilidad de visualizar en la propia pantalla de evaluación lo que hacia el piloto. Cuando se intentó pasar lo diseñado en *mockups* al motor gráfico, hubo bastantes problemas de diseño respecto a como se mostraban las imágenes, botones, distribución de elementos en la pantalla e información.

Lista de control

- Arreglar los problemas de diseño
- Añadir una pantalla en la que se evaluará al instructor, respecto a las decisiones tomadas a la hora de elegir eventos.
- Modificar las visualizaciones, añadiendo visualizaciones del exterior y de la información de vuelo.

4.1.1.5. 5ªIteracion

En esta iteración se arreglaron los problemas con el diseño indicados en la iteración anterior, se añadió una pantalla en la que se mostraban los errores cometidos por el instructor a la hora de elegir los eventos, esta pantalla actuaba como evaluación del propio instructor y también se añadieron las visualizaciones extra, del exterior de la aeronave y de la información meteorológica y relativa al vuelo. Se añadieron y corrigieron todas las cosas indicadas en la lista de control anterior, aunque algunas no se consiguieron de forma satisfactoria.

Lista de control

- Modificar algunos textos mostrados para que muestren nombres y términos mas acordes con el ámbito de la aeronáutica y la aviación.
- Preparar una prueba de concepto.
- Añadir musica de fondo
- Mejorar el mapa mostrado respecto a lo que a texturas y modelos se refiere.

4.1.1.6. 6ºIteracion

Esta iteración se planteaba originalmente como la ultima , se cambiaron los nombres y términos mostrados, se añadió el sistema de audio aunque no funcionase correctamente, pues no reproducía el sonido aunque el botón funcionase y lanzase la pista de audio y el mapa mostrado fue mejorado con modelos y texturas de mas calidad.

Aunque no se consiguieron realizar todas las tares de la lista de control de forma satisfactoria, se realizo la grabación de la prueba de concepto, pero cuando se mostró a algunos especialistas, estos indicaron algunas cosas que deberían ser modificadas, así que se planteo una ultima iteración que además se aprovecho para limpiar algunas variables y funciones que al final perdieron su utilidad.

Lista de Control

- Poner correctamente las siglas de las competencias.

- La posibilidad de evaluar las calificaciones dadas por el instructor respecto a la actuación del piloto.
- Modificar las imágenes de las visualizaciones por modelos 3D o vídeos.
- Mostrar la información mediante los indicadores que vería el piloto.
- Revisar las clases limpiando las variables o funciones que ya estén obsoletas o no se usen.
- Arreglar el sistema de audio.

4.1.1.7. 7ºIteracion

En esta ultima iteración no se consiguió todo lo que se indicaba en la lista de control anterior, en gran parte por el tiempo del que disponíamos, pues a estas alturas era imposible dedicar tanto tiempo a por ejemplo modelar en 3D las cabinas o añadir vídeos sobre las simulaciones, que tendrían que ser personalizados para cada evento y piloto.

Otro aspecto que tampoco se planteó con el tiempo que disponíamos era la capacidad de poder evaluar el como calificaba el instructor a los pilotos en función de las acciones realizadas en la simulación, era algo muy subjetivo y difícil de cuantificar, pues lo que califica el instructor son comportamientos y reacciones , que aunque sujetos a una normativa cada instructor podrá dar mas o menos importancia, por lo tanto aunque se deja planteado no era viable a estas alturas del proyecto.

Aunque conscientes de la imposibilidad de realizar algunos de estos cambios era necesario incluirlos en la lista de control, de cara la posibilidad de que se continuase trabajando en este proyecto en un futuro.

Pero había cosas con las que si era posible lidiar y estas fueron abordadas y arregladas, como el sistema de audio que ya funcionaba correctamente y las siglas y términos mostrados que ya estaban corregidos.

Una vez hecho esto se volvió a grabar el vídeo de la prueba de concepto en el que se mostraban algunas pero no todas las utilidades del simulador, preparando todo para que algunos especialistas relativos al sector aeroespacial al que esta orientado este simulador lo evaluasen.

Aunque es la ultima iteración es conveniente y correcto poner una ultima lista de control, se plantea como trabajo futuro para poder continuar con este simulador y verlo convertido en algo mas grande de lo que es en estos momentos.

Lista de Control

- La posibilidad de evaluar las calificaciones dadas por el instructor respecto a la actuación del piloto.
- Modificar las imágenes de las visualizaciones por modelos 3D o vídeos, posibilitando incluso que sean simulaciones reales de un piloto.
- Mostrar la información mediante los indicadores que vería el piloto.

4.2. Herramientas

Para el correcto desarrollo de este simulador, fue necesario utilizar una serie de herramientas que abarcan una diversa variedad de aspectos dentro del desarrollo de un proyecto, como la programación o la comunicación por mencionar algunos de ellos. A continuación se explicarán los distintos aspectos y las herramienta utilizadas en cada uno de ellos.

4.2.1. Desarrollo

Para poder crear este simulador para instructores de vuelo las herramientas destinadas a su desarrollo son realmente imprescindibles, con esto nos referimos a todas aquellas herramientas destinadas a la planificación y programación del proyecto.

4.2.1.1. Balsamic Mockups

Esta herramienta es la que fue utilizada para hacer toda la parte destinada a la parte de prototipado en "papel", por lo tanto fue una herramienta muy necesaria sobre todo tuvo mucha importancia en el primer bloque de iteraciones en las que usaban "mockups".

Es muy útil porque esta destinada al diseño de interfaces gráficas y webs, y con su aspecto de bosquejos o croquis puede ayudar al desarrollador a crear en los estados iniciales del desarrollo un prototipo muy básico y sin ninguna función pero que ayuda a tener una forma clara de lo que se quiere conseguir. Por otro lado es capaz de convertir en un PDF los diseños realizados, permitiendo navegar entre los diferentes menús de forma coherente a lo diseñado. Este PDF puede presentarse como un prototipo inicial al cliente, para que de esta forma aunque no cumpla con las funciones requeridas el cliente pueda tener una idea del diseño y de la navegación entre las distintas ventanas o pantallas así como de su interacción con ellas.

4.2.1.2. Unreal Engine

Es el motor gráfico creado por Epic Game que fue decidido desde un primer momento que sería el utilizado para crear este simulador.

Inicialmente la idea del simulador estaba pensada más hacia los pilotos que hacia los instructores, (aunque la figura de los instructores también estaba presente y tenia mucha importancia) por lo tanto utilizar este motor gráfico tenia mucho más sentido, dada la carga visual de la simulación que sería vista por los pilotos. Pero conforme fue avanzando el desarrollo se fue perfilando un simulador mas orientado a los instructores de vuelo, y en el que la importancia visual se fue difuminando y no parecía tan importante dada la alta carga de menús que tendríamos.

A pensar de esta evolución se mantuvo la idea de utilizar Unreal Engine y estas fueron algunas de las razones.

Primero de cara a poder implementar la parte de simulación del piloto en un futuro, el no tener que utilizar otro motor o programa y luego unirlo teniendo que revisar todo completamente con los consecuentes errores que suelen ocurrir al juntar dos proyectos diferentes con dos motores o entornos diferentes.

Por otro lado el sistema de diseño de *widget* y *UMG* que tiene es muy intuitivo de usar y permite diseñarlo visualmente desde un inicio, en vez de crearlo en código y tener que compilarlo cada vez para comprobar si los cambios son los correctos.

También un motor gratuito de los mas importantes ahora mismo en el ámbito de los videojuegos, no deja de evolucionar y revisarse para poder solucionar posibles problemas o añadir nuevas funcionalidades. Respecto a este punto la otra posible opción a elegir sería Unity pero ya tenia conocimientos previos de Unreal Engine por lo que la elección de este motor respecto a Unity fue relativamente sencilla.

Aunque estaba orientado a videojuegos inicialmente, cada vez se esta siendo mas utilizando para hacer aplicaciones (sobre todo en el ámbito arquitectónico y de diseño). De esta forma podemos pensar en que este simulador no como un videojuego, si no como una aplicación didáctica, que se pueda utilizar en un futuro para que un instructor de vuelo pueda evaluar a un piloto y a la vez este sea evaluado por la aplicación permitiendo así el crecimiento de ambos. Y además contamos con la posibilidad de que esta aplicación pueda acabar en un futuro simulando completamente el escenario creado por el instructor con la mejor calidad visual en ambos casos o incluso plantear la posibilidad de utilizar realidad virtual, lo cual son puntos a favor para elegir Unreal Engine como motor gráfico.

Por otro lado ofrece la posibilidad de poder trabajar tanto con código como con "blueprint", en este casos en concreto se ha trabajado con "blueprint"pues la creación de menús mediante los "widget.ªl ser algo orientado principalmente a diseño era mejor trabajarlo de esta forma, además de esta forma se nos permitía ver el flujo del programa cuando navegábamos entre los diferentes menús lo cual es muy útil.

4.2.1.3. Unreal Motion Graphics (UMG)

Dentro de Unreal Engine, cabe destacar la importancia de UMG, sobre todo en un proyecto como con una carga tan alta de menús y *widget* como este.

Unreal Motion Graphics es una herramienta que aparece en la versión 4.5 y es especialmente útil en la creación y modificación de UI [1] y de menús.

Los *widget* mencionados anteriormente son el núcleo de UMG, son funciones prefabricadas que podremos usar para crear otros *widget* personalizados y más grandes o usar los básicos proporcionados para crear nuestra *interface*.

UMG utiliza dos pestañas para la creación y modificación de los *widget*, una para el diseño visual y otra para definir la funcionalidad del *widget* y cada uno de los componentes.

En este proyecto ha sido especialmente útil, dado la alta carga de menús y ha permitido definir tanto la realización del diseño visual de cada uno de los menús y pantallas, como la funcionalidad para que cada uno realizase las funciones pertinentes.

Y el uso de UMG ha sido una de las razones por las que se ha realizado la programación mediante los *blueprint* y nodos que Unreal Engine proporciona en vez programar utilizando código C++.

4.2.1.4. Visual Studio

Es un entorno de desarrollo que puede ser utilizado para múltiples lenguajes de programación. En este caso el lenguaje utilizado es C++ que en el que esta basado Unreal Engine.

La decisión de utilizar este en particular es porque se pueden instalar algunas funciones que ayudan al ahora de utilizar Unreal Engine.

[1] User interface

Aunque Unreal Engine tiene una interfaz gráfica que permite programar mediante el uso de "blueprintz o nodos, que fue principalmente lo que se uso para programar las funciones de esta aplicación, la utilización de Visual para ejecutar Unreal Engine permite hacer un mejor “debug” para encontrar errores. Además en casos como los problema con el sonido, en los que no era un problema de la programación o del diseño en si, pero algo hacia que se cerrase toda la aplicación de Unreal Engine, la utilización de visual como lanzador del motor se vuelve vital para capturar la excepción lanzada y poder buscar información sobre ella en caso de no conocerla.

También fue necesaria durante un periodo del desarrollo en el que se planteo utilizar MongoDB junto con Unreal Engine para tener una base de datos, en la que se pretendía guardar la información sobre pilotos, eventos ..., al final esta idea fue descartada, pues era mas conveniente utilizar diccionarios para poder guardar esta información.

4.2.2. Comunicación

Otro aspecto importante a la hora de desarrollar el proyecto era la comunicación entre los distintos miembros, para poder informar sobre el desarrollo del proyecto, hacer consultas o incluso para organizar y tener reuniones en caso de que no pudiesen ser presenciales. Este aspecto de las reuniones no presenciales fue especialmente importante en el desarrollo de este proyecto[2], por lo que fue imprescindible la utilización de herramientas de videoconferencia y en nuestro caso nos vimos obligados a usar varios programas diferentes en según que ocasiones pues hubo veces en las que fallaron algunos aspectos de las aplicaciones obligando a cambiar de aplicación o de plataforma.

Por estas razones hay varias herramientas que fueron utilizadas para este propósito.

[2]Debido a la crisis sanitaria ocasionada por la pandemia de Covid-19

4.2.2.1. Slack

Una herramienta de comunicación que era utilizada principalmente para comunicarme con mi director de proyecto. Esta herramienta facilita la comunicación entre los distintos miembros de un equipo de desarrollo y permite el poder tener varios equipos de desarrollo a la vez creando grupos para cada uno de ellos. Esta es una de las razones por las que se esta utilizando en algunas empresas, la capacidad de separar los distintos grupos de desarrollo dentro de un gran proyecto, pero mantener algunos grupos en los que todos se puedan comunicar.

4.2.2.2. Skype

Es un software que permite la comunicación mediante llamadas de audio o de video. Lo usamos mucho en las primeras fases del proyecto para discutir la documentación encontrada y el rumbo que teníamos pensado darle al proyecto o cuando no era posible hacer las reuniones de forma presencial. Pero cuando se avanzo en el proyecto y se necesitaba compartir la pantalla recurríamos a otras plataforma porque a pesar de que con Skype también se puede, estas otras herramientas permitían mas control a la hora de dar permisos o plantear preguntas sin interrumpirse.

4.2.2.3. Google Meet

Es un servicio de vídeo comunicación ofrecido por Google, recurríamos a el cuando era necesario compartir la pantalla o cuando las reuniones eran con mas personas a parte del director de proyecto y de mi. Pero en algunos casos tuvimos problemas con el audio y no nos escuchábamos, así que nos vimos obligados a recurrir a otra plataforma para las reuniones.

4.2.2.4. Collaborate

Es una herramienta proporcionada por el campus virtual, proporciona soporte para videoconferencias interactivas. Facilita el uso de pantallas compartidas y ofrece al moderador gran control sobre dar permisos al resto de participantes para poder compartir ficheros o participar durante la videoconferencia. No tuvimos ningún problema con esta herramienta y fue la que mas usamos al final.

4.2.3. Alojamiento compartido y control de versiones

También es necesario el ser capaz de compartir los recursos necesarios para la documentación y el desarrollo, así como los avances producidos en el proyecto durante su desarrollo y para eso las herramientas que decidimos usar son las siguientes.

4.2.3.1. Google Drive

Como sistema de alojamiento y compartición de ficheros utilizamos Google Drive, ya que para ficheros de documentación o para material de investigación que encontrábamos era mucho más cómodo. Además se fueron alojando las distintas presentaciones que se realizaron a lo largo del año para mostrar el avance del proyecto y las iteraciones y pruebas de concepto que se realizaron.

Se eligió esta herramienta sobre otros servicios de alojamiento en la nube debido a que posee herramientas de edición de documentos *on-line* con las que estábamos familiarizados y que nos permitía poder escribir actas de las reuniones rápidamente. Otra de las razones es porque poseemos almacenamiento ilimitado gracias a que contamos con cuentas proporcionadas por la Universidad Complutense de Madrid, que tiene un acuerdo con *Google* para ofrecer ventajas a los alumnos de dicha universidad.

Fue este también el método elegido para compartir el proyecto en si, dado que pudimos aprovechar ese espacio ilimitado solucionando las limitaciones que encontramos en otras herramientas a la hora de subir el proyecto.

4.2.3.2. *Perforce*

Es un sistema de control de versiones, se pensó en este en lugar de en usar *Github* por algunas dificultades como la limitación de no poder subir ficheros superiores 100 MB, que aunque mediante el uso de *Git LFS* pude solucionarse esta limitación, con el uso de Perforce no tenemos este tipo de limitaciones.

Además *Perforce* es capaz de conectarse directamente con Unreal Engine permitiendo subidas desde el propio proyecto facilitando y agilizando así su uso.

4.2.4. Redacción de la memoria

Aunque mediante notas a mano o archivos alojados en *GoogleDrive* nos pasábamos y guardábamos información necesaria para poder redactar la memoria del proyecto, era necesaria una herramienta que permitiese mantener el formato presentado por la Universidad Complutense de Madrid para la redacción de la memoria y facilitar de esta forma una maquetación sencilla coherente y que además permitiese el uso simultaneo por parte de varios usuarios, muy necesario a la hora de corregir y mostrar los errores contenidos durante la redacción.

4.2.4.1. *Overleaf* y *LaTeX*

Para maquetar el texto empleamos *LaTeX*, que permite la redacción y maquetación de documentos mediante el uso de código fuente, es bastante utilizado en el ámbito académico-científico. *Overleaf* es una herramienta *online* de edición de proyectos en *LaTeX*, que permite la implantación de plantillas preconstruidas facilitando así su uso.

Se decidió el uso de esta herramienta de escritura y edición de documentos como aplicación principal para desarrollar esta memoria. *Overleaf* es una herramienta *online* del conocido sistema de composición de textos *LaTeX* y que además permite la implantación de plantillas como por ejemplo la que está siendo usada, que es la que propone la Facultad de Informática de la Universidad Complutense de Madrid.

Capítulo 5

Análisis, diseño e implementación

Para explicar el proceso de desarrollo del sistema propuesto hay que conocer el uso de ciertas estructuras y tipos de datos que fue necesarios crear, y por otro lado debemos explicar todos los diferentes menús que se crearon para poder ir realizando cada una de las acciones necesarias.

A continuación se explicará la función, el tipo de utilización y el porqué de todas estas clases y menús interactivos de forma que sea lo más fácil posible entender las decisiones de diseño realizadas.

5.1. Clases y estructuras

Para la información que se tenia que guardar o usar era necesario crear algunas clases y estructuras propias. A continuación se procederá a explicar en que consisten cada una de estas clases y cual es su propósito.

5.1.1. *EAircraft*

Esta clase es un enumerado, en nuestro caso sólo se han añadido dos tipos de aeronaves, pero esta preparado para poder añadir tantos modelos de aeronave como sean necesarias, de manera que se podría elegir entre todas ellas a la hora de protagonizar un escenario de entrenamiento determinado.

5.1.2. *EPhasesFlight*

En este enumerado lo que se detallan son las distintas fases de vuelo existentes. Se utilizará para indicar la fase de vuelo en la que ocurren los distintos eventos, de forma que los eventos se podrán clasificar impidiendo que se pueda producir un evento exclusivo de una fase en otra diferente y no relacionada.

Las diferentes fases son:

- **ALL** Si aplicamos esta fase a un evento querremos indicar que es un evento genérico que puede aparecer en cualquiera de las distintas fases de vuelo.
- **GND (Ground)** Esta fase se refiere a cuando la aeronave aun se encuentra en tierra, o aún esta en el rodaje hasta llegar al sitio indicado para empezar con la siguiente fase. Respecto a esta fase se ha dividido en "first" y "end" indicando así si es la fase inicial antes de despegar o la fase final cuando ya se ha aterrizado.
- **TO (Take-off)** Esta fase es el momento en el que el piloto aumenta el empuje de la aeronave con el propósito de despegar. Finaliza en el momento en que se consigue una cierta altitud y velocidad indicadas para poder continuar con la subida.
- **CLB (Climb)** Esta fase da inicio cuando se establece una altitud y velocidad indicadas para poder continuar el ascenso de la aeronave hasta poder alcanzar la altitud y velocidad óptimas para entrar en la fase de crucero.
- **CRZ (Cruise)** Esta fase dará inicio cuando se alcanza una cierta altitud y la velocidad de crucero, y se mantiene hasta que se comience con el descenso que es la siguiente fase.
- **DES (Descend)** Esta fase sigue al abandono de la velocidad de crucero con una reducción de la altitud y concluye cuando se inicien los cambios necesarios para facilitar ya la aproximación a pista.

- **APP (Approach)** Esta fase da comienzo cuando se consigue una configuración de velocidad y altura necesarios para poder iniciar la aproximación a una pista de aterrizaje determinada y finaliza cuando la configuración permite iniciar la toma de tierra.

- **LDG (Landing)** En esta fase es cuando tomamos tierra en la pista, momento en el que da inicio la fase y se finalizará cuando termina el rodaje hacia el estacionamiento y detención total del aparato.

5.1.3. *SInfoPilot*

cc

Figura 5.1: Representación de la estructura de datos referente a la información del piloto

Esta estructura esta creada para poder tener junta toda la información del piloto. Se utiliza principalmente al añadir un nuevo piloto o modificar uno cuyos datos ya estuvieran en la base de datos.

Esta información también será necesaria a la hora de calcular qué competencias son las que el piloto necesita entrenar.

Esta estructura también se utiliza en la *Game Instance*, dentro de uno de los diccionarios representa al tipo de los valores. Este diccionario se utiliza como una "base de datos" donde se guardarán los datos de todos los piloto.

La razón de utilizarse en un diccionario dentro de la *Game Instance* es porque las variables utilizadas en esta clase actúan como variables globales y por lo tanto siempre son accesibles, pudiendo así acceder a la información del piloto siempre que sea necesario.

A continuación serán explicados cada uno de los atributos contenidos en esta estructura:

- **Image** Es un atributo de tipo *texture*, permitirá incluir la imagen del piloto para poder ser mostrada a la hora de crear la lista de pilotos.
- **Name** Es un atributo de tipo *string* necesario para poder guardar el nombre de nuestro piloto.
- **LastName** Es un atributo de tipo *string* necesario para poder guardar los apellidos de nuestro piloto.
- **Age** Es un atributo de tipo *integer* necesario para poder guardar la edad que tiene nuestro piloto.
- **IAPK** Es un atributo de tipo *float* necesario para poder guardar el valor que tenia nuestro piloto en la competencia *Application of procedures* cuando fue añadido inicialmente.
- **ICOM** Es un atributo de tipo *float* necesario para poder guardar el valor que tenia nuestro piloto en la competencia *Comunication* cuando fue añadido inicialmente.

- **IFPA** Es un atributo de tipo *float* necesario para poder guardar el valor que tenia nuestro piloto en la competencia *Flight path management, automation* cuando fue añadido inicialmente.

- **IFPM** Es un atributo de tipo *float* necesario para poder guardar el valor que tenia nuestro piloto en la competencia *Flight path management, manual control* cuando fue añadido inicialmente.

- **ILTW** Es un atributo de tipo *float* necesario para poder guardar el valor que tenia nuestro piloto en la competencia *Leadership and teamwork* cuando fue añadido inicialmente.

- **IPSD** Es un atributo de tipo *float* necesario para poder guardar el valor que tenia nuestro piloto en la competencia *Problem solving and decision making* cuando fue añadido inicialmente.

- **ISAW** Es un atributo de tipo *float* necesario para poder guardar el valor que tenia nuestro piloto en la competencia *Situation awareness* cuando fue añadido inicialmente.

- **IWLM** Es un atributo de tipo *float* necesario para poder guardar el valor que tiene actualmente nuestro piloto en la competencia *Workload management* cuando fue añadido inicialmente.

- **AAPK** Es un atributo de tipo *float* necesario para poder guardar el valor que tiene actualmente nuestro piloto en la competencia *Application of procedures.*

- **ACOM** Es un atributo de tipo *float* necesario para poder guardar el valor que tiene actualmente nuestro piloto en la competencia *Comunication.*

- **AFPA** Es un atributo de tipo *float* necesario para poder guardar el valor que tiene actualmente nuestro piloto en la competencia *Flight path management, automation.*

- **AFPM** Es un atributo de tipo *float* necesario para poder guardar el valor que tiene actualmente nuestro piloto en la competencia *Flight path management, manual control.*
- **ALTW** Es un atributo de tipo *float* necesario para poder guardar el valor que tiene actualmente nuestro piloto en la competencia *Leadership and teamwork.*
- **APSD** Es un atributo de tipo *float* necesario para poder guardar el valor que tiene actualmente nuestro piloto en la competencia *Problem solving and decision making.*
- **ASAW** Es un atributo de tipo *float* necesario para poder guardar el valor que tiene actualmente nuestro piloto en la competencia *Situation awareness.*
- **AWLM** Es un atributo de tipo *float* necesario para poder guardar el valor que tiene actualmente nuestro piloto en la competencia *Workload management.*
- **NotesInstructor** Es un atributo de tipo *string* que sera utilizado para poder añadir notas o comentarios que tiene que comentar un instructor sobre el piloto en cuestión. Está pensado para que el usuario, en su papel de instructor, ponga por escrito las observaciones sobre el piloto a evaluar que considere convenientes..
- **Difficulty** Es un atributo de tipo *integer* necesario para poder asignar una "dificultad" al hecho de entrenar a este piloto, lo cual en realidad es un simple atajo para indicar si el sistema debe o no mostrar recomendaciones sobre los eventos que son adecuados para introducir en el escenario con el que se piensa entrenar las competencias del piloto.

Con todos estos atributos tendremos toda la información que necesitamos en este simulador para crear un piloto y poder trabajar con él.

5.1.4. *SInfoEvent*

Esta estructura es la utilizada para guardar la información que caracteriza como es un evento y que necesitaremos a la hora de clasificarlos y de calcular, por ejemplo, si el instructor ha seleccionado un evento que no ha sido recomendado por el sistema.

Figura 5.2: Representación de la estructura de datos referente a la información de los eventos

A continuación y viendo cómo es la estructura en la imagen, se explicarán cada uno de los atributos que la componen.

- **EventSet** Este atributo es un *string* en el se da una descripción del evento en cuestión, para que el instructor sepa que está seleccionando.
- **FlightPhase** Es un enumerado de la clase *EPhaseFlight* que hemos creado para poder clasificar los eventos entre las diferentes fases del vuelo.
- **APK** Es un *boolean* que indicará si este evento es indicado para entrenar la competencia APK.
- **COM** Es un *boolean* que indicará si este evento es indicado para entrenar la competencia COM.

- **FPA** Es un *boolean* que indicará si este evento es indicado para entrenar la competencia FPA.
- **FPM** Es un *boolean* que indicará si este evento es indicado para entrenar la competencia FPM.
- **LTW** Es un *boolean* que indicará si este evento es indicado para entrenar la competencia LTW.
- **PSD** Es un *boolean* que indicará si este evento es indicado para entrenar la competencia PSD.
- **SAW** Es un *boolean* que indicará si este evento es indicado para entrenar la competencia SAW.
- **WLM** Es un *boolean* que indicará si este evento es indicado para entrenar la competencia WLM.

5.1.5. *SInfoMeteorology*

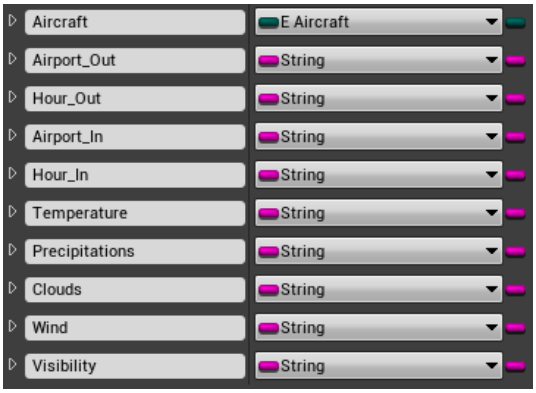


Figura 5.3: Representación de la estructura de datos referente a la información meteorológica y del vuelo

Esta estructura se utiliza para guardar la información que se muestra al escoger un escenario preconstruido, o que guardaremos si escogemos construir desde cero nuestro propio escenario de entrenamiento. En ella se muestra información relativa al vuelo pero también información sobre la meteorología que se espera durante la simulación.

A continuación se explicarán los diferentes campos que contiene esta estructura.

- **Aircraft** Este campo es de tipo *EAircraft*, se indicará el tipo de aeronave que se va a utilizar para la simulación.
- **AirportOut** Este campo indicará cual es el aeropuerto desde el que saldrá la aeronave y sera de tipo *string*, normalmente el nombre del aeropuerto viene marcado también por el nombre de la ciudad en que se encuentra este.
- **HourOut** Representado con un *string*, indicará la hora de salida de nuestro vuelo.
- **AirportIn** Este campo indicará cual es el aeropuerto en el que aterrizará la aeronave y sera de tipo *string*, normalmente el nombre del aeropuerto viene marcado también por el nombre de la ciudad en que se encuentra este.
- **HourIn** Representado con un *string*, indicará la hora a la que llegará al destino nuestro vuelo.
- **Temperature** Se representará mediante un *string* e indicará cuál sera la temperatura que se espera durante el trascurso del vuelo.
- **Precipitation** Refiriéndose a las precipitaciones, es decir la cantidad de lluvia (none, light, medium, heavy) que habrá durante la duración del vuelo, viene indicado mediante un *string*.
- **Cloud** Con esta información se dará una referencia de como se espera que este el cielo en lo que a nubes se refiere, si esta despejado o si por el contrario estará muy cubierto. Esta información se representará mediante un *string*.

- **Wind** Con este campo lo que se indica es la velocidad del viento, y esta se guardará mediante un *string* en el que almacenamos su valor.

- **Visibility** Guardado mediante un *string*, este campo indicará cómo es la visibilidad que tiene el piloto desde la cabina.

5.1.6. SPredCase

Figura 5.4: Representación de la estructura de datos referente a la información necesaria para un escenario predeterminado

En esta estructura se guardará la información que define un escenario preconstruido, y que es necesario para mostrar los eventos que pertenezcan a dicho escenario y la información de vuelo que se muestra.

A continuación se explicarán cada uno de los campos de esta estructura y cual es su función.

- **Description** Es una breve descripción del escenario en la que se mostrará un poco de información y se dirá a que competencia está mas orientado. Para guardar esta información utiliza el tipo *string*.

- **InfoMeteorology** Viene definido por un tipo *EInfoMeteorology*, y se utilizará para guardar la información necesaria para poder mostrar la información sobre el vuelo y la meteorología.

- **EventSelect** Este campo es un *array de Integer*, cada una de las posiciones del *array* define una de las fases de vuelo, por lo tanto este *array* sera de ocho elementos. Y por otro lado el valor de cada posición mostrará el identificador del elemento que ha sido asignado a esa fase de vuelo.

5.1.7. SInfoEventEvaluation

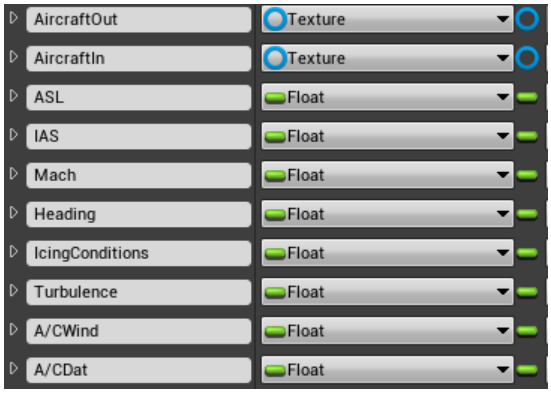


Figura 5.5: Representación de la estructura de datos referente a la información necesaria para evaluar a un piloto

Esta estructura es la necesaria para poder guardar la información que será mostrada en la pantalla de evaluación. Esta información es la referente a las imágenes visualizadas que hacen la función de "simulación" y la información referente a la meteorología y el estado del avión que se muestra y es modificada cada vez que cambia el evento en la evaluación.

Una vez explicado esto pasaremos a explicar cada uno de los campos de esta estructura para que quede aun mas clara su utilidad.

- **AircraftOut** De tipo *texture*, este campo tiene la función de poder guardar la imagen del exterior de la aeronave. En esta imagen podremos ver el estado de del avión, viendo cómo está reaccionando a lo que hace el piloto.

- **AircraftIn** De tipo *texture*, con este campo podremos, al contrario que con el campo anterior, ver el interior de la aeronave, mas concretamente la cabina. De esta forma el instructor podría ver como reacciona el piloto y qué acciones realiza dentro de la cabina mientras está pilotando.

- **ASL** Son las siglas de *Above sea level*, indicará la altitud sobre el nivel del mar y se mide en pies, será indicado en esta estructura mediante un tipo *float*.

- **IAS** Son las siglas de *Indicated airspeed*, normalmente se mide en millas por horas o en nudos, y para mostrarlo se usará un tipo *float*.

- **Mach** Es una medida de velocidad utilizada en aeronáutica y que hace referencia a la velocidad del sonido, tomado como referencia. En esta estructura se utilizará un tipo *float* para poder indicarla.

- **Heading** Medida en grados, indica la inclinación del morro de la aeronave y se utilizará un tipo *float* para poder indicar su valor.

- **IcingConditions** Utilizado para indicar las condiciones de formación de hielo durante el vuelo y se utilizará un tipo *float* para poder medirlo.

- **Turbulence** En esta estructura utilizaremos un tipo *float*, y lo que se indica son las turbulencias durante cada evento, se utilizará un porcentaje para indicar el nivel de turbulencias que hay.

- **A/CWind** Indica el ángulo con el que incide el viento con la aeronave durante el vuelo y se utilizará un tipo *float* para poder mostrar su valor.

- **A/CDat** Este campo se usa para indicar la temperatura del aire durante el vuelo y se utilizará un tipo *float* para poder mostrar su valor.

Figura 5.6: Representación de la estructura de datos referente a la información necesaria para saber los eventos erróneos o seleccionados

5.1.8. SErrorReport

Esta es una estructura que es utilizada para poder indicar los eventos que pertenecen al escenario y los *trigger event* que elegirá el instructor para lanzar manualmente durante la simulación del entrenamiento.

Esta estructura se utiliza para ver cuáles son las elecciones que hace el usuario a la hora de modificar un escenario preconstruido o al configurar uno desde cero, y también es el que se utilizará para ver cuáles son los errores que comete el instructor, entendiendo por errores aquellas elecciones que no son recomendadas por el simulador.

Los distintos campos son:

- **Event** Es un tipo *array de integer* y su utilidad es indicar las elecciones del instructor hechas sobre los eventos pertenecientes a los escenarios. La posición indicará la fase de vuelo y el valor indicará el identificador del evento.
- **Hide Event** Este campo también es un tipo *array de integer*, las posiciones y los valores indicarán lo mismo que el campo anterior pero en este caso, los eventos son los seleccionados por el instructor para los *trigger event*.

5.1.9. GISimGI

Es una instancia del *game instance*, su utilidad es la de guardar como variables globales la información sobre pilotos eventos...

Utilizamos esta clase sustituyendo la base de datos, que no se gestiona en un sistema externo para evitar complicar la instalación y no obligar a que exista conexión a internet.

En esta clase para guardar toda la información necesaria se hará uso de diccionarios, todos ellos con claves de tipo *integer* y respecto a la clase de sus valores cada una es diferente en función de la información que guarden. A saber:

- **ListStudent** De tipo *SInfoPilot*, se utiliza para guardar la información de los pilotos.
- **ListEvent** De tipo *SInfoEvent*, se utiliza para guardar la información de los diferentes eventos.
- **ListCase** De tipo *SPredCase*, se utiliza para guardar la información de los escenarios que podemos escoger.
- **ListEvaluationEvent** De tipo *SInfoEventEvaluation*, se utiliza para guardar la información mostrada al instructor durante el periodo de evaluación de los pilotos.

5.2. Menus y Widgets

Una vez explicadas las diferentes estructuras y enumerados que fue necesario crear para tratar la información que requerida para el simulador, hay que explicar la función que tiene cada menú y *widget* así como las funciones y variables que contienen.

Como es un simulador y hay que configurar bastantes aspectos de la simulación, tiene una alta carga de menús entre los que hay que navegar. Aunque algunos son sencillos, en otros hay que hacer bastantes cálculos necesarios tanto para crear el propio menú como para tratar la información y trasmitirla.

Debido a todo esto, a continuación se explicarán cada uno de los menús, así como los *widget* y su función dentro el conjunto de menús que forman el simulador.

Figura 5.7: Widget que representa un caso concreto dentro de un escenario

5.2.1. WEventCase

Este *widget* es el que se utiliza dentro del simulador para representar los diferentes eventos que contiene un escenario, por lo tanto tiene tanto la información del evento en sí como algo de información extra que necesita el simulador para crear el escenario y poder evaluar al instructor.

5.2.1.1. Variables

A continuación se explicarán las diferentes variables que contiene este *widget* así como sus tipos y función dentro del *widget*.

- **InfoEventCase** Esta variable es de tipo *SInfoEvent*, se utilizará para poder guardar la información del evento en cuestión y después de ser guardada, utiliza la información para poder modificar el *widget* de tal forma que represente el evento.
- **Key** Es de tipo *integer*, representa la clave que identifica a este evento dentro del diccionario donde tenemos guardados todos los eventos.
- **Select** Es de tipo *boolean*, se utiliza para saber si el evento puede ser seleccionado, o su selección esta bloqueada pues pertenece a un escenario predefinido.
- **Recomen** Es de tipo *boolean*, indicará si el simulador estima oportuno que el evento es recomendado para entrenar alguna competencia que el piloto tiene baja o cuya evaluación está suspensa.

- **CheckBox de Competencias** Para explicarlos se han identificado aquí como *CheckBox* de Competencias pero en realidad son un conjunto de variables. Son variables de tipo *CheckBox* y cada una representa una competencia del piloto, en función de si está seleccionado o no, mostrará si el evento es indicado para entrenar esa competencia del piloto.

- **Description** Es de tipo *text* y permitirá escribir una breve descripción del evento en cuestión para que el instructor pueda hacerse una idea sobre en qué consiste el evento que va o no a seleccionar.

- **Phase** También una variable de tipo *text*, indicará la fase de vuelo a la que pertenece este evento, de forma que es mas fácil clasificarlo.

- **Recomend** Esta variable es de tipo *Image* y contiene un asterisco rojo. Su utilidad es mostrar si el evento se recomienda para el piloto de tal forma que si se recomienda el asterisco sera visible pero si no es recomendado el asterisco no se mostrara.

- **Select** Es de tipo *boolean*, e indicará si el evento ha sido seleccionado o no para pertenecer al escenario.

5.2.1.2. Funciones y Macros

A continuación se explicará el conjunto de funciones necesarias para poder crear o modificar el *widget* y dejarlo de la forma que se necesite.

- **InitEventCase** Esta función es la que se utilizará para inicializar el *widget*, de tal forma que se pasarán como parámetros, la información del evento, la clave del evento, si esta seleccionado o recomendado y también la dificultad del piloto. Con todos estos parámetros recorreremos todas las variables de *widget* y las modificaremos en consecuencia y una vez hecho esto el *widget* estará creado.

- **ActivateSelect** Esta función sera utilizada para poder indicar que el evento puede ser seleccionado, es decir que habilitará el *checkBox* que permite seleccionar el evento.

- **ActivateRecomend** Recibiendo como parámetro el valor *recomend*, esta función modificará la variable correspondiente y decidirá en función de este valor si la imagen debe ser visible o invisible mostrando así si el evento esta recomendado o no para el piloto.

- **SelectEvent** Es igual que la función explicada anteriormente, pero en este caso lo que se hace es seleccionar o no al evento para que pertenezca al escenario.

5.2.2. MStudentPhoto

Figura 5.8: Widget con la información e imagen del piloto que utilizaremos para seleccionarlo

Este *widget* tiene la función de mostrar cierta información del piloto para que pueda ser seleccionado, pero en su interior contiene toda la información sobre el piloto.

5.2.2.1. Variables

- **InfoPilot** Esta variable es necesaria para poder guardar toda la información sobre el piloto en cuestión, para ello esta variable necesita ser de tipo *SInfoPilot*, que es la estructura creada para este propósito.

- **Key** Esta variable es de tipo *integer* y su utilidad es la de guardar la clave que tiene este alumno en el diccionario. Esto es útil porque de esta forma no necesitaremos pasar la estructura anterior al siguiente menú y solo pasaremos un *integer* que obviamente ocupa menos espacio.

- **Modify** Esta variable de tipo *boolena* indicará si se ha seleccionado este piloto para modificar su información o simplemente para empezar a configurar un escenario para él.

- **Profile** Esta variable se utiliza para mostrar la imagen del piloto en cuestión y por lo tanto su tipo es *image*.

- **Pilot** Es de tipo *buttom*, y por lo tanto define un botón que permitiría al usuario ser capaz de seleccionar a este piloto.

- **Name** Esta variable de tipo *string* sera utilizada para poder mostrar en el *widget* el nombre y los apellidos del piloto.

- **Easy, Medium, Hard** Estas son tres variables diferentes, pero son del mismo tipo *image* y que permitirán identificar la dificultad de entrenar a este piloto en función de cuales de ellas sean visibles.

5.2.2.2. Funciones y Macros

La única función que se encuentra en este *widget* es *InitPhoto* y su cometido es el de inicializar los valores de todas las variables posibles con los atributos que le pasamos, que son *modify*, *key* e *info pilot*, de tal forma que una vez hecho todo devolverá un *MStudentPhoto* totalmente creado.

Por otro lado, y aunque no es una función como tal, al pulsar el botón de este *widget* y en función del valor de *modify* llamará a la inicialización del menú que corresponda, o bien *MInfoStudentModify* o bien *MInfoStudent*.

5.2.3. WCaseDescription

Figura 5.9: Widget representativo de un escenario preconstruido

5.2.3.1. Variables

- **Case** Esta variable será utilizada para poder guardar toda la información referente al escenario creado desde cero o a un escenario preconstruido y por lo tanto el tipo de la variable es *SPredCase*.
- **KeyCase** De tipo *integer*, esta variable será útil para guardar la clave que ocupa este escenario en el diccionario.
- **KeyPilot** Al igual que la variable anterior es referente a la clave dentro del diccionario, pero en este caso es la clave del estudiante que había sido seleccionado para simular este escenario.
- **CaseDescription** Esta variable de tipo *text* será útil para poder indicar una pequeña descripción del escenario en cuestión.
- **CaseSel** Es una variable referente al botón que necesitamos para poder seleccionar este escenario.
- **NumCase** De tipo *text* lo usaremos para poder identificar al escenario a la hora de seleccionarlo.

5.2.3.2. Funciones y Macros

Tiene una única función denominada *InitCaseDesc* y será la encargada de poder inicializar la información con los atributos que se le pasan, devolviendo así un *widget* completamente inicializado.

Aunque no es una función definida como tal al pulsar el botón se inicializará el menú correspondiente al que se expandirán las claves que son necesarias para poder conseguir cierta información necesaria para formar dicho menú o que se necesitará en menús posteriores.

5.2.4. WSound

Figura 5.10: Widget que representa el botón de sonido

Este *widget* consta únicamente de un botón cuya función será la de activar o desactivar la música de fondo.

5.2.4.1. Variables

- **BActivate** Esta variable de tipo *boolean* indicará si el audio esta activado o desactivado.
- **Activate** De tipo *text*, indicará visualmente en el botón si el audio esta activado o no.
- **Audio** Esta variable identifica al botón en sí, siendo de tipo *buttom*.
- **Music** Esta variable de tipo *sound cue* es donde se guarda la pista de audio, y donde se le dice que permanezca en bucle.

5.2.4.2. Funciones y Macros

Aunque no cuenta con funciones como tal contiene un par de eventos. Uno de ellos es el evento de construcción que indica que nada mas crearse este *widget* se debe desactivar el audio y mostrarlo en la variable *activate*, por otro lado el otro evento es el referente a la pulsación del botón y que hará de forma intermitente la activación y desactivación de la musica mostrándolo en el texto de *activate*.

Habiendo explicado todos los *widget* auxiliares que se necesitan para construir ciertos menús como se requiere, a continuación se explicarán cada uno de los menús.

5.2.5. MInitMenu

cc 

Figura 5.11: Es el menú inicial del simulador

Es el menú que se muestra inicialmente cuando se abre el simulador y muestra las opciones que se pueden realizar.

Estas funciones son: en primer lugar comenzar una simulación, lo que llevará a tener que elegir un piloto y seleccionar un escenario para él, otra de las opciones es el aula virtual, que serviría para que el instructor pueda dar una clase mostrando los mandos de la aeronave para apoyarse en sus explicaciones, y por último quedará la opción de salir de la aplicación.

5.2.5.1. Variables

Las variables de este menú representan los botones asignados a cada una de las opciones que podemos elegir en este menú y que han sido explicadas anteriormente.

5.2.6. MMenuVirtualClass

cc

Figura 5.12: Es el menú que permite seleccionar el tipo de aula virtual

Este es el menú que aparece cuando se quiere dar una clase virtual y lo que muestra son las posibilidades de volver hacia atrás, o escoger las aulas virtuales de una aeronave, ya sea un avión, en este caso el Airbus A220, o un helicóptero, el Airbus H145.

5.2.6.1. Variables

Las variables de este menú al igual que el anterior son las que representan las diferentes opciones que tenemos y que se han explicado anteriormente.

5.2.7. WInitPlane y WInitHelicopter

Figura 5.13: Pantallas de inicio de las diferentes aulas virtuales

Estos dos menús se explican juntos porque son iguales, sólo cambia el aula virtual al que pueden acceder. Es un menú de transición para iniciar el aula virtual o volver atrás.

5.2.7.1. Variables

En estos menús las únicas variables que tienen vuelven a ser los botones con las posibilidades de volver hacia atrás o de iniciar el aula virtual.

5.2.8. MControlPlane y MControlHelicopter

En esta pantalla se puede decir que uno está oficialmente en el aula virtual. Según sea la del avión o la del helicóptero, cambiarán los mandos que se mostrarán, de forma que el instructor enseñará basándose en los mandos mostrados, bastante realistas, para que un piloto los pueda visualizar para hace más claras las explicaciones.

Figura 5.14: Pantallas de las aulas virtuales

5.2.8.1. Variables

Las únicas variables que tienen son la imagen en la que se mostrarán los mandos y el botón para dar por finalizada la clase y volver al menú inicial.

5.2.9. MMenuPilot

cc

Figura 5.15: Menú con las opciones sobre los pilotos

Este sera el menú al que se llegará en el caso de escoger la simulación en el menú anterior. En este menú se podrá seleccionar, añadir o modificar a un piloto, dado que para iniciar la selección o creación de un escenario de entrenamiento primero se necesita tener claro para quien esta destinado.

5.2.9.1. Variables

En este menú las únicas variables, como en los últimos menús explicados, son los botones que representan las diferentes opciones. A saber: seleccionar, añadir o modificar un piloto; existiendo también la opción de retroceder.

5.2.10. MAddPilot

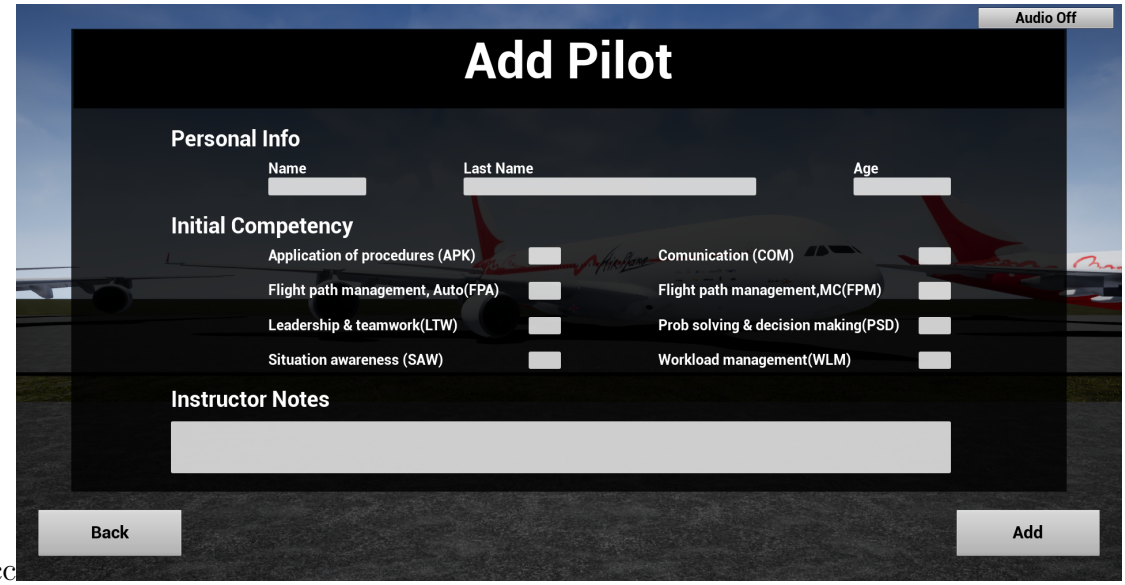


Figura 5.16: Pantalla donde añadiremos al piloto

Hasta este menú se llegará si se elige añadir un piloto. Su función será la de añadir toda la información necesaria para crear un piloto, de forma que se añada al diccionario donde guardamos todos los pilotos para que pueda ser mostrada cuando sea necesario.

5.2.10.1. Variables

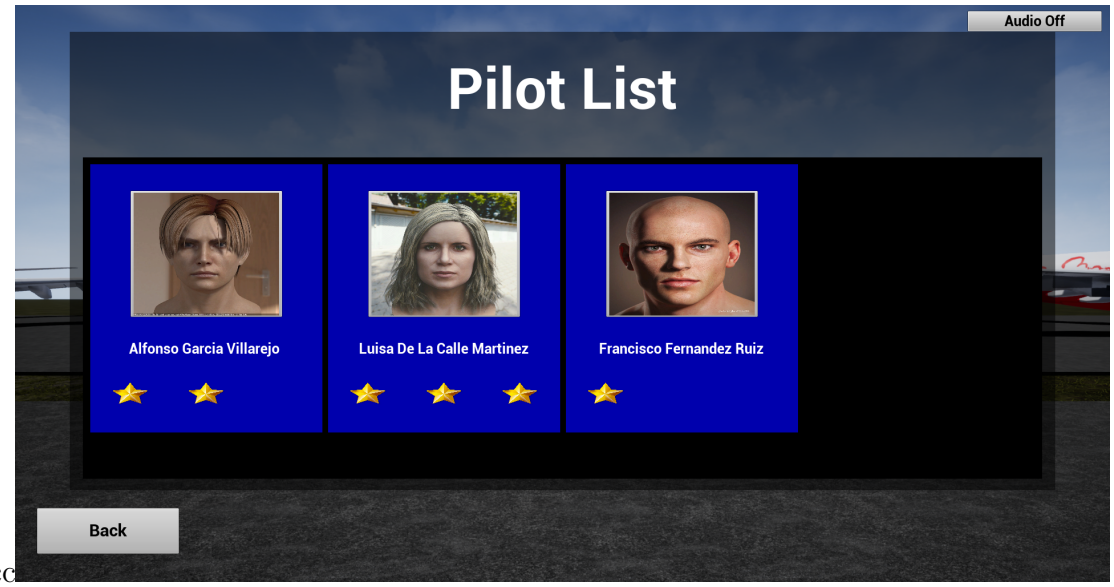
- **InfoStudent** Esta variable será la utilizada para ir añadiendo toda la información para que pueda guardarse, por lo tanto es necesario que sea de tipo *SInfoPilot*.
- **SEditableName y SEditableLastName** Esta variable es de tipo *text box*, un tipo que permite que se pueda escribir en su interior, y en este caso se utilizará para escribir tanto el nombre como los apellidos del piloto, de tal forma que puedan ser guardados.
- **SEditableAge** Esta variable es de tipo *text box* como la anterior, pero en este caso sera usada para guardar la edad del piloto.
- **MultiLineEditableNotes** También de tipo *text box*, pero esta permite escribir varias lineas y sera usada para escribir notas o comentarios que el instructor desea añadir sobre el piloto.
- **BAdd y BBack** Son los botones cuya función es o bien aceptar la información del piloto para poderla añadir al diccionario, o bien volver hacia atrás en el menú.
- **MWarning** Esta variable de tipo *MWarning* es una ventana emergente para confirmar si queremos guardar los datos del piloto.

También contiene variables de tipo *text box* para cada una de las competencias y cuya utilidad es la misma que en las anteriores, poder guardar información, en este caso sobre el valor de las competencias del piloto

5.2.10.2. Funciones y Macros

- **SaveInfoStudent** Con esta función lo que se hace es coger toda la información que hay en los *text box* y se le dan valores a todos los campos de la variable *InfoStudent*. Una vez que ya están todos los valores se guardarán en el diccionario correspondiente a los pilotos, de forma que se añadirá un nuevo piloto.

5.2.11. MPilotList y MPilotListModify



cc

Figura 5.17: Pantalla con la lista de los pilotos

Estos menús mostrarán una lista con todos los posibles pilotos que hay, en uno de los casos al seleccionar al piloto se le permitirá modificar la información del mismo, pero en el otro caso lo que se permitirá sera fijarle como el piloto elegido para crear su escenario de simulación, pero visualmente son iguales, sólo cambia su objetivo.

5.2.11.1. Variables

La única variable a resaltar es *Row* que al usar un *grid panel* es la que ayudará a poder cambiar de final cuando lo necesitemos. El resto de variables son las que representan al propio panel o a los botones.

5.2.11.2. Funciones y Macros

Respecto a las funciones solo contiene una, que es la que se encarga de borrar el menú cuando un piloto es seleccionado. Su nombre es *ClickingEvent*.

5.2.12. MInfoPilot

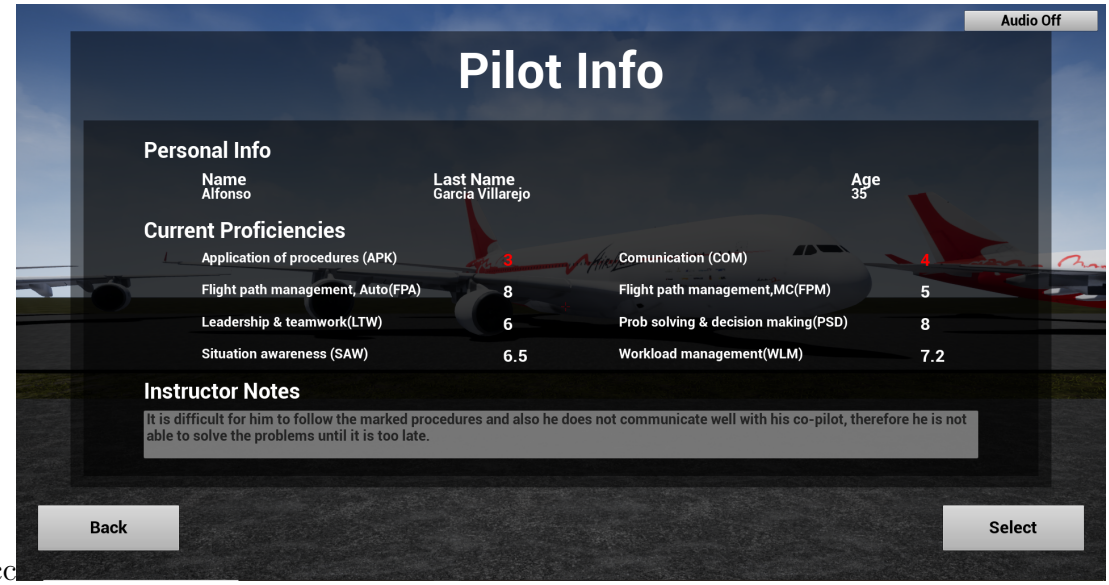


Figura 5.18: Pantalla con la información del piloto

Este menú se mostrará si cuando se decide seleccionar a un piloto y lo que hará sera mostrar la información del piloto, de forma que se conozca la información antes de empezar a crear un escenario para su simulación.

5.2.12.1. Variables

- **Key** Esta variable de tipo *integer* identifica la clave perteneciente a un piloto específico en el diccionario donde esta la información de todos los pilotos.
- **InfoStudent** En esta variable es donde se guarda toda la información del piloto que es mostrada en este menú

El resto de variables contenidas en este menú hacen referencia a botones para poder continuar o retroceder, y también son las que representan el texto que es modificado para poder mostrar la información.

5.2.12.2. Funciones y Macros

La única función de este menú es la que inicializa todos los valores mostrados con la información del piloto y recibe el nombre de *startValues*.

5.2.13. MInfoPilotModify

Este menú es muy parecido al utilizado para añadir un piloto, la única diferencia es que en este caso hay más campos a modificar pues por cada competencia podremos modificar el valor inicial que tenia el piloto y el valor que tiene actualmente, dado que no está añadiendo un nuevo piloto si no modificando uno ya existente.

5.2.13.1. Variables

Las variables de este menú son las mismas que las del menú explicado anteriormente, la única modificación es que en este caso la información la tiene que poner el propio instructor, por lo tanto en vez de ser variables de tipo *text* son variables de tipo *text box*.

5.2.13.2. Funciones y Macros

- **VerifValue** Esta función se encarga de verificar si los valores referentes a las calificaciones de las diferentes competencias son correctos y no se salen de los parámetros permitidos.
- **SaveVarG** Con esta variable se podrán guardar todos los valores añadidos por el instructor como la información que hay que modificar del piloto.

5.2.14. MMenuSimulation

En este menú, una vez que ya se tiene seleccionado el piloto objetivo se dará la opción de elegir un escenario predefinido, o bien construir desde cero un escenario.

cc 

Figura 5.19: Menú que permite elegir el tipo de escenario

5.2.14.1. Variables

- **Key** Esta variable de tipo *integer* identifica la clave perteneciente a un piloto especifico en el diccionario donde esta la información de todos los piloto.

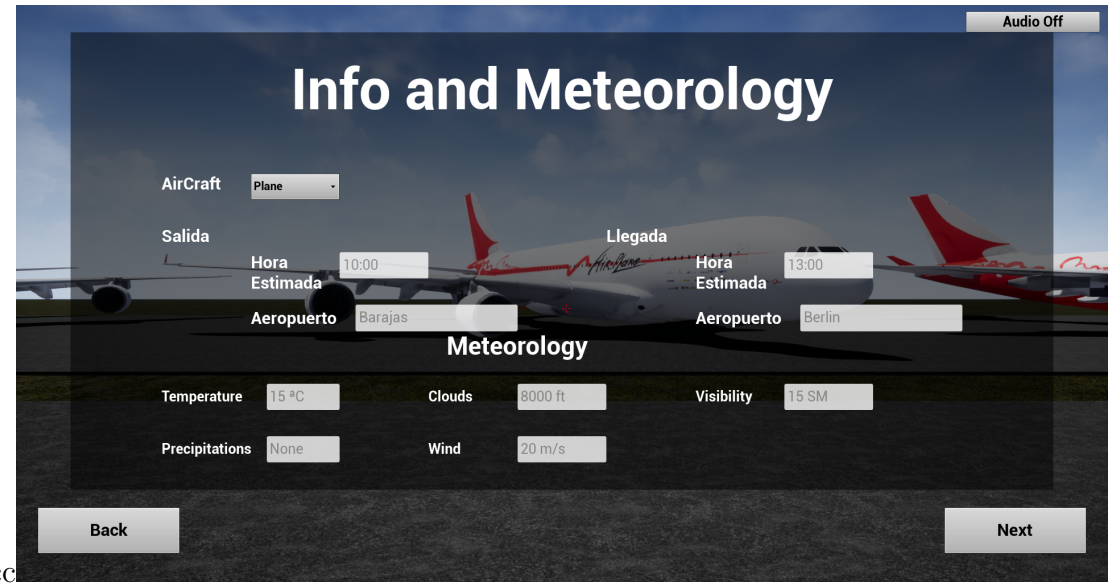
El resto de variables son aquellas que representan los botones con las diferentes acciones con las que cuenta este menú.

5.2.15. MMeteorologyConf

Este menú tiene la función de poder modificar la información de vuelo y meteorológica, de forma que el instructor pone él las información que estima oportuna para la simulación.

5.2.15.1. Variables

- **KeyStudent** Esta variable de tipo *integer* representa a la clave que tiene el piloto seleccionado en el diccionario donde se guarda su información.



cc

Figura 5.20: Pantalla donde se podrá configurar la información meteorológica y del vuelo

- **InfoMeteo** Es una variable donde guardaremos toda la información sobre el vuelo y la meteorología, es de tipo *SInfoMeteorology*.

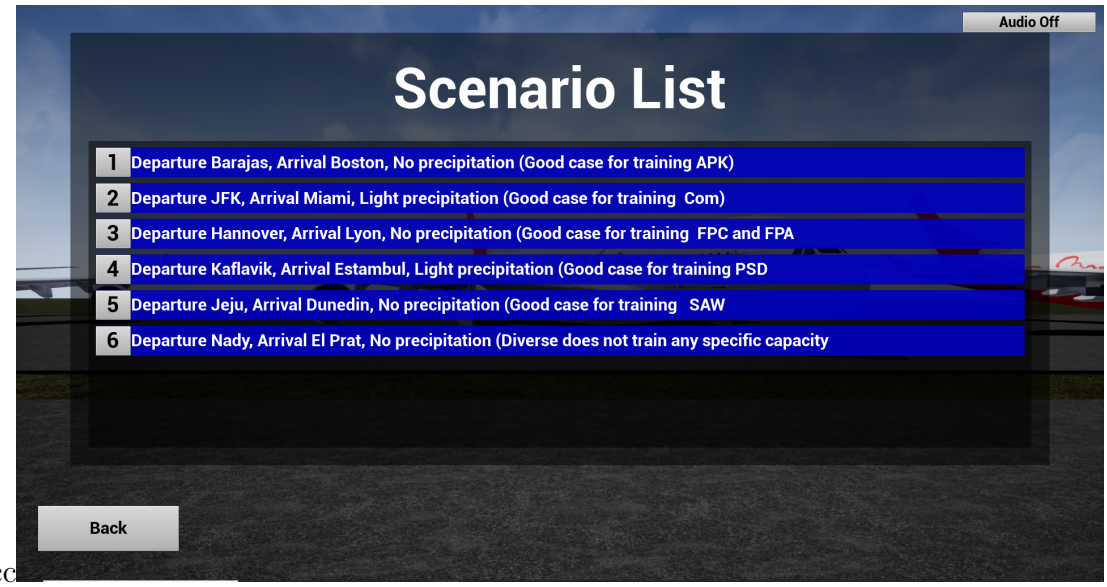
El resto de variables que tiene este menú son aquellas que representan a *text box* y son de ese mismo tipo, cada una se encarga de recibir información sobre algún aspecto específico.

5.2.15.2. Funciones y Macros

Sólo tiene una función que es la encargada de guardar toda la información necesaria sobre el vuelo y la meteorología, su nombre es *SaveInfo*.

5.2.16. MScenarioList

Este menú contiene la lista de todos los escenarios posibles de que dispone el simulador, de forma que muestra para cada uno de ellos una breve descripción y su identificador donde se podrá seleccionar.



cc

Figura 5.21: Menú donde seleccionaremos el escenario preconstruido

5.2.16.1. Variables

- **KeyStudent** Esta variable representa la clave que tiene el piloto seleccionado para evaluar, de tal forma que se podrá buscar su información en el diccionario.
- **BBack** Representa el botón que permitirá volver hacia atrás en caso de desearlo.
- **VerticalPanel** Esta variable identifica a un panel vertical en el que se introducen todos los *widget* de los diferentes escenarios disponibles.

5.2.16.2. Funciones y Macros

Respecto a las funciones sólo contiene una que es la que se encarga de borrar el menú cuando un piloto es seleccionado su nombre es *ClickingEvent*

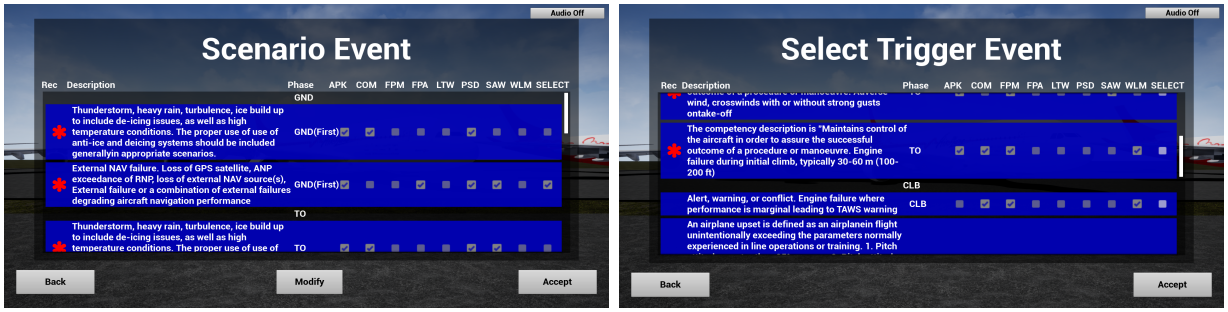


Figura 5.22: Menús de los escenarios predefinidos donde elegiremos los eventos que lo forman

5.2.17. MPredCaseTriggerEvent y MInfoPredeterminateCase

La razón de poner estos dos menús juntos es su similitud, tanto en función como en contenido.

MInfoPredeterminateCase permite visualizar todos los eventos posibles, marcando cuales están seleccionados para el escenario preconstruido que hemos elegido pero permitiendo si lo deseamos modificar estos eventos y seleccionar otros diferentes.

MPredCaseTriggerEvent por otro lado lo que permite es ver los eventos para poder seleccionar aquellos que el instructor quiera usar como *trigger event.*

Por lo demás tanto estéticamente como en su contenido de variables y funciones son muy similares

5.2.17.1. Variables

Como estamos viendo las variables de dos pantallas y aunque hay muchas similitudes se indicará cuando alguna variable pertenece solo a una de las pantalla.

- **Error y Selection** Estas dos variables están contenidas en los dos menús y tendría sentido que fuese de otra forma, la razón de esto es que el tipo de esta variable es *SErrorReport*. *SErrorReport* es un tipo que consta de dos *arrays* uno de ellos identificando los eventos normales del caso y otro que identifica a los *trigger event* tanto si es porque son seleccionados para pertenecer o porque son erróneos pues han sido seleccionados sin ser recomendados. Sabiendo esto hay que explicar que cuando se completa un *array* en *MInfoPredeterminateCase* se pasa la variable a *MPredCaseTriggerEvent* para poder completar la variable con la información que falta.

- **KeyStudent** Esta variable de tipo *integer*, indicará la clave que tiene el piloto seleccionado en el diccionario que contiene la información de los pilotos. En estos menús sera útil cuando necesitemos información del piloto para poder recomendar los eventos y conocer cuales son sus peores competencias. También es una información que vamos transmitiendo entre los diferentes menús.

- **KeyCase** Esta variable de tipo *integer* solo estará en el menú *MInfoPredeterminateCase*. Esto se debe a que es la clave de un escenario contenido en el diccionario que guarda la información sobre los escenario y como en este menú es donde hay que mostrar los eventos que pertenecen al escenario, necesitaremos conocer la información del escenario.

El resto de variables que identificaremos en estos menús son los que utilizaremos para identificar los paneles de cada fase de vuelo donde se guardan los *widget* de los eventos y también tienen en común los botones que tienen la función de avanzar o retroceder, excepto *modify* en *MInfoPredeterminateCase* que permitirá modificar los eventos que pertenecen al escenario.

5.2.17.2. Funciones y Macros

- **BeginConstruct** Esta función se encargará de recorrer todos los eventos guardados en el diccionario y ponerlos en el menú clasificándolos según la fase de vuelo correspondiente, de forma que construye el menú tal y como debe mostrarse al instructor.

- **AddAll** Esta función se utilizará cuando un evento pertenezca a todas las fases de vuelo por lo tanto deberá clasificarlos en todas y cada una de las fases. La única información que necesitará es la clave del evento para poder tener acceso a su información, si esta seleccionado, es decir si pertenece al escenario y el *widget* del evento en sí.

- **OnlyOne** Esta función es la encargada de evitar que en una misma fase de vuelo tengamos mas de un evento seleccionado, por lo tanto cuando un evento es seleccionado quita la selección de los demás en caso de estar seleccionados.

- **WorstGrade** Con esta función somos capaces de identificar cuál es la peor calificación que tiene un piloto en sus competencias y por lo tanto cual es la competencia en la que más tendremos que enfocarnos a la hora de elegir eventos para entrenarlo.

- **Recomend** Esta función indicará si un evento esta recomendado para un piloto atendiendo a cuál es su peor calificación y a qué competencias entrena dicho evento.

- **CreateEventWidget** Crea un *widget* de un evento concreto para poderlo añadir a los diferentes paneles que representan las distintas fases de vuelo, esta función sólo se utiliza si el evento esta clasificado para todas las fases pues necesita crear duplicados del *widget* para cada uno de los paneles.

- **ActivateModify** Esta función es la que permite que los eventos se puedan seleccionar pues en otro caso estarán desactivados inicialmente.

- **Select/ErrorEvent** Esta función permite, cuando hemos decidido dar el siguiente paso, guardar las elecciones hechas en el menú actual y ver los errores cometidos por el instructor al elegir, para poder transmitir estas elecciones y fallos al siguiente menú.

5.2.18. MConfCase y MConfTriggerEventCase

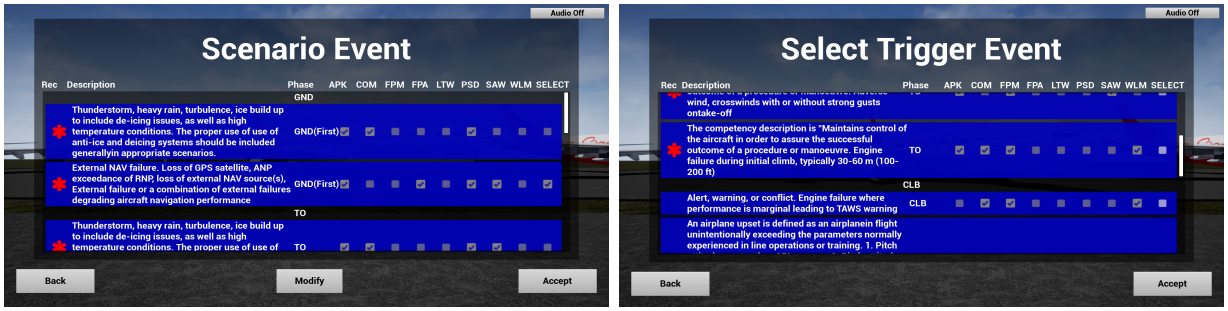


Figura 5.23: Menús donde construiremos un escenario desde cero seleccionando los diferentes eventos

Sobre estos dos menús no se entrará en detalles sobre sus funciones y variables, esto se debe a que son homólogos a los explicados anteriormente, la única diferencia es que los anteriores trabajan sobre un escenario predefinido, pero en el caso de estos están construidos completamente desde cero. Esto tiene más importancia en *MConfCase* que en *MConfTriggerEventCase*, dado que en el *MConfTriggerEventCase* da igual si el escenario esta preconstruido o no.

5.2.19. WEvaluation

Esta es la pantalla mas importante de la aplicación desarrollada, pues en ella es donde el instructor verá la simulación para poder evaluar al piloto en función de su comportamiento y elecciones tomadas durante el desarrollo del entrenamiento.

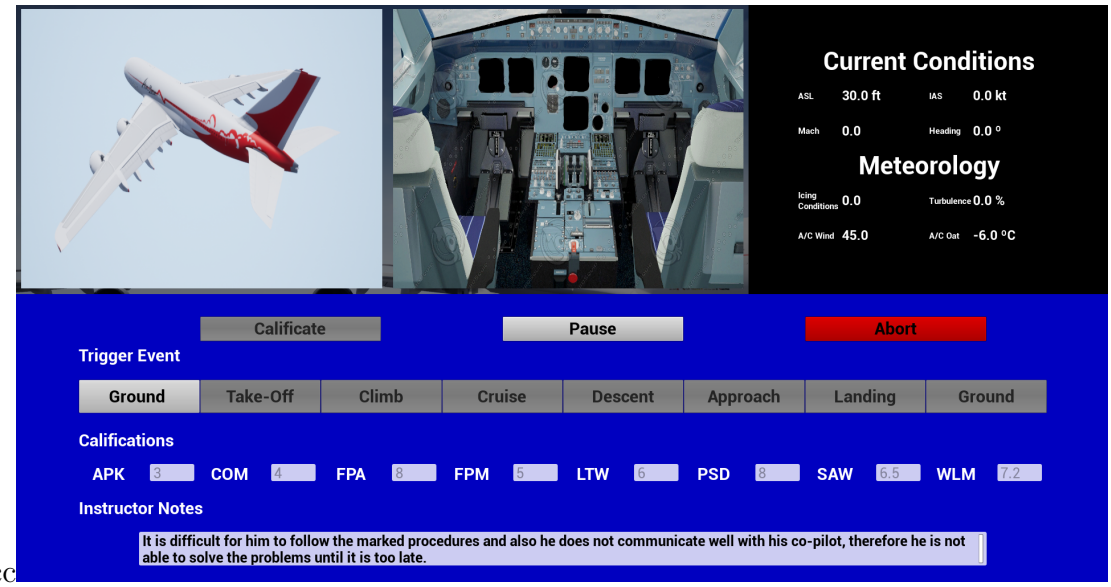
cc 

Figura 5.24: Pantalla donde el instructor evaluará al piloto

Para empezar la pantalla esta dividida en dos partes: la parte superior, donde se mostrará la simulación e información relevante sobre la misma, y la parte inferior, en la que el instructor deberá seleccionar qué evento lanzar y finalmente calificar la actuación del piloto.

En la parte superior se ven tres pantallas, una mostrará el exterior de la aeronave, otra mostrará el interior de la cabina donde el instructor podría ver el comportamiento del piloto en los mandos, y por último otra pantalla en la que se mostrará información sobre el vuelo y la meteorología.

Ahora en las pantallas sólo se muestran imágenes fijas e información muy simplificada, pero a futuro lo ideal sería que en estas pantallas se mostrasen vídeos o incluso simulaciones reales, y respecto a la pantalla con información, aunque ahora muestra información muy simplificada, en un futuro lo ideal sería que se mostrasen todos los indicadores que vería el piloto dentro de la cabina, de tal forma que las imágenes de la cabina se utilizasen exclusivamente para monitorizar la actuación del piloto.

5.2.19.1. Variables

- **KeyStudent** Esta variable de tipo *integer* identifica la clave del alumno que está siendo evaluado, de tal forma que podremos buscarlo en el diccionario para poder modificar sus notas en función de las calificaciones que le pondrá el instructor.

- **Time** Con esta variable de tipo *float* se podrá controlar el tiempo que transcurre de forma que se activen o desactiven los diferentes botones en función de la fase de vuelo en la que se encuentra.

- **Selection y Errors** Estas dos variables están construidas de la misma forma pues las dos son del tipo *SErrorReport* y lo que hacen es indicar las elecciones y errores respectivamente que hemos hecho en el escenario y en los *trigger event*.

- Pause Es una variable de tipo *boolean* que indicará si el tiempo esta pausado o no, bloqueando en consecuencia la activación de los botones correspondientes a los *trigger event*.

El resto de variables son las que identifican a los botones y *text box*. Respecto a los botones se utilizan para lanzar los diferentes *trigger event* y estarán activados o desactivados según la fase de vuelo. Por otro lado también tenemos los botones que permitirán pausar la evaluación o dar por finalizada la evaluación, calificando finalmente al piloto.

Y también tenemos las *text box* en las que el instructor pondrá las notas para cada una de las competencias del piloto, escribiendo también, en caso de desearlo, notas y observaciones sobre el comportamiento del piloto durante la simulación.

5.2.19.2. Funciones y Macros

- **Func(...)H** La parte con puntos suspensivos indicará el nombre de la fase de vuelo. En estas funciones se modificarán según corresponda por la fase vuelo y el *trigger event*, las imágenes e información mostrada en la parte superior de la pantalla. Esta información vendrá determinada por la elección de *trigger event* que el instructor ha hecho previamente.

- **Func(...)** La parte con puntos suspensivos indicará el nombre de la fase de vuelo. En estas funciones se modificarán según corresponda por la fase vuelo, las imágenes e información mostrada en la parte superior de la pantalla, esta información vendrá determinada por la elección de eventos que el instructor ha hecho previamente para definir el escenario.

- **FuncPilotInfo** Esta macro escribirá las calificaciones del piloto en las *text box* para que el instructor sepa las notas que tiene actualmente el piloto. También escribirá las notas del instructor para permitir ver las notas previas escritas por este mismo u otros instructores.

- **CalcNotes** Esta función calcula si se incrementa o disminuye la nota actual del piloto y en qué cantidad, teniendo en cuenta si la nota recibida es mayor o menor a la nota que tenía actualmente.

- **ModificationsNotes** Habiendo calculado cuanto variará la nota del piloto con la función anterior, esta función se encargará de coger la información del piloto y modificarla aplicando la variación de la nota calculada.

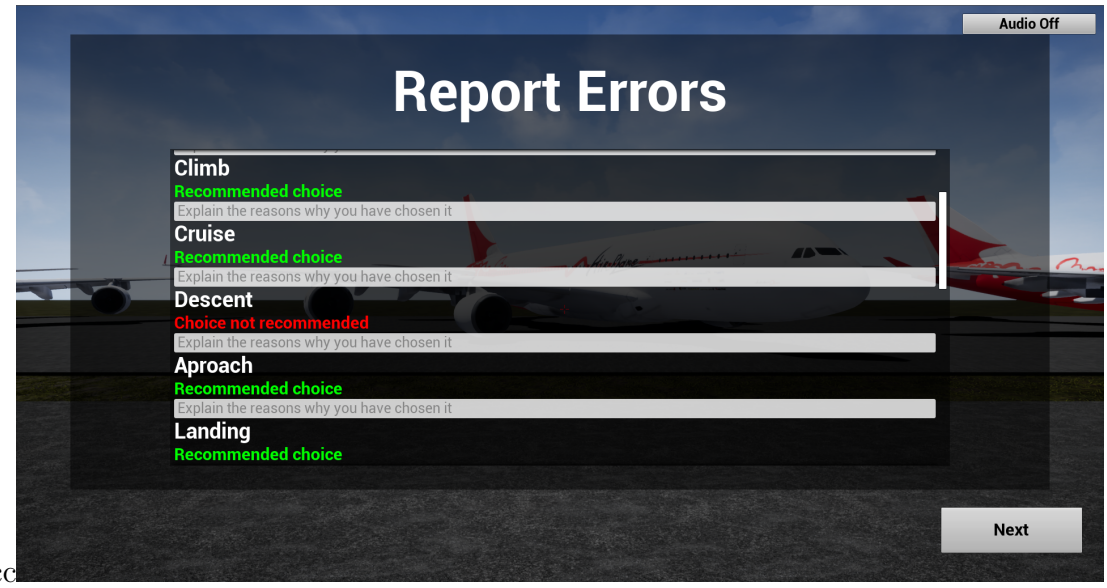
cc 

Figura 5.25: Pantalla donde el instructor recibe su evaluación

5.2.20. MReportErrors

Esta es una de las pantallas mas importantes del simulador, pues lo que hará sera decirle al instructor cuáles han sido los errores que ha cometido a la hora de configurar el escenario para el piloto. Sólo muestra si la elección que ha hecho es diferente a la que el simulador cree que es la recomendada, pero con esto el instructor ya puede hacerse una idea de cuáles son sus fallos.

5.2.20.1. Variables

- **Errors** Esta variable de tipo *SErrorReport* es usada para comprobar cuáles han sido los errores cometidos al elegir los eventos. Consta de dos *arrays*, uno referente a los eventos normales del escenario y otro a los *trigger event*, de forma que si el valor de alguna posición en estos *arrays* es distinta de cero, demostrará que se ha cometido un error.

- **TextError y TextOk** En estas variables se guardarán los mensajes que se muestran cuando se ha hecho una mala o buena elección respectivamente, por lo tanto es necesario que sean de tipo *string*.

- **Abort** Es una variable de tipo *boolean* que indica si el instructor ha decidido abortar la simulación.

El resto de variables son todas las que hacen referencia a textos que son modificados en función de si son buenas o malas elecciones y también las que identifican las *text box* donde el instructor deberá indicar los motivos de las elecciones que se han tomado en caso de querer explicarlas y clarificar la razón por las que pensó que tenía que elegirlas.

5.2.20.2. Funciones y Macros

Sólo tenemos una función denominada *WriteText*. Esta función es la que se encarga de escribir el mensaje indicando si la elección es correcta o no modificando tanto el texto como el color de la fuente según corresponda.

5.2.21. MCongrats

cc 

Figura 5.26: Pantalla de enhorabuena

No es tanto un menú como una pantalla de transición, en la que después de haber evaluado a un piloto y ser informado de los errores que se han cometido como instructores a la hora de elegir los eventos, se da un mensaje donde se felicita al usuario por haber terminado, de forma que ahora volverá al menú inicial donde se podría o bien salir del simulador o bien comenzar otra evaluación.

5.2.21.1. Variables

La única variable de esta pantalla es la que representa al botón que hará volver al usuario al menú inicial al ser pulsado.

Capítulo 6

Pruebas y resultados

Para poder comprobar si realmente un simulador para instructores de vuelo es útil o al menos para tratar de responder si el propuesto en este proyecto lo es, se realizo una prueba de concepto con intención de validarla antes especialistas del sector aeroespacial. Esta prueba de concepto consistía en un breve vídeo mostrando el funcionamiento paso a paso de la aplicación, donde es posible conocer y evaluar la funcionalidad del sistema. En dicho vídeo se pasaba por los diferentes menús, comentando cada posible actuación del usuario en el papel de instructor, y señalando algunas de las funciones principales que tiene este simulador.

La idea de esta prueba de concepto es que, una vez mostrada a los especialistas, estos respondiesen un formulario para poder identificar: por un lado, cómo se encuentra actualmente la formación respecto a los pilotos e instructores, y por otro, una evaluación específica de la prueba de concepto de nuestro simulador, para escuchar sus opiniones sobre errores que deberían corregirse para la versión final, y otras mejoras en las que no se ha pensado aún, pero que sería interesante realizarlas, según la opinión de los especialistas.

Debido a la crisis ocasionada por la pandemia de COVID-19, la evaluación de la prueba de concepto no ha podido realizarse de manera presencial como estaba previsto inicialmente. Debido a ello, se optó por distribuir el vídeo con un enlace a un cuestionario en línea entre nuestros especialistas. Hemos publicitado este evento de realización de pruebas a través del sitio web y las redes sociales de Narratech Laboratories.

6.1. Resultados

A continuación serán explicados y analizados los resultado obtenidos en la encuesta realizada a 13 especialistas del sector aeroespacial, conocedores de la problemática del pilotaje de aeronaves. Para empezar hay que decir que la encuesta se ha dividido en tres grandes bloques:

- Especialista. Se pregunta información personal al especialista, nada íntimo que pueda incomodarle, pero sí suficiente información como para que destaque su pertenencia al sector aeroespacial al que esta orientado este proyecto, y que la cantidad de experiencia que posee.

- Sector. Se hacen preguntas orientadas a conocer la opinión del especialista sobre el sector aeroespacial, aunque sea una opinión subjetiva gracias a las respuestas de todos ellos es posible hacerse una idea general de cómo le está yendo al sector.

- Simulador. Una vez que conocemos al especialista y su sector, es hora de conocer qué piensan sobre el simulador de instructores propuesto, o al menos sobre las explicaciones sobre el mismo que se muestran en el vídeo. Las preguntas están enfocadas sobre todo a conocer fallos producidos, sugerencias de mejora y saber si la aplicación parece lo suficientemente intuitiva y sencilla de usar como debería.

6.1.1. El especialista

Respecto a las preguntas para conocer los antecedentes profesionales de los especialistas hemos preguntado principalmente por los puestos que ocupan el nivel de formación y años de experiencia en el sector. Todos los especialistas trabajan en el sector aeroespacial y cuentan en general con bastantes años en el sector.

Respecto a los trabajos que desempeñan, las respuestas han sido:

- Piloto

- Instructor de vuelo
- Instructor de vuelo de helicópteros
- Ingeniero de diseño - Special Fabs - ITP Aero
- Oficial en Comando de Control Aeroespacial
- Jefe de Instrucción y entrenamiento
- Jefe Dpto Académico Escuela de Oficiales
- Jefe de la sección operaciones del Ala Aérea N3
- CFI [1]
- Miembro de la Fuerza Aérea del Perú

Al observar los distintos puestos podemos observar que muchos de ellos son en áreas dedicadas a la enseñanza y la instrucción dentro del ámbito aeroespacial, que es concretamente el usuario objetivo que tiene este simulador. Respecto a los años de experiencia se encuentran mayoritariamente entre los 15 y los treinta años, lo cual es un bagage importante para acreditar su valía como evaluadores del sistema.

Y por último, en lo referente a la nacionalidad todos los especialistas son de nacionalidad española o peruana, por lo tanto debemos asumir que las respuestas sobre el sector que abordaremos en el siguiente bloque están orientadas a sus experiencias en estos países, aunque también es razonable pensar que hayan desarrollado su trabajo en otros países distintos del de origen y que las respuestas no serían muy diferentes.

6.1.2. El sector

En este bloque todas las preguntas están orientadas a conocer la opinión del especialista sobre el ámbito aeroespacial y sus sugerencias sobre como podría mejorar la formación en el mismo.

[1] Jefe de instrucción en vuelo

Las preguntas, para las que se pedían general una respuesta en la escala de Likert de 5 valores, fueron:

- **La formación actual en pilotaje de aeronaves es suficiente.**

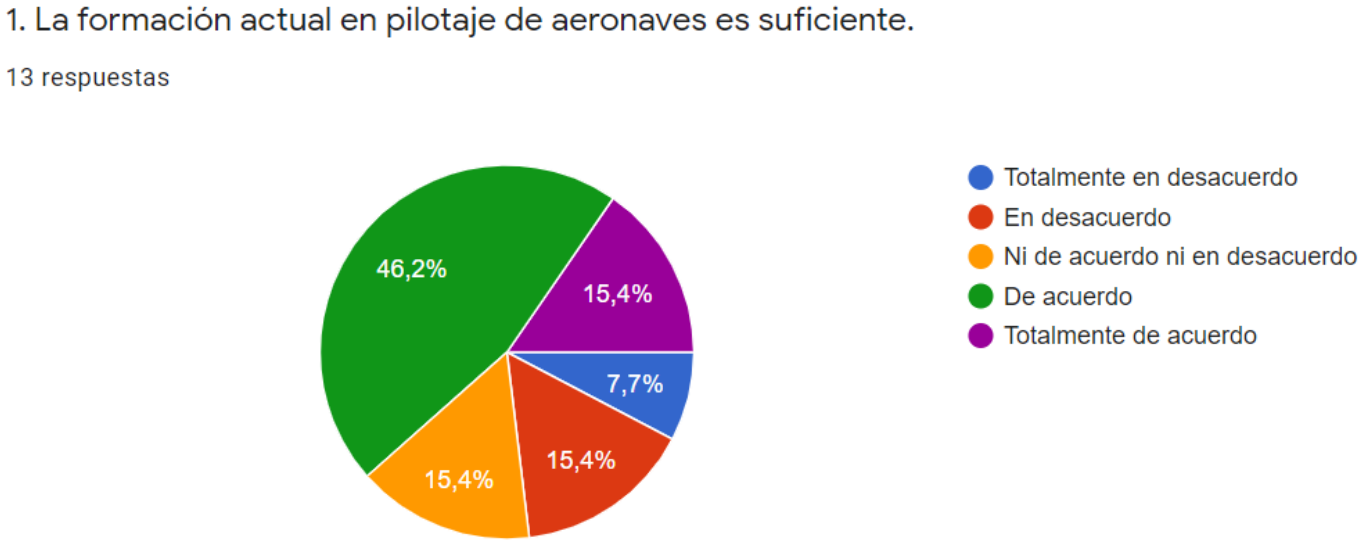


Figura 6.1: Respuesta 1º Pregunta

De esta imagen podemos observar que un mayor numero de personas se muestran de acuerdo con esta afirmación en mayor o menor medida, estando un 15,4 % totalmente de acuerdo y un 46,2 % de acuerdo. Del resto de especialistas el 15,4 % se muestra en desacuerdo y un 7,7 % totalmente en desacuerdo de lo que se puede deducir que opinan que a las formación actual de los pilotos le hace falta algo más para complementarla y por último hay un 15,4 % que no se muestra ni de acuerdo ni en desacuerdo.
De lo que deducimos que, en rasgos generales, la formación que tienen los pilotos es satisfactoria y les permite realizar adecuadamente su trabajo, aunque es posible que pueda ser mejorada en algunas áreas.

- **¿Tiene alguna propuesta para mejorar la formación en pilotaje de aeronaves?**

  Varios especialistas han indicado que la mejora de la formación pasa por los simuladores, tienen que usarse más tiempo, tienen que ser mas completos y mejores, centrados en escenarios reales y en las distintas fases, no sólo el vuelo en fase de crucero.

  Otra propuesta ha sido que "se siga un plan de entrenamiento... de acuerdo al análisis respectivo producto de seminarios y experiencias de años anteriores", de lo que se entiende que habría que analizar la respuesta de los pilotos e instructores a los seminarios y diferentes experiencias para poder mejorar año a año la formación. Una de las ideas de nuestro sistema es precisamente guardar las observaciones de los instructores de evaluación en evaluación.

  Otra respuesta se ha referido al CRM[2]. Son procesos diseñados para reducir el error e incrementar la efectividad de las tripulaciones aéreas (Wiener, Kanki, y Helmreich, 1993).

  También se ha mencionado la importancia de hacer cursos orientados a diferentes aspectos de la aviación y el pilotaje.

- **¿Tiene experiencia con el uso de simuladores de vuelo?**

  En esta pregunta todos han contestado afirmativamente, por lo tanto todos han hecho uso de un simulador de vuelo en algún momento demostrando así la importancia de estos a la hora de trabajar en el sector aeroespacial.

- **La utilización actual de simuladores de vuelo por parte de los instructores de vuelo es herramienta suficiente para la formación de los pilotos.**

[2]El CRM (Crew Resource Management), que puede definirse como la óptima utilización, por parte de una tripulación, de todos los recursos disponibles (información, equipos materiales y recursos humanos) para la consecución de operaciones de vuelo seguras y eficientes (Lauber, 1984)

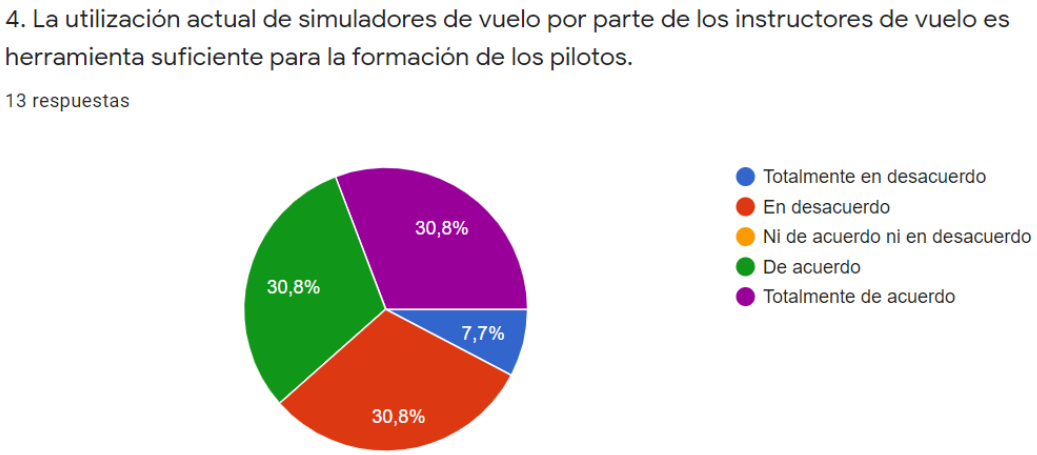


Figura 6.2: Respuesta 4º Pregunta

Las respuestas a esta pregunta están bastante divididas, aunque curiosamente no hay ningún especialista que no este ni de acuerdo ni en desacuerdo por lo que todos tienen una opinión firme al respecto. Se trata de un tema de intenso debate en la comunidad, según nos cuentan.

Un 30,8 % de los especialistas se han mostrado de acuerdo con esta afirmación, mientras que otro 30,8 % está de ellos completamente de acuerdo. De esto se deduce que creen que únicamente con entrenamiento en simuladores se pueden llegar a obtener los conocimientos y aptitudes necesarios para poder enseñar a pilotar una aeronave a un tercero, aunque hay por otro lado un 30,8 % en desacuerdo y un 7,7 % totalmente en desacuerdo y por lo tanto creen en la posibilidad de que quizás otros medios podrían reforzar más la formación de los instructores.

Estos especialistas que se muestran en desacuerdo creen realmente que, aparte de los simuladores, son necesarias otras herramientas, como métodos, seminarios o clases para poder entrenar a alguien que pretende convertirse a su vez en un instructor de vuelo.

- **La formación actual de los instructores de vuelo es suficiente para desempeñar correctamente su trabajo.**

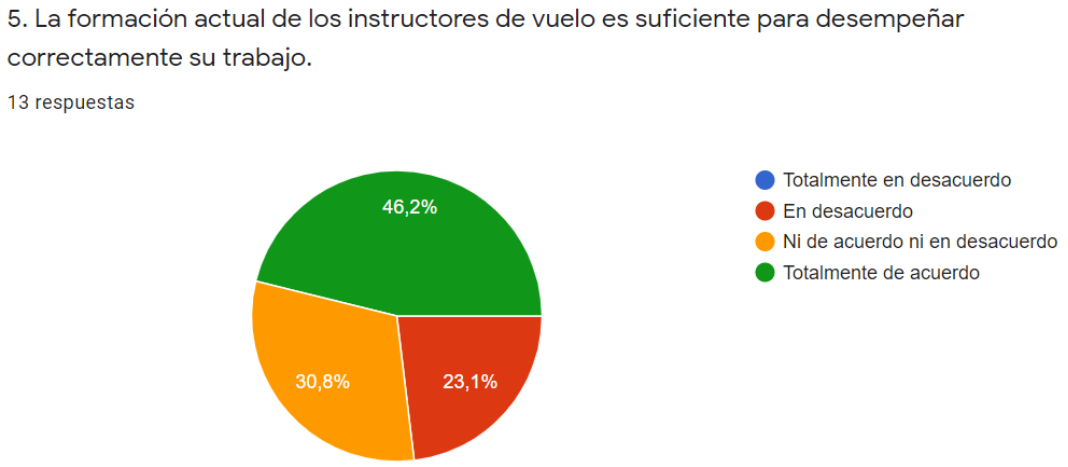


Figura 6.3: Respuesta 5º Pregunta

En este gráfico podemos observar que hay un46,2 % de los especialistas que piensan que actualmente los instructores tienen la formación adecuada para poder instruir a los pilotos.

Aquí hay un 90,8 % de los especialistas que no se muestran ni de acuerdo ni en desacuerdo, pero hay una minoría de expertos que representan al 23,1 % de los especialistas que están en desacuerdo, por lo tanto creen que los instructores necesitan más formación de la que reciben para poder desempeñar correctamente su trabajo. Este es el resquicio por donde queremos plantear nuestra aplicación como una vía de mejora.

- **¿Tiene alguna propuesta para mejorar la formación de los instructores de vuelo?**

  Para esta pregunta hay una amplia diversidad de opiniones, el incremento de la experiencia y la unificación de criterios son algunas de estas propuestas.

Respecto a la experiencia, en una de las respuestas se indica que hay una falta de experiencia que se debe a un cambio en el modelo que ahora está plenamente basado en competencias, y que es precisamente algo en lo que ponemos mucho énfasis en nuestra aplicación: que el instructor sea capaz de evaluar estas competencias en el piloto.

Otras propuestas para mejorar la formación indican que es necesario que la enseñanza esté más orientada a la práctica docente, que sea mas interactiva, dando importancia a la identificación del perfil, a la parte emocional y a las necesidades del piloto por parte del instructor.

También indican que es importante seguir con los planes de entrenamiento, así como la selección de personal capaz de enseñar y compartir sus conocimientos, por lo que ven necesario una mejora en las competencias pedagógicas de los instructores.

De estas respuestas podemos observar que hay una gran diversidad de ideas que podrían aplicarse, aunque algunas coincidan en algunos puntos esenciales de mejora. Quizás las distintas respuestas dadas pueden deberse a la diferencia del puesto ocupado o la posición en la jerarquía de trabajo, dado que en función de esta, el enfoque de la propuesta varía ya que lo observado en el trabajo también varía.

6.1.3. El simulador

Por último este bloque se enfoca en lo que mas nos interesa a nosotros, que es la evaluación de la prueba de concepto sobre el simulador propuesto, recopilando opiniones y sugerencias que tienen lo especialistas sobre este sistema.

- **¿Se entiende el concepto del simulador que presentamos?**

Figura 6.4: Respuesta 4º Pregunta

De forma general todos han entendido el concepto del simulador, excepto dos personas y una de ellas ha sugerido que cambiando el nombre a “Flight Instructor Simulator” podría clarificar el objetivo del simulador.

- **La información sobre piloto es suficiente.**

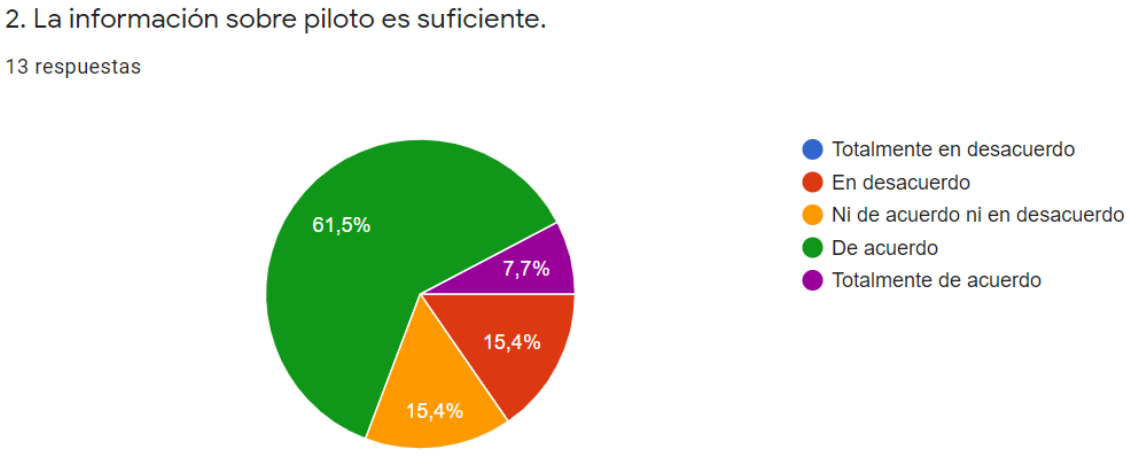


Figura 6.5: Respuesta 4º Pregunta

En lo que respecta a esta pregunta una gran mayoría de los especialistas representado por un 61,5 %, se han mostrado a de acuerdo estando un 7,7 % totalmente de acuerdo, un 15,4 % no se muestran ni de acuerdo ni en desacuerdo y un 15,4 % se muestran en desacuerdo por lo que opinan que falta información a la hora de completar el perfil de un piloto. Si es verdad que nosotros hemos usado intencionadamente un perfil muy básico, para no intimidar a los usuarios con excesivos datos personales de sus aprendices.

- **¿Qué información sobre el piloto cree oportuno añadir?**

  Varios de los especialistas han planteado que sería oportuno añadir información sobre el perfil psicológico de los pilotos. Entre las razones por las que sería recomendable añadirlas se mencionaba la compatibilidad entre el piloto y el instructor, y también para saber lo receptivo que estará el piloto en cada sesión.
  Otro información que se ha repetido en varias respuestas es la inclusión de la experiencia previa que tiene el piloto.
  Estas respuestas son las respuestas que más se han repetido por lo que hay bastante consenso al respecto y sería recomendable plantearse el incluir esta información en el perfil del piloto cuando desarrollemos la versión final del producto.

- **La información mostrada en la descripción inicial de los escenarios de entrenamiento predefinidos es suficiente.**

  A excepción de una persona que no se a mostrado ni a de acuerdo ni en desacuerdo, todos los demás se han mostrado de acuerdo estando un 7,7 % totalmente de acuerdo y un 84,6 % de acuerdo, de modo que piensan que aunque la información es suficiente quizás se podría añadir alguna información más.

Figura 6.6: Respuesta 4º Pregunta

- **¿Qué información cree oportuno añadir sobre los escenarios de entrenamiento?**

  Una de las repuestas ha indicado que el mundo de los helicópteros y de las aeronaves de ala fija es muy diferente, por lo que en base esta respuesta quizás sería recomendable dividir los escenarios en función de el tipo de aeronave a la que esta destinado, o centrarnos exclusivamente en un simulador para un único tipo de aeronave.
  Las demás respuestas han sido bastante diversas al contrario que con preguntas anteriores y algunas de las propuestas son: tráfico en superficie, mantenimiento, presión atmosférica y de temperatura, el CRM en las distintas fases de vuelo. Por lo tanto revisar todos estos puntos y plantear un incremento en la información mostrada sería recomendable para evolucionar el prototipo que tenemos hoy día.

- **La información mostrada en los eventos contenidos en los escenarios es clara**

Figura 6.7: Respuesta 4º Pregunta

De todos los especialistas un 76,9 % se han mostrado de acuerdo y un 15,4 % totalmente de acuerdo, a excepción de uno de ellos que no se a mostrado ni de acuerdo ni en desacuerdo. De estas respuestas sacamos la conclusión de que algunos añadirían alguna información más a los eventos, que aunque quizás no sea demasiado relevante podría enriquecer su descripción.

- **¿Qué información plantearía añadir a los diferente eventos contenidos en los escenarios?**

  En esta pregunta hay dos sugerencias que se han repetido bastante mostrando la importancia de esta información en los eventos y que por lo tanto sería recomendable añadir, que son el trafico aéreo y las condiciones meteorológicas, que aunque ya están añadidas en el simulador piensan que debería ampliarse la información mostrada. Esto demuestra que es algo a tener en cuenta y que se debería incluir en un futuro.

- **La pantalla de evaluación es lo suficientemente clara.**

Figura 6.8: Respuesta 4º Pregunta

Un 76,9 % esta de acuerdo con esta afirmación y un 15,4 % esta totalmente de acuerdo, aunque hay un 7,7 % que no están ni de acuerdo ni en desacuerdo, por lo que sería adecuado pensar que modificaciones podrían ayudar a que se muestre la información y el objetivo de cada parte de forma mas clara y entendible.

- **El desplazamiento entre las diferentes ventanas y menús parece intuitivo**

Figura 6.9: Respuesta 4º Pregunta

Todos han mostrado a favor, estando un 23,1 % totalmente de acuerdo y un 76,9 % de acuerdo , por lo que al navegar entre los diferentes menús está claro el objetivo de cada uno y se sabe a donde hay que ir para realizar las acciones que usuario quiere en cada momento. Este era uno de los puntos más trabajados al diseñar la interfaz, la usabilidad de la aplicación que buscábamos.

- **¿Qué errores graves en la aplicación observa, que convendría arreglar de forma prioritaria?**

Se ha indicado que el no haber considerado la experiencia previa es un error, y al haber sido indicado en respuestas anteriores, sera algo que habrá que tener en cuenta y modificar.

Otra respuesta ha indicado que la evaluación se hace entre 1 y 4 o entre 1 y 5 dependiendo de la compañía y que esto tendría que ser tenido en cuenta y explicarlo. Nuestro enfoque de valorar entre 0 y 10, que pensábamos sería intuitivo, ha resultado extraño en este colectivo.

También se ha indicado que la instrucción debe ser por fases, por lo cual la conclusión sacada es que quizás sería recomendable establecer que no solo se configuren escenarios completos. Quizás sería recomendable añadir la opción de entrenar a los pilotos en fases únicas de vuelo, de tal forma que se evalué a un piloto en fases concretas del vuelo.
Y otra respuesta ha indicado de forma muy completa que quizás sería recomendable que antes de la evaluación se mostrase la información tanto del alumno como del escenario creado, de forma que el instructor pudiese realizar cambios, digamos de ultima hora antes de empezar la evaluación. También de esta forma el instructor podría no aceptar las recomendaciones del simulador y explicar sus razones a la hora de disentir respecto al criterio del simulador. La mejor forma de implementar esto sería que antes de empezar la evaluación se crease un informe comparativo entre el piloto y el escenario creado, dando la opción de modificar algunas de las opciones.

- **¿Qué cosas cree que se podrían mejorar en la aplicación?**

  En esta pregunta los especialistas han indicado varias cosas tanto de la prueba de concepto como del simulador en si.

  Han opinado que como prueba de concepto debería aproximarse mas a la aplicación definitiva, sobre todo han indicado que debería quedar mas claro las pantallas en las que habrá animación o simulación y no dejarlo a la imaginación del que evalúa la prueba de concepto.

  También han sugerido mas opciones en el menú, para que sea mas interactivo y amigable.

Y por último han indicado la necesidad de que sen visibles las acciones del piloto en la fase de evaluación, sustituyendo la imagen estática por una simulación, de forma que el instructor pueda calificar y evaluar correctamente. Esto último es realmente importante y mejoraría sustancialmente al simulador pero se escapa del alcance de esta iteración de proyecto y recomendamos plantearlo como trabajo futuro, tal vez como un nuevo Trabajo Fin de Grado.

Con todas las respuestas analizadas podemos ver cómo se nos abre ahora un camino que podemos tomar para mejorar y definir este simulador en el futuro. También podemos determinar de las respuestas que plantear un simulador para instructores de vuelo no es algo descabellado y que, aunque obviamente es un desafío tecnológico y nunca va a sustituir la formación actual que reciben los instructores, podría ser un añadido interesante que ayude a la formación didáctica, que además es algo sobre lo que algunos expertos han contestado que faltaba en la formación de los instructores.

Capítulo 7

Conclusiones

Después de haber analizado los resultados de las encuestas realizadas al visualizar el vídeo de la prueba de concepto, las conclusiones a las que se han llegado son las siguientes:

El concepto de simulador de instructor de vuelo está claro para la mayoría de los especialistas, los cuales se muestran partidarios de la existencia de un simulador de este tipo como complemento a su formación. Por lo tanto hay cabida para un simulador orientado a instructores de vuelo, demostrando a través del criterio de estos veteranos especialistas, que el planteamiento de este proyecto tiene sentido y proyección en el sector aeroespacial.

Por otro lado, los errores o la falta de información que han indicado los diferentes expertos, en rasgos generales, estaban orientados en las mismas áreas y por lo tanto, de arreglarse o incluirse esta información en el simulador en las partes indicadas, mejoraría mucho la calidad y la funcionalidad del simulador.

Podemos decir que este simulador, o este tipo de simuladores orientados a mejorar las capacidades didácticas de los instructores a la hora de impartir conocimientos y evaluar a los pilotos, se podría convertir en un futuro en una parte importante de sus contenidos didácticos, aunque en ningún momento sustituirán los medios actuales de formación de los instructores de vuelo.

A continuación hay que analizar los objetivos que se proponían de cara a este proyecto, ¿se han cumplido? ¿en que medida? ¿qué supone su cumplimiento o incumplimiento? Para analizar todo esto debidamente revisaremos uno por uno los objetivos e intentaremos dar respuesta a estas preguntas.

Los objetivos eran los siguientes:

- Crear las clases necesarias para guardar la información imprescindible que necesitamos (Pilotos, Escenarios, Casos, Simulación). Inicialmente se planteó la posibilidad de utilizar bases de datos como *MongoDB* o *MySQL* para poder guardar la información de estas clases, pero al final fue descartado, entre otras cosas para evitar la necesidad de tener que estar conectados para acceder a ellas, por lo que se opto por utilizar diccionarios donde almacenar la información necesaria donde los valores de estos diccionarios eran las estructuras creadas para guardar la información de pilotos, escenarios, etc.
  Este objetivo por lo tanto se ha cumplido en su totalidad aunque por algunas respuestas de los expertos, sería conveniente ampliar la información contenida en algunas de estas estructuras y enriquecer el modelo.

- Crear un sistema de menús que sea lo mas claro o intuitivo, de forma que el instructor sólo tenga que centrarse en las partes importantes como la creación de los escenarios.

  Por las respuestas obtenidas en los formularios este objetivo también ha sido cubierto de forma satisfactoria, pues todos han respondido positivamente a la claridad y sencillez del diseño a la hora de desplazarnos entre los distintos menús.

- Diseñar e implementar un sistema por el que un instructor pueda seleccionar o crear escenarios para poder evaluar las capacidades de un piloto.

  En el estado actual del simulador estos dos aspectos son posibles, y el instructor es capaz de crear escenarios desde cero o seleccionar algunos ya creados. Y por supuesto es capaz de seleccionar a los pilotos que se tendrán que enfrentar a estos escenarios para ser evaluados.

- Diseñar e implementar un menú o pantalla mediante el cual un instructor pueda evaluar el desempeño de un piloto durante la simulación de un escenario previamente creado.

  Este objetivo también esta cumplido, pues el instructor puede evaluar cada una de las competencias en función de las aptitudes que observe en el piloto durante la simulación. La limitación actual más evidente que podemos encontrar es que ahora mismo no se lanza la ejecución de una simulación completa, sino que los eventos muestran una imagen estática representativa, en lugar de una animación controlada por código.

- Creación de un sistema que evalué la toma de decisiones de un instructor a la hora de seleccionar los eventos a los que se tendrá que enfrentar un piloto.

  Por último este objetivo también ha sido satisfecho pues después de evaluar al piloto, al instructor se le muestra una pantalla con su propia evaluación (que podría realizar un inspector o alguna figura de autoridad superior a la del instructor), donde puede observar los fallos cometidos y poder explicarse en caso de creer que la evaluación no es correcta en algún momento.

- La inclusión de un sistema de musica ambiental.

  Se ha incluido musica ambiental o de fondo, para hacer mas agradable la navegación entre menús, pero se podría añadir mas musica y efectos sonoros en un futuro pues en la actualidad cuenta únicamente con una pista de audio.

A continuación hay que analizar una serie de objetivos que aunque no se comentaron inicialmente se podría decir que eran objetivos secundarios del simulador.

- La creación de un sistema que permita tener un aula virtual.

Podríamos decir que este objetivo se ha completado de forma parcial, pues aunque contamos con un apartado que llamamos "aula virtual", actualmente son imágenes estáticas donde el instructor puede explicar los mandos de la aeronave con una simple referencia visual no interactiva, y por lo tanto aún esta lejos de la idea que teníamos en mente al plantear ese subsistema de aula virtual.

Para conseguir superar satisfactoriamente este objetivo tendríamos que simular una cabina en 3D y hacer que la modificación de los controles por parte del instructor o los pilotos modificase los indicadores, de forma que el instructor pudiese explicar correctamente el uso de estos indicadores y la causa y efecto entre los controles y la información mostrada. Cabría la opción también de plantearse usar realidad virtual para esta opción como ya se utiliza en muchos otros proyectos de la industria, pues de esta forma tendríamos un aprendizaje mucho más inmersivo.

- Mejorar en la medida de lo posible la estética.

  Se ha conseguido, aunque claramente hay margen de mejora. El mapa muestra aviones y helicópteros, y los modelos y texturas son correctos, aunque podría haberse montado un mapa mas parecido a un aeropuerto que a una simple pista de aterrizaje. Y respecto a los menús se han diseñado de forma que no estén excesivamente recargados y sean fáciles de ver y de entender, pero siempre pueden estar sujetos a cambios y mejoras para presentar mejor la información.

- Hacer lo mas real posible la simulación.

  Este objetivo es el único que fue descargado pronto y no se ha cumplido. Se ha optado por mostrar imágenes fijas como una representación de los eventos concretos, pero lo ideal habría sido una simulación interactiva e inmersiva al 100 por 100, o en un caso intermedio lanzar un vídeo prefijado mostrando lo que haría el piloto. Para realizar esto se tendrían que haber modelado las cabinas y personajes que representen a los pilotos, además de haberse hecho el *rigging* y el pesado de los personajes, etc.

Por lo tanto después de analizar el estado de los objetivos iniciales podemos concluir que:

Este proyecto, que empezó con la idea de desarrollar un simulador de vuelo tomando el enfoque del entrenamiento basado en escenarios (SBT), desvió ligeramente su objetivo yéndose hacia un terreno no tan explorado como es el de la idea de formar al formador y es precisamente con este enfoque que el producto podría llegar a ser relevante en el sector, convirtiéndose de esta manera en el primer simulador para instructores de vuelo basado en escenarios de entrenamiento que conocemos.

De esta forma se pone al instructor de vuelo en un entorno en el que tiene que entrenar a pilotos atendiendo a las competencias de que este carece, teniendo así que crear escenarios virtuales de simulación mediante eventos que sean más adecuados para entrenar dichas competencias, de tal forma que el piloto entrene y mejore su calificación en ellas.

Este método de enseñanza basado en competencias es algo que se lleva tiempo intentando integrar en el ámbito aeroespacial, y en el que aun no se tiene mucha experiencia, por lo que parecía importante poder tomar ya mismo esa dirección en el proyecto.

Después de crear estos escenarios, el instructor deberá evaluar al piloto ficticio que está tratando de entrenar según su comportamiento en dicho escenario y después, el instructor mismo sera evaluado por el simulador puesto que el objetivo es precisamente ese: dar realimentación al instructor.

Creo que es una buena aportación respecto al *software* destinado al entrenamiento de pilotaje de aeronaves, puesto que toma la idea de formar al formador, que no es una idea muy utilizada actualmente, y por otro lado si nos enfocamos en el ámbito aeroespacial, estamos utilizando el modelo de competencias que es un modelo emergente que se están empezando a implantar y en los que aún no se tiene mucha experiencia, por lo que es relevante poder formar a los instructores utilizando este modelo.

7.1. Trabajo Futuro

Respecto al trabajo futuro de este simulador hay que empezar a hablar sobre qué habría que corregir, qué cambios se tendrían que ir haciendo después y por último en qué se podría terminar convirtiendo todo esto en un futuro. Es decir, la evolución completa paso a paso hasta llegar al posible cenit de un simulador comercial con estas mismas ideas.

Para empezar, algo que habría que corregir, de seguir con el desarrollo, es la ventana de evaluación: la información que se le muestra al instructor, de forma que lo que vea sea el conjunto de todos los indicadores, señales y mediciones a las que el piloto está expuesto en la cabina. De esta forma puede hacerse una idea de por qué el piloto toma las decisiones que toma en cada momento y podrá evaluar mejor si lo que hace es correcto o no.

También otro posible cambio sería que el instructor pudiese escuchar las conversaciones en cabina y las conversaciones del piloto con la torre de control o con un copiloto y que se le mostrase al instructor como subtítulos. Creo que sería interesante para el instructor, pues de esta forma sería capaz de evaluar la competencia referente a la comunicación.

Y por último, para la siguiente iteración de la interfaz se tendrían que sustituir al menos las imágenes fijas por vídeos que mostrasen el comportamiento del piloto en cabina y del exterior de la aeronave.

Otro avance interesante creo que sería permitir que en cada fase de vuelo puedan ocurrir mas de un evento. Para esto habría que estudiar y controlar las incompatibilidades entre los distintos eventos y quizás esto también se podría contemplar a la hora de asignar dificultad, de tal forma que al elegir un evento se bloqueasen los eventos incompatibles en caso de una dificultad fácil.

También podemos en este caso utilizar la dificultad de tal forma que de tener una fuerte dificultad se puedan asignar cualquier evento aunque sea incompatible pero a la hora de evaluar las elecciones del instructor se le penalice por elegir eventos incompatibles.

Y creo que como evolución final, si se sigue trabajando con este simulador, sería convertirlo en una herramienta viable de interacción entre pilotos e instructores. Con esto lo que quiero decir es que habría que programar un simulador de vuelo para los pilotos y fusionar ambos simuladores. De esta forma los escenarios guardados o los configurados por el instructor se crean como una simulación real para el piloto, por lo tanto ahora las imágenes mostradas en la ventana de evaluación serían imágenes reales de la ejecución sobre como interactúa el piloto con el simulador, sin que sean imágenes fijas o vídeos como comentábamos anteriormente.

Esta parte también se podría probar inicialmente con realidad virtual para cuando se simule el vuelo del piloto, y finalmente implementar el simulador en una recreación hardware de una cabina real como es el caso de muchos de los simuladores de vuelo actuales. Con estas modificaciones el instructor calificaría a un piloto real y él también sería evaluado por el simulador.

De esta forma podríamos decir que se estarían formando a la vez tanto pilotos como instructores, o eliminando la evaluación de los instructores se estaría creando un simulador de vuelo basado en SBT donde el instructor de vuelo crearía escenarios orientados a entrenar competencias concretas del piloto.

De cara a este último escenario planteado habría que estudiar cómo implementar que la evaluación del instructor no sea sólo en función de las elecciones que hace a la hora de configurar el escenario, si no que también evalúe como el instructor califica al piloto. Para este problema lo que tendríamos que hacer es que el propio simulador sea capaz de evaluar al piloto y por lo tanto comparar al final la nota recibida por el instructor con la nota recibida por el simulador y evaluar al instructor en consecuencia.

Creo que esta última propuesta es la mas complicada de realizar pues es difícil cuantificar o evaluar los aspectos no técnicos a la hora de pilotar una aeronave. Si nos preguntasen por un proceso, una definición o por cómo se realiza una maniobra determinada, las respuestas dadas serían diferentes, y las calificaciones a estas respuestas también serían distintas pues a no ser que se especificase el proceso o la definición exacta y la respuesta fuese idéntica punto por punto, cada instructor o profesor lo calificaría de una forma diferente pues cada uno pone el énfasis en aspectos diferentes.

También se podría plantear la evaluación en función del resultado final pero sería algo no aconsejable pues si atendemos a esto, el resultado final de un vuelo sería ir del punto A al punto B y que el avión no sufra daño, pero si un instructor viese a un piloto que sin ninguna razón justificable empieza a hacer acrobacias en pleno vuelo, aunque llegase a su destino sin ningún daño no sería razonable que le diese buena nota, por esta razón no es aconsejable fijar una evaluación atendiendo sólo al resultado final.

Por lo tanto y considerando todo lo explicado anteriormente, el escenario último de este simulador podría ser convertirlo en una herramienta que pudiese servir para que un instructor crease escenarios para evaluar pilotos y a su vez se generasen estos escenarios para que pilotos reales puedan utilizarlos como simulador, de forma que el instructor pudiese ser evaluado por el simulador mejorando su forma tanto de evaluar como de crear escenarios y el piloto también mejorase las capacidades necesarias para los pilotos.

Referencias

Dapica, R., y Peinado, F. (s.f.). Evaluation of a similarity function for generation of flight simulator training scenarios using case-based reasoning.

F., W., Moroney, y Moroney, B. W. (1991). Utilizing a microcomputer based flight simulation in teaching human factors in aviation. , *35*, 523–527.

FrontierDevelopments. (2016). *Football manager.*

FrontierDevelopments. (2019). *Planet zoo.*

IATA. (2013). Evidence-based training implementation guide, 1st edition. *Montreal: International Air Transport Association*.

ICAO. (2013). Manual of evidence-based training. *Montreal: International Civil Aviation Organization*.

JUMPER, E., y BAUGHN, J. (2012). The use of microsoft flight simulator in aerospace education. , *127*, 875–883.

Kharoufah, H., Murray, J., Baxter, G., y Wild, G. (2018). A review of human factors causations in commercial air transport accidents and incidents: From to 2000–2016. *Progress in Aerospace Sciences*, *99*, 1–13.

Lauber, K., J. (1984). Resource management in the cockpit. , *53*, 20–30.

MirageGameStudios. (2019). *Little big workshop.*

Ortega, R. d. (2017). *El rol del docente en la escuela de aviación de la policía nacional* (Inf. Téc.). ESAVI. Recuperado de: http://hdl.handle.net/10654/16147.

Salas, E., Priest, H. A., Wilson, K. A., y Burke, C. S. (2006). Scenario-based training: improving military mission performance and adaptability. *Military Life: The Psychology of Serving in Peace and Combat*, *2*, 32-53.

Salas, E., Wilson, K. A., Lazzara, E. H., King, H. B., Augenstein, J. S., Robinson, D. W., y Birnbach, D. J. (2008). Simulation-based training for patient safety: 10 principles that matter. *Journal of Patient Safety*, *4*(1), 3-8.

SportsInteractive. (2019). *Football manager.*

Sánchez Ramírez, M. K. (2014). Implementación del modelo de gestión por competencias en la escuela de aviación civil colombiana aviacol ltda.

Wiener, E., Kanki, B., y Helmreich, R. (1993). Cockpit resource management.

Apéndice A

Title, abstract and keywords

Flight Simulator Instructors Education
based on Virtual Training Scenarios

Flight Simulator Instructors Education based on Virtual Training Scenarios

Alfredo Hernández Burgos

FACULTY OF COMPUTING

SOFTWARE ENGINEERING DEPARTMENT

AND ARTIFICIAL INTELLIGENCE

COMPLUTENSE UNIVERSITY OF MADRID

Final Degree Project in Computer Engineering

Madrid, June 26, 2020

Director: Prof. Dr. Federico Peinado Gil

Title

Development of a Control System for Unreal Engine Bots based on Utility Theory

Abstract

Despite all the technological improvements that exist in the aeronautical sector, the majority of accidents are due to human factors. cite kharoufah2018review. That is why in recent years, the International Civil Aviation Organization footnote url https://www.icao.int (ICAO) It has been promoting a paradigm that integrates all kinds of skills in crew performance, both technical and non-technical. It is sought that the training in each session is specific to the training needs of each pilot, all based on the data obtained in both real operations and training. This alternative paradigm is called Evidence-Based Training (EBT) cite international2013manual.

A fundamental part of training scenarios is the use of a flight simulator. In these simulators the pilot is subjected to situations specifically designed to test his skills. The design of these scenarios usually corresponds to the instructor, who often have little time to prepare new material and reuse the old one, and despite being such a critical element in the training of aircraft piloting, the truth is that this work of the instructors is not properly regulated or monitored.

Our proposal is to study the possibility of creating an educational computer application for flight instructors, so that they can face situations similar to those that occur in the real world but without risks or real costs, so that they can improve their skills in a controlled environment. This application would be focused on the ambit of aviation, and could be used in teaching or training in the aerospace sector to improve the knowledge or skills of instructors or pilot instructor candidates.

As a result of this study we propose the design of this "flight instructor simulator", so that users can put themselves in the role of an instructor and can select a fictitious pilot to train them in the skills that need to improve this , using a training scenario that you can also select and even configure. Subsequently, said user must evaluate the performance of the pilot in the proposed scenario and end up receiving a report on his work as an instructor and pilot evaluator.

To carry out this project, different flight simulators were investigated, to know what experience they offered and what information they showed, because although it was not about developing a flight simulator, the proposed system was similar and was based on the same knowledge (events that can occur during flight, competitions of pilots who must be trained and are subject to evaluation, etc.).

The first thing that was done next was to make sketches and diagrams on paper, to immediately work with Balsamiq on the first models whose ease of use could be tested. Once the concepts were clear and there were no big unknowns, the next step was to start working directly on a functional prototype in Unreal Engine where features were added that were not initially proposed.

This iterative improvement process continued until we reached the final design that we wanted to test to verify the usefulness of the application and consider possible modifications, polishing details according to the criteria of experts in the aerospace sector.

Keywords

Flight simulator, Aeronautics, Aviation, Learning, Virtual Classroom, Accidents, Incidents, Events, Scenarios, Instructors, Competencies, Evaluation, Pilot, Plane, Helicopter

Apéndice B

Introduction

B.1. Memory structure

The chapters of which this work is composed are the following:

1. In Chapter 1: Introduction. The initial approach of the project, the reasons for choosing a project that is supported in the aerospace ambit, the reasons for deciding to make a simulator and how the initial focus of the simulator ended up being modified giving rise to a different type of simulator.

2. In chapter 2: State of the art. The history of aviation will be discussed to locate the ambit in which we are going to move throughout the project, ending the history of flight simulators to further specify the scope. Then some of the types of software intended for education will be discussed, because in the end the project is a simulator that continues to be a type of software intended for education. And some of the existing simulators will be discussed, as well as the aerospace knowledge on training on which the project is based.

3. In chapter 3: Objectives and specification. In this chapter the initial objectives of the project and its evolution during development will be mentioned, and finally they will define and explain what the objectives of this project were. On the other hand, the software requirements of this application will be mentioned and explained, both functional and non-functional.

4. In chapter 4: Methodology and tools. The reasons that gave rise to the choice of the methodology used for this project will be explained, as well as the tools used for the development of the project and how they were used in it.

5. In chapter 5: Analysis, design and implementation. We will go through all the classes, structures and menus created, explaining their function, the variables or ambits they contain, and the functions created inside, explaining what they are used for.

6. In chapter 6: Tests and results. The reasons for conducting a proof of concept will be explained and we will see and analyze the specialists' responses to the form they filled out after watching the proof of concept video.

7. In chapter 7: Conclusions. After analyzing the responses obtained, in this chapter we will see the conclusions we reached and explain the possible future of this project, how it could be developed in the future.

B.2. Initial approach

For the purpose of this project, it was thought that in many cases these projects are focused solely on the ambit of computing, each focused on a specific area or ambit within it, but only taking computing into account. For this reason, to carry out this project, it was thought that it would be a recommendable change to rely on a different ambit from that of computer science, so that using the knowledge acquired about computer science throughout the course, one could help in a different ambit.

Thinking that it would be advisable to select a ambit that already had some relationship with computing, since trying to join two ambits that have nothing to do at least so far can be a difficult task to face and to which you must dedicate time that you do not was available.

So after thinking that another knowledge discipline could support this project, it was decided that it would be a good idea to use aeronautics. Computer science and aeronautics are already quite related, both due to the existence of the systems mounted within the different aircraft, and the development of flight simulators, and it was precisely on simulators that this project would finally focus.

Once this was decided, it began to think what type of simulator would be developed since there are already many simulators on the market, and taking as reference the article *Evaluation of a Similarity Function for Generation of Flight Simulator Training Scenarios using Case-Based Reasoning* (Dapica y Peinado, s.f.), we thought it was a good idea to propose a flight simulator based on SBT [1].

Even though this approach was clear, as the project progressed, it was modified and evolved. One of the biggest changes he underwent was the shift in focus from being a flight simulator to being a simulator for flight instructors.

The reasons for this were, first, that due to the number of existing simulators, it was difficult to do something new or to delve into any specific aspect of the simulators, thus improving the flight simulators that currently exist.

On the other hand, there were no simulators designed for flight instructors, therefore it was a facet of the simulators that could be exploited without competition.

And another reason was that it would be a good idea to be able to help flight instructors to improve their abilities to instruct and evaluate pilots, because apart from being able to recycle their knowledge through courses and certifications, a flight instructor can only use flight simulators, but in this way you can only train your flight skills as a pilot, but not the didactic skills to impart or assess pilots' knowledge.

[1] Scenario-based training

Facing the evaluation that the instructor would make to the pilot, currently a model is being established for both learning (Ortega, 2017) and management (Sánchez Ramírez, 2014) based on competencies, both in the aerospace ambit and in other areas, therefore It was thought that it would be good to take this idea for the project, in such a way that the instructor designed the scenarios based on the competencies that the pilot would need to train.

Once it was proposed that it would be a simulator for flight instructors, and that it would take the focus of the competitions to evaluate the pilot, it did not undergo very important changes, but some functionalities were added and the design of the different menus was gradually arriving.

Although some characteristics of the project were modified, the basis was always the training based on scenarios, in which the instructor would eventually modify the different events that make up the scenario, to adapt it to the needs of the pilot. In this way, depending on the competencies that the pilot has and needs to train, the instructor should choose the events accordingly to form a suitable scenario.

B.3. Subjects

The subjects that were most useful for carrying out this specific project were Programming Foundations, Data and Information Structures, the Unreal Engine course and Internships in companies, below it will be indicated in which aspects these subjects were relevant.

- Programming basics: Despite using the *blueprint* and nodes provided by Unreal Engine, the way and the rules when programming are the same as if we program with code as we are taught in the different programming subjects and specifically on programming fundamentals.

- Data and information structure: This course has been especially useful given the number of structures that must be created to be able to represent the information of pilots, events, scenarios ..., as well as identifying which classes were most useful to represent this information . Another aspect that has been very useful in this subject is the ability to see the consumption of space according to the structures and ways of transmitting this information in more efficient ways.

- Unreal Engine Course: Although it is not a subject, it was knowledge taught at the faculty and it allowed having a knowledge base to use this graphic engine.

- Practices in companies: In my specific case I was working with Unreal Engine and one of the jobs I spend more time working on is creating interfaces for video games, so I learned how to handle the *widget* in Unreal Engine

Also, although I did not end up being relevant, the subject of Databases, Networks and Expansion of operating systems and networks were useful at the time when the creation of a database that had to be connected to Unreal Engine was considered, but it was finally abandoned This idea, given that it did not seem correct that it was necessary to connect to the network to be able to access the data that was required, so finally another approach was taken.

Apéndice C

Conclusions

After having analyzed the results of the surveys carried out when viewing the video of the proof of concept, the conclusions that have been reached are the following:

The concept of simulator a flight instructor simulator is clear to most specialists, and they have argued for the existence of such a simulator. Therefore, there is room for a simulator oriented to flight instructors, demonstrating that the realization of this project makes sense and that time has not been wasted.

On the other hand, the errors or the lack of information that the different experts have indicated, in general features were oriented in the same areas and therefore, to fix or include this information in the simulator in the parts indicated by the specialists themselves, It would greatly improve the quality and functionality of the simulator.

We can say that this simulator, or this type of simulator aimed at improving the didactic capacities of instructors when it comes to imparting knowledge and evaluating pilots, could become an important part of their didactic content in the future, although at no time will replace the current means of training flight instructors,

Next, we must analyze the objectives that were proposed for the completion of this simulator. Have they been met? what extent? What does its fulfillment or non-compliance mean? To analyze it correctly, we will review the objectives and try to answer these questions.

The objectives were as follows:

- Create the necessary classes to store the essential information we need (Pilots, Scenarios, Cases, Simulation).

  Initially, the possibility of using databases such as *MongoDB* or *MySQL* to store the information of these classes was considered, but in the end it was discarded, among other things to avoid the need to have to be connected to access them, so I opted to use dictionaries where to store the necessary information where the values of these dictionaries were the structures created to store information about pilots, scenarios ... This objective has therefore been fully met, although it would be advisable to expand the information contained in some of these structures by some responses from the experts.

- Create a menu system that is as clear or intuitive as possible, so that the instructor only has to focus on the important parts such as creating the scenarios.

  For the answers obtained in the forms, this objective has also been satisfactorily met, since all have responded positively to clarity when moving between the different menus.

- Design and implement a system by which an instructor can select or create scenarios to assess the capabilities of a pilot.

  In the current state of the simulator these two aspects are possible, and the instructor is able to create scenarios from scratch or select some already created. And of course it is capable of selecting the pilots that will have to face these scenarios.

- Design and implement a menu or screen through which an instructor can evaluate the performance of a pilot during the simulation of a previously created scenario.

  This objective is also fulfilled, since the instructor can evaluate each of the competences based on the skills that the instructor observes in the pilot during the simulation. The only flaw we could find is that it is not really a simulation as such as the events show a static image instead of a simulation.

- Creation of a system that evaluates the decision-making of an instructor when selecting the events that a pilot will have to face.

  Finally, this objective has also been met because after evaluating the pilot, the instructor is shown a screen with his own evaluation, where he can observe the mistakes made and be able to explain himself if he believes that the evaluation is not correct at any time. .

- The inclusion of an environmental music system.

  Background music or background music has been included to make navigation between menus more pleasant, but more music could be added in the future, as it currently only has an audio track.

Next, it is necessary to analyze a series of objectives that, although they were not initially commented, could be said to be secondary objectives of the simulated ones.

- The creation of a system that allows having a virtual classroom.

  We could say that this objective has been partially completed, because although we have a section that we call virtual classroom, it is a static image where the instructor can explain the controls of the aircraft with a visual reference in front and therefore it is still far from the image we had in mind of a virtual classroom.
  In order to successfully overcome this objective, we would have to simulate a 3D cockpit and have the controls modified by the instructor or the pilots to modify the indicators, so that the instructor could correctly explain the use of these indicators and the cause and effect. between the controls and the information displayed. It would also be an option to consider using virtual reality for this option as it already exists in the industry, this way it would be much more immersive.

- Improve aesthetics as much as possible.

It has been achieved, although it can be improved. The map shows planes and helicopters, and the models and textures are correct, although a map more similar to an airport than a simple runway could have been assembled. And regarding the menus, they have been designed so that they are not excessively overloaded and are easy to see and understand, but they can always be subject to changes and improvements.

- Make the simulation as real as possible.

  This objective has not been met, still images have been put in as a representation of the specific events, but the ideal would have been a real simulation or, in an intermediate case, a video showing what the pilot would do. To do this, the cabins and characters representing the pilots would have to have been modeled, and the *rigging* and weighing would also have to be done.

Therefore, after analyzing the state of the initial objectives, we can conclude knowing that:

This project, which started with the idea of developing a flight simulator taking the scenario-based training (SBT) approach, slightly changed its objective moving towards a not so explored terrain that is the idea of training the trainer and it is with this approach that This project could become relevant, thus becoming a simulator for flight instructors based on training scenarios.

In this way, the flight instructor is placed in an environment in which he has to train pilots attending to the competitions that he lacks, thus having to create virtual simulation scenarios through events that are better for training these competitions, in such a way that the pilot trains and improves in them.

This method based on competencies is something that is beginning to be integrated in the aerospace ambit and in which there is not much experience yet, so it seemed important to be able to take that direction in the project.

After creating these scenarios, the instructor must evaluate the pilot according to their behavior in said scenario and then this will be evaluated by the simulator since the idea is to train the instructor, not the pilot.

I think it is a good contribution regarding the *software* intended for education, since it takes the idea of training the trainer, which is not a widely used idea at the moment and, on the other hand, if we focus on the aerospace ambit, we are using the Competency model that is an emerging model that is beginning to be implemented and in which there is not much experience yet, so it is relevant to train instructors using this model.

C.1. Future Work

Regarding the future work of this simulator, it is necessary to start talking about what would have to be corrected, what changes would have to be made afterwards and finally, what could end up becoming a future. That is to say, the complete evolution step by step until reaching the possible zenith of a simulator of this type.

To start, what should be corrected to continue with the development is in the evaluation window. The information shown to the instructor, so that what he sees is all the indicators, signals and measurements to which the pilot is exposed in the cockpit. In this way you can get an idea of why the pilot makes the decisions he makes at all times and be able to better assess whether what he does is correct or not.

Also another possible change would be that the instructor could listen to the conversations in the cockpit and the conversations of the pilot with the control tower or with a co-pilot and be shown to the instructor as subtitles, I think it would be interesting for the instructor when it comes to being able to qualify the competition referring to the communication qualification.

And finally, for this window, at least for this initial evolution, the still images would have to be replaced by videos showing the behavior of the pilot in the cockpit and outside the aircraft.

Another interesting advance I think would be to allow more than one event to occur in each flight phase, for this it would be necessary to study and control the incompatibilities between the different events and perhaps this could also be considered when assigning difficulty, in such a way that when choosing an event the incompatible ones were blocked in case of an easy difficulty, but in the case of having a strong difficulty, any event can be assigned even if it is incompatible but when evaluating the instructor's choices, you will be penalized for choosing incompatible events .

And I think that as a final evolution, that is to say that the objective if we continue working with this simulator would be to turn it into a viable tool for interaction between pilots and instructors.

With this what I mean is that you would have to program a flight simulator for the pilots and merge both simulators. In this way the saved scenarios or those configured by the instructor are created as a real simulation for the pilot, therefore now the images shown in the evaluation window would be real images of how the pilot actually interacts with the simulator, without being still images or videos as previously mentioned.

For this part, it could also be initially tested with virtual reality for the pilot when simulating his flight and finally implement the simulator in a recreation in a real cockpit, as is the case with some of the current flight simulators. With these modifications the instructor would qualify a real pilot and he would also be evaluated by the simulator.

In this way we could say that both pilots and instructors were being trained at the same time, or by eliminating the evaluation of the instructors, a flight simulator based on SBT would be created where the flight instructor would create scenarios aimed at training specific pilot competencies.

Facing this last scenario, it would be necessary to study how to implement that the instructor's evaluation is not only based on the choices he makes when configuring the scenario, but also how the instructor qualifies the pilot. For this problem, what we would have to do is for the simulator itself to be able to evaluate the pilot and therefore compare at the end the note received by the instructor with the note received by the simulator and evaluate the instructor accordingly.

I think that this last proposal is the most complicated to make because I think it is difficult to quantify or evaluate some of the aspects when piloting an aircraft. We can see this in any ambit, especially in non-scientists, because although 2 + 2 is equal to 4 and any result different from this we can conclude that it is an erroneous result.

If we were asked about a process, a definition or how a maneuver is performed, the answers given would be different, and the ratings for these answers would also be different, unless the process or the exact definition were specified and the answer was the same point. per point, each instructor or professor would rate it in a different way because each one puts the emphasis on different aspects.

The evaluation could also be considered based on the final result but it would be somewhat inadvisable because if we attend to this, the final result of a flight would be going from point A to point B and that the plane does not suffer damage, but if an instructor saw A pilot who without any justifiable reason begins to drill and curl, even if he arrives at his destination without any damage, it would not be reasonable to give him a good grade, for this reason it is not advisable to set an evaluation based only on the final result.

Therefore, and taking into account everything explained above, the ultimate scenario of this simulator could be to turn it into a tool that could be used by an instructor to create scenarios to evaluate pilots and, in turn, generate these scenarios so that real pilots can use them as a simulator. , so that the instructor could be evaluated by the simulator, improving his way of evaluating and creating scenarios, and the pilot also improved the necessary skills for the pilots.