UNIVERSIDAD DE EXTREMADURA
Escuela Politécnica
Grado en Ingeniería Informática en Ingeniería del Software

Trabajo Fin de Grado
Aplicación de DCU en el desarrollo de interfaces con nuevos paradigmas

Autor: Alejandro Franco Silva
Fdo.:

Director/es: Julia González Rodríguez
Fdo.:

Tribunal Calificador

Presidente: José María Conejero Manzano
Fdo.:

Secretario: María Ángeles Mariscal Araújo
Fdo.:

Vocal: Pedro José Clemente Martín
Fdo.:

CALIFICACIÓN:
FECHA:

Contenido

Resumen 4
Introducción al problema 6
Planificación inicial 7
Realidad Aumentada 8
Hitos 8
Ámbitos 9
El futuro de la Realidad Aumentada 11
Gafas de Realidad Aumentada - Google Glass 11
Android 11
Comparativa de herramientas actuales de desarrollo en AR 12
Tabla comparativa de las herramientas 15
Conclusiones tablas 20
Evolución tabla comparativa 20
Principales problemas del desarrollo de la tabla 22
Diseño centrado en el usuario 23
Introducción a la metodología de desarrollo 23
Propuesta de aplicación: Aplicación de apoyo al estudio 26
Definición de usuarios de la aplicación 26
Usuario primario: Álvaro Rodríguez 26
Escenarios de uso diario: 27
Usuario secundario: Ana María Garrido 27
Escenarios de uso diario: 28
Escenarios de uso necesario: 29
Escenarios de casos límites 29
Desarrollo de prototipo 29
Evolución del prototipo 31
Primera versión 31
Segunda versión 32
Tercera versión 33
Cuarta versión 35
Quinta versión 36
Tareas a realizar en las entrevistas 38
Resultados de entrevistas para validar el diseño del prototipo 39
Primera versión 40
Segunda versión 41

Tercera versión 43
Cuarta versión 44
Propuesta de aplicación: Tarjetas de visita 44
Definición de usuarios de la aplicación 45
Usuario primario: Juan Rodríguez 45
Escenario de uso diario: 46
Escenario de uso necesario: 46
Escenarios de casos límites 47
Usuario secundario: Aseret Ibáñez 47
Escenario de uso diario: 48
Usuario secundario: Enrique Meléndez 49
Escenario de uso diario: 49
Desarrollo de prototipo 50
Evolución del prototipo 50
Primera versión 50
Segunda versión 52
Tercera versión 53
Tareas a realizar en las entrevistas 54
Resultados de entrevistas para validar el diseño del prototipo 56
Primera versión 57
Segunda versión 57
Tercera versión 57
Desarrollo de las aplicaciones y comparativa de los SDK seleccionados 58
Desarrollo primera aplicación – Aplicación de apoyo al estudio 58
Preguntas previas 58
Desarrollo en Unity 3D con Vuforia 58
Múltiples Targets 71
Animaciones en Unity 73
Principales problemas encontrados 75
Desarrollo con Metaio SDK y Creator 76
Principales problemas encontrados durante el desarrollo 82
Metaio SDK en Unity 3D 82
Desarrollo con ARMedia SDK y Unity 82
Resumen del desarrollo de la propuesta 82
Resumen de Unity 82
Resumen de Metaio 83

Desarrollo segunda aplicación – Tarjetas de visita 84
Preguntas previas 84
Desarrollo en Unity 3D con Vuforia 85
Animación de las imágenes 87
Principales problemas encontrados 91
Desarrollo con Metaio SDK 91
Principales problemas encontrados 92
Resumen del desarrollo de la propuesta 92
Resumen de Unity 92
Resumen de Metaio 93
Desarrollo de las aplicaciones en Android Studio 93
Proyectos exportados desde Unity 3D 93
Clase MainActivityNavigationDrawerMaterial 96
Clase DialogActivity 97
Principales problemas encontrados 99
Proyectos exportados con Metaio Creator 100
Principales problemas encontrados 106
Resumen del desarrollo en Android Studio 107
Comparativa SDK durante el desarrollo 108
Planificación final 112
Proyectos relacionados en desarrollo 114
Proyecto con metodología DCU: Guía de Olivenza 114
Proyecto de Realidad Aumentada: Monumentos Olivenza 115
Conclusiones 117
Bibliografía 119
Índice de ilustraciones 125
Anexo 1 – Modelos y Animaciones 129
Creación de modelos con Blender 129
Creación de modelos con Unity 3D 131
Creación de modelos con SketchUp 133
Creación de animaciones (2D) con GIMP 137

Resumen

En el presente trabajo, desarrollaremos la evolución de la Realidad Aumentada (RA o AR como abreviatura en inglés de “Augmented Reality”), los diferentes ámbitos en los que se aplican actualmente y en los que podrían aplicarse por cubrir una necesidad y de qué ejemplos encontramos actualmente de ella. Dentro del ámbito del desarrollo de aplicaciones, aprovecharemos el estudio anterior para realizar una comparativa de las características más importantes de las herramientas y sobre todo de las que más puedan interesar a los desarrolladores para seleccionar un “SDK” (Software Development Kit, es decir Kit de Desarrollo de Software) u otro.

Para continuar se realizarán diferentes propuestas de desarrollo de una aplicación formal con varios de los SDKs para ver si realmente las características encontradas en la investigación son correctas y si realmente podrían interesar tanto al desarrollador como al usuario final. Además, para ello se utilizará un desarrollo centrado en el usuario, que consiste en conocer el perfil de aquellos que utilizarán la aplicación y a posibles usuarios secundarios y en qué contextos o escenarios lo harían. A todos ellos se les realizarán entrevistas en las que se averiguará cómo creerían que debería ser el diseño en base a varios prototipos que irán evolucionando conforme avancen las entrevistas hasta encontrar el prototipo final que el usuario principal considere mejor y con el que se encuentre más cómodo, y en el que los usuarios secundarios se encuentren cómodos también, vean que es intuitivo y fácil de utilizar.

La primera propuesta que se realizó sobre el trabajo fue la investigación de la existencia de un emulador de Google Glass sobre el que poder trabajar y desarrollar algún tipo de aplicación que estuviera relacionada con el ámbito de la universidad, contando también con el diseño centrado en el usuario. Al final la propuesta tuvo que ser descartada por la imposibilidad de encontrar un SDK o librería estable, lo mejor que se encontró fueron las aplicaciones propias de las Google Glass para instalar en un dispositivo Android. Pero ni eran las versiones finales ni eran en absoluto estables, por lo que fue desechada la idea.

La segunda propuesta del trabajo y la planificación inicial de esta es tener la investigación de las herramientas, usuarios, escenarios y los prototipos en Pencil Project. A partir de ahí se utilizarían los diferentes SDKs que se hayan considerado en la primera parte para el desarrollo de la aplicación basándonos en el Diseño Centrado en el Usuario (DCU) como metodología de trabajo y desarrollando la aplicación para el sistema Android.

El proyecto terminará con los prototipos de ambas propuestas desarrollados con varias herramientas, la comparación de ambas durante el desarrollo y propuestas sobre cómo podríamos mejorarlas de cara a una aplicación final.

Introducción al problema

Durante los últimos años, sobre todo con el auge de los dispositivos móviles y la gran potencia que empiezan a tener, nos podemos encontrar con un aumento significativo de aplicaciones que utilizan el paradigma de RA sobre todo en el ámbito de la publicidad donde podemos encontrar ejemplos como el de "McDonald's"[1] o juegos desarrollados para un público juvenil o infantil como "Invizimals" de la plataforma "PSP". Además, están empezando a aparecer algunas aplicaciones relacionadas con el ámbito universitario que son interesantes, como en la "Universidad de Sevilla"[2].

El desarrollo de este tipo de aplicaciones está creciendo notablemente gracias a que las grandes empresas se han empezado a interesar recientemente y porque el ciudadano de a pie tiene cada vez acceso a más dispositivos portátiles potentes y a menor precio como hemos comentado anteriormente. Algunas empresas como Google o Microsoft están invirtiendo en sus propios dispositivos con sorprendentes avances, "SDKs" disponibles gratuitamente para los desarrolladores junto con dispositivos de prueba y precios cada vez más competitivos para poder llegar al gran público.

Todo ello nos lleva al estudio y comparación de las diferentes herramientas que encontraremos en la web para desarrollar una aplicación de este tipo y comprobar qué nos ofrece y los problemas que puedan surgir durante el desarrollo de esta.

[1] https://www.youtube.com/watch?v=0l14awnnyqg - Último acceso: 18/02/2015
[2] http://realidadaumentada.us.es/ - Último acceso: 24/08/2015

Planificación inicial

Como primera planificación, creemos que podemos tener preparada la historia y la tabla comparativa de las herramientas de RA para principios del mes de abril, en el cuál procederemos al desarrollo de varios prototipos de aplicaciones para el sistema operativo móvil Android y con las evaluaciones a los usuarios usando la metodología de Diseño Centrado en el Usuario (DCU) y estar listo para realizar la entrega y presentación del trabajo durante el mes de junio.

Esta información la podemos ver más clara en la siguiente tabla:

Actividad	Inicio esperado	Duración esperada
Historia AR	18 de febrero de 2015	3 días
Comparativa SDK AR	20 de enero de 2015	30 días
Diseño Centrado en el Usuario y propuestas de aplicación	11 de marzo de 2015	20 días
Desarrollo de los prototipos sin Android Studio	1 abril de 2015	50 días
Desarrollo de los prototipos en Android Studio	20 de mayo de 2015	7 días

Tabla 1 - Planificación inicial

Realidad Aumentada

La realidad aumentada es el término que se usa para definir una visión de un dispositivo tecnológico cuyos elementos se combinan con elementos virtuales para la creación de una realidad mixta en tiempo real; en otras palabras, es una tecnología que consiste en nutrir de información digital el mundo físico que conocemos. La diferencia principal con la realidad virtual es que no sustituye la realidad física, sino que añade una capa de datos informáticos sobre el mundo real.

Con la ayuda de la tecnología actual la información sobre el mundo real alrededor del usuario se convierte en interactiva y digital. La información artificial que se muestra sobre el medio ambiente y los objetos que se van observando puede ser almacenada y recuperada posteriormente para mostrarla.

Una de las definiciones más conocidas de Realidad Aumentada fue dada por Ronald Azuma en 1997: "la realidad aumentada combina elementos reales y virtuales, es interactiva en tiempo real y está registrada en 3D"[3]. También es la incorporación de datos e información digital en un entorno real mediante reconocimiento de patrones desarrollados por software.

Hitos

La historia de la AR es muy amplia, hasta llegar al estado tecnológico actual de potencia en las librerías AR, han sido necesarios numerosos avances tecnológicos a lo largo de los años. De todos los grandes momentos que tuvieron lugar en la historia de la AR, podemos destacar los siguientes ordenados según el año en el que tuvieron lugar:

- 1962: Morton Heilig, director de fotografía, crea un simulador de motos conocido por "Sensorama".
- 1968: Ivan Sutherland crea el primer sistema de realidad aumentada[4] que también es primero de realidad virtual. Utilizó una pantalla transparente montada sobre la cabeza a la que seguía dos rastreadores, uno mecánico y otro de ultrasonidos. El dispositivo mostraba gráficos de tipo alambre en tiempo real por la capacidad de la tecnología de la época.
- 1985: Nacimiento de Hurto. Myron Krueger crea un "videoplace" que permite a los usuarios interactuar con objetos virtuales por primera vez.

[3] http://www.cs.unc.edu/~azuma/ARpresence.pdf (pág. 2) - Último acceso: 24/08/2015
[4] http://dl.acm.org/citation.cfm?id=1476686 – Último acceso: 18/02/2015

- 1992: Tom Caudell crea el término realidad aumentada[5] para referirse a la superposición de material informático-presentado sobre el mundo real.
- 1994: Steven Feiner, Blair MacIntyre y Doree Seligmann realizan la primera utilización importante de un sistema de Realidad Aumentada en un prototipo, KARMA; presentado en una conferencia de la interfaz gráfica.
- 1996: Jun Rekimoto presenta los marcadores de matriz 2D[6], conocidos también como códigos de barra de forma cuadrada, que constituyeron uno de los primeros sistemas de marcadores para permitir el seguimiento de la cámara.
- 1999: Hirokazu Kato desarrolla ARToolKit en el HitLab y se presenta en SIGGRAPH de ese mismo año.
- 2000: Bruce H. Thomas desarrolla ARQuake[7] el primer juego al aire libre con dispositivos de realidad aumentada.
- 2008: AR Wikitude sale a la venta con el teléfono Android G1.
- 2009: ARToolkit es portado a Adobe Flash y la realidad aumentada llega a los navegadores Web.
- 2012: Google se lanza al diseño de unas gafas que crearían la primera realidad aumentada comercializada. Su nombre es Project Glass[8].
- 2015: Microsoft presenta unas potentes gafas de AR que hace efectos de hologramas, de donde proviene su nombre; Microsoft HoloLens[9].

Como podemos observar, la mayoría de estos avances tuvieron lugar en la década de los 90, seguida de un período con menos avances en este campo que se extendió hasta la primera década del siglo XXI, cuando gracias a las tecnologías móviles, dispositivos más portables y la aparición de las grandes empresas de tecnología, se encuentran en pleno auge.

Ámbitos

El impacto en la sociedad de la AR abarca desde el mundo de los tatuajes hasta aplicaciones para la lucha en contra del Alzheimer. Buscando un poco de información nos podemos encontrar proyectos de medicina, videojuegos, arquitectura e interiorismo, arte, idiomas, educación, publicidad y un gran etcétera. Muchas de las aplicaciones de

[5] http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=183317&isnumber=4717&punumber=378&k2dockey=183317@ieeecnfs – Último acceso: 18/02/2015
[6] http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=704151&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D704151 – Último acceso: 18/02/2015
[7] http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.19.6863 – Último acceso: 18/02/2015
[8] https://www.google.com/glass/start/ - Último acceso: 18/02/2015 – Actualmente cerrada.
[9] http://www.microsoft.com/microsoft-hololens/en-us – Último acceso: 18/02/2015

AR se usan en museos, exhibiciones o parques temáticos por el coste que estas tienen aún.

Como proyectos educativos, sabemos que en los últimos años la Realidad Aumentada está consiguiendo un protagonismo cada vez más importante en diversas áreas de conocimiento mostrando su versatilidad y las posibilidades que presenta. Estos proyectos abarcan desde un modelo de un cráneo con Realidad Aumentada[10] hasta aprender a soldar sin riesgos y sin costes de material[11].

En el campo de la cirugía, el cirujano que esté operando en el momento tendrá la posibilidad de acceder a datos del paciente en tiempo real o compartir los datos con otros cirujanos para compartir diferentes puntos de vista[12,13].

Cada vez es más habitual ver aplicaciones para dispositivos móviles en el que se haga uso de esta tecnología, pero quizás uno de los ejemplos más conocidos sea el del juego "Invizimals" de la plataforma PSP.

En la navegación mediante GPS también hay algunos ejemplos de este tipo de aplicaciones ya que el uso de la realidad aumentada puede ayudar mucho a la hora de encontrar un lugar mediante indicaciones en la pantalla del dispositivo del camino que debes seguir.

En el campo de la publicidad también tiene mucho por dónde crecer. Si una valla publicitaria que aparentemente te anuncia un coche o unas zapatillas te permitiera, además, ver un modelo en 3D u otro anuncio en formato audiovisual con solo enfocar tu dispositivo sobre ella, captaría muchísimo más la atención del público.

En la prensa escrita o en revistas de productos como puede ser la cadena "IKEA" ya podemos encontrar aplicaciones de RA. Es un campo digno de tener en cuenta por la capacidad que otorga de obtener información más actualizada o ampliar una noticia. Si ves por ejemplo una fotografía y al enfocarla se reprodujera un vídeo sobre esa noticia, o en la prensa deportiva pudieras ver las mejores jugadas de un evento, podríamos considerarlo, sin lugar a dudas, un enriquecimiento de la información que este medio informativo puede transmitir.

Con la RA también podemos darle nuevas funciones a elementos que se han quedado ya algo desfasados, como pueden ser las tarjetas de visitas, ya que puede ser muy

---

[10] http://mahei.es/iskull.php?os=ios&lang=es - Último acceso: 20/01/2015

[11] https://www.youtube.com/watch?v=TvgrG wxY – Último acceso: 20/01/2015

[12] http://www.xataka.com/accesorios/el-futuro-de-la-cirugia-segun-philips-y-accenture-pasa-por-google-glass – Último acceso: 20/01/2015

[13] http://www.lavanguardia.com/salud/20130621/54376177193/operacion-google-glass.html – Último acceso: 20/01/2015

interesante que a la hora de darle una a alguien, este pueda, enfocándola con la cámara de su teléfono o tablet tener acceso a tu número de teléfono, a un enlace a tu perfil de LinkedIn o a un vídeo privado de YouTube con tu currículum, entre otras muchas posibilidades.

El futuro de la Realidad Aumentada

La realidad aumentada va a tener un desarrollo importante en el futuro con la implementación de los "Wearables", una nueva generación de dispositivos como gafas, relojes o pulseras que se han etiquetado como inteligentes.

Hemos visto como en el año 2012 Google presentó sus conocidas Google Glass, lo que ha abierto una puerta a los desarrolladores para crear productos similares y aplicaciones para dicho dispositivo. Varios competidores de Google han anunciado productos similares incluyendo diversas mejoras, el último de ellos ha sido Microsoft con su proyecto HoloLens que como ya hemos visto, genera "hologramas" en la vista de los usuarios con los que se podrá interactuar.

Pese a la gran popularidad que están adquiriendo en la actualidad estos "wearables", en especial las gafas de RA, todo apunta a que los avances tecnológicos tienden a la miniaturización de sus componentes paro con mayores pantallas. Y en esta dirección acabarán desembocando en la creación de una lentilla que permitiría el acceso a la RA.

Gafas de Realidad Aumentada - Google Glass

Como comentamos antes brevemente, la primera propuesta de proyecto fue la realización de un estudio sobre la Realidad Aumentada en Google Glass y reconocimiento de objetos o imágenes. La propuesta continuaba con el desarrollo de una aplicación para dicho dispositivo con un diseño centrado en el usuario, como en la propuesta actual.

En la búsqueda de emuladores e información para Google Glass, nos hemos encontrado con algunos "ports" de aplicaciones, o aplicaciones portadas; que funcionan sobre Android emulando la interfaz y sus comandos de voz de modo semi-funcional, además de algunos emuladores online para las aplicaciones.

Android

Para este sistema operativo, también el originario de las Google Glass, nos encontramos con que hay varios links interesantes sobre el emulador en cuestión como por ejemplo

una guía[14] en la que se nos enseña a instalarlo en un dispositivo Nexus 7 con la versión de Android Kit Kat 4.4.2.

También encontramos otros enlaces[15, 16] en los que nos dicen las aplicaciones que tenemos que instalar, el orden y la configuración que debemos ponerles para poderlas utilizar. Estas son las originales de las gafas, pero en una versión no final.

La funcionalidad de las estas podemos ponerla en entredicho, ya que reconoce los comandos de voz, pero da bastantes errores a la hora de usarla.

Además de lo anterior, encontramos en relación con las Google Glass un emulador de la API[17], la web oficial de desarrollo para Google Glass de Google[18] y un simulador de las gafas para el ordenador[19] que nos permite modificar el HTML y CSS de plantillas para ver en tiempo real los cambios que se producen. Las plantillas son las que se mostrarán en la pantalla de las gafas cuando se ejecute alguna de sus aplicaciones o comandos de voz.

Comparativa de herramientas actuales de desarrollo en AR

A la hora de comenzar el desarrollo de una aplicación de realidad aumentada (AR a partir de ahora), debemos saber qué nos ofrece cada entorno de desarrollo, herramienta y lenguaje ante los demás. Muchos de ellos se pueden incluir como un plugin externo a Unity 3D, entorno de desarrollo y motor de videojuegos multiplataforma. Sólo está disponible para Windows y OS X y permite crear aplicaciones para gran cantidad de dispositivos, además de aceptar plugins para ampliar aún más su abanico de posibilidades. Todo lo anterior hace que las herramientas que son compatibles con él estén más asentadas entre los desarrolladores.

Como posibles herramientas tenemos:

1. **ARToolKit[20] (1999):** Es una librería que fue desarrollada por Hirokazu Kato en 1999 y publicado por el HIT Lab[21] de la Universidad de Washington. Tiene gran cantidad de ejemplos y proyectos para ella. En cambio, si queremos que nuestra aplicación trabaje sobre Android, quizás no sea la más acertada ya que ARToolKit trabaja con C/C++ y habría que utilizar el NDK de Android para que

[14] http://elekslabs.com/2013/11/google-glass-development-without-glass.html – Último acceso: 22/01/2015
[15] https://github.com/zhuowei/Xenologer – Último acceso: 22/01/2015
[16] http://imgur.com/a/IBqFf – Último acceso: 22/01/2015
[17] http://glass-apps.org/google-glass-emulator – Último acceso: 22/01/2015
[18] https://developers.google.com/glass/ – Último acceso: 29/08/2015
[19] http://glasssim.com/ – Último acceso: 22/01/2015
[20] http://sourceforge.net/projects/artoolkit/ – Último acceso: 23/01/2015
[21] http://www.hitl.washington.edu/home/ – Último acceso: 20/01/2015

ejecute código en C. Al no ser este el lenguaje para el que fue diseñado Android contamos con que puede haber algún error en la ejecución e incompatibilidades. Como punto a favor, es de código abierto y multiplataforma.

2. **Total Immersion (D'Fusion)[22] (1999):** es una de las más utilizadas en el ámbito comercial. Dispone de una plataforma con herramientas para diseñar proyectos AR para móviles basados en la web. Tiene variedad de ejemplos comerciales[23] en la actualidad y cuenta con proyecto propio para probarse gafas mediante AR. Es multiplataforma; se encuentra disponible tanto para dispositivos móviles como para equipos de sobremesa. Algunos de sus clientes son: Toyota, Orange, Milka, Volkswagen, EBay, Paramount Pictures, Dior, entre otros; como se puede observar, su uso se extiende por diversos ámbitos.
3. **Metaio[24] (2003):** proporciona un SDK compatible con dispositivos móviles y pc, es de los que más características presenta, incorporando soporte para GPS, marcadores, reconocimiento de caras, búsqueda visual,… Incluye también un plugin para su desarrollo en Unity (si trabajamos con ese entorno), pese a tener el suyo propio, sólo compatible con Windows y Mac OS. Algunos de sus clientes son Mitsubishi Electric, Audi, Ikea, McDonald's, Toys 'r us y Lego.
4. **ARPA[25] (2006):** es una empresa española. Tiene un software propio de AR para la creación de modelos. Es compatible con IOS y Android y tiene un plugin para Unity3D. Según comentarios en el foro oficial, para poder crear tus propias imágenes y marcas necesitas una licencia comercial, aunque se puede tener acceso al SDK de forma gratuita. Entre sus socios se encuentran Bosch y Airbus Military.
5. **NyARToolKit[26] (2008):** es una biblioteca de clases de ARToolKit para máquinas virtuales, en particular para Java, C# y Android. Cuenta con poca documentación, es de código libre y dispone de un plugin para Unity3D.
6. **Wikitude[27] (2008):** es un SDK derivado de la famosa aplicación homónima. Incluye varias herramientas para que te centres solamente en el contenido de la aplicación, no en la complejidad de trabajar con AR. El contenido se basa en los estándares de HTML5. Puedes incluir tu aplicación dentro de la propia de

[22] http://www.t-immersion.com/partners/partners-programs/developer – Último acceso: 29/08/2015
[23] http://www.t-immersion.com/projects – Último acceso: 29/08/2015
[24] http://www.metaio.com/ – Último acceso: 29/08/2015
[25] http://www.arpa-solutions.net/es – Último acceso: 23/01/2015
[26] http://nyatla.jp/nyartoolkit/wp/ – Último acceso: 30/08/2015
[27] http://www.wikitude.com/ – Último acceso: 30/08/2015

"Wikitude" o exportarla aparte. Entre sus socios se encuentran empresas de la talla de Google, Epson, Adobe, Red Bull, Samsung, Sony, LG o Qualcomm.

7. **ARmedia[28] (2008):** es compatible con iOS, Android y Web. Cuentan también con el desarrollo de Windows Phone para ampliar su mercado. Su SDK es multiplataforma. Reconoce tanto marcadores convencionales como imágenes reales para proyectar su AR. Entre sus socios encontramos a Microsoft y Autodesk. Y sus clientes Intel, Sony, Ikea, Boeing o Ford.
8. **Layar[29] (2009):** al igual que Wikitude, publicó un SDK a partir del éxito de su aplicación. Es compatible con iOS y Android. Tiene bastante éxito en el ámbito empresarial. Proporciona también una API para que puedas usar sus datos.
9. **AndAR[30] (2010):** es la API de Java propia para Android. Es similar a ARToolkit, pero su desarrollo se detuvo en el 2010. Es de código abierto, multiplataforma y su desarrollo está probado en Android 2.1 y anteriores.
10. **Vuforia[31] (QAR 2010 – Vuforia 2012):** es un SDK de AR para dispositivos móviles que nos permite la creación de aplicaciones de AR. Es muy potente, cuenta con mucha documentación y no necesita de marcas concretas para poner un objeto, ya que se pueden usar imágenes u objetos reales. Uno de sus contras es que no se puede ejecutar desde el PC. Uno de sus ejemplos comerciales: https://www.youtube.com/watch?v=0l14awnnyqg. Algunos de sus clientes son Samsung, McDonald's o Toyota.
11. **AprilTags[32] (2011):** es más preciso que ARToolKit y útil para variedad de tareas. Utiliza los marcadores cuadrados y en blanco y negro. Además, se utiliza más en el ámbito de la robótica ya que sus marcadores fueron diseñados para una detección más robusta, rápida y con mayor ángulo de visión. Está disponible tanto en Java como en C, aunque la de Java ha quedado obsoleta y recomiendan usar la de C mediante JNI para su ejecución. Es de código libre y multiplataforma.
12. **ARLab[33] (2011):** es una empresa española, tiene diferentes SDK para AR. Se basa sobre todo en navegadores web, capturas de imágenes, 3D y botones virtuales. Tienen varios proyectos en su web y su licencia no es gratuita.

[28] http://www.inglobetechnologies.com/our-products/#developers – Último acceso: 30/08/2015
[29] https://www.layar.com/products/ – Último acceso: 30/08/2015
[30] https://code.google.com/p/andar/ – Último acceso: 24/01/2015
[31] https://developer.vuforia.com/ – Último acceso: 30/08/2015
[32] http://april.eecs.umich.edu/wiki/index.php/AprilTags – Último acceso: 30/08/2015
[33] http://www.arlab.com/ – Último acceso: 25/01/2015

13. **DroidAR[34] (2011):** es un proyecto Open Source sobre Android. No es compatible con iOS y funciona sobre Eclipse como entorno de desarrollo. Los marcadores con los que trabaja son los estándares en blanco y negro. Están trabajando en una segunda versión con varias mejoras y nuevas características.
14. **Aurasma[35] (2011):** tiene acceso a un SDK gratuito el cuál es compatible con aplicaciones en iOS y Android. Está muy extendido entre profesores para usarlo en las aulas. Algunos de sus clientes son Marvel, Disney, Universal, Kellogg's la BBC o Mercedes-Benz.
15. **Xloudia[36] (2012):** es compatible con todo tipo de dispositivos, proporciona gran variedad de opciones para el seguimiento de la cámara además de reconocimiento facial y de objetos. Como contra, podemos señalar que su SDK es de prueba durante 30 días y después se requiere el pago de licencia. Entre sus clientes encontramos Audi, Disney, Hugo Boss, "Le Louvre Museum" o HP dentro de un largo etcétera.
16. **ARcrowd[37] (2013):** es una aplicación online que permite crear y ver contenido AR sin precisar conocimientos de programación por parte del usuario. No necesita de ninguna instalación en pc, Smartphone o Tablet, simplemente con el navegador web puedes crear o reproducir contenido. Funciona mediante las marcas clásicas blanco/negro y permite asociar a ellas imágenes, vídeo, sonido y archivos "gif" entre otros.

Tabla comparativa de las herramientas

Para realizar una comparación simple en una tabla, vamos a seleccionar en primer lugar para qué SO móvil es compatible. Comprobaremos también si se puede ejecutar la aplicación creada desde un navegador Web y/o desde un PC para aumentar compatibilidad y alcance. Un rasgo que también tomaremos en consideración será el tipo de marca que soportan, ya sea las conocidas marcas en 2D (cuadradas y en blanco y negro) u objetos reales.

La siguiente característica que comprobaremos será la conocida en inglés como "3D Object Tracking" que consiste en la posibilidad de reconocer objetos como una esquina de la pared, o la forma de una caja para interactuar con ella; esta característica con su desarrollo puede llegar a reconocer edificios concretos. Una característica derivada de

[34] https://github.com/simon-heinen/droidar (actualizado) – Último acceso: 05/02/2015
[35] http://www.aurasma.com/ – Último acceso: 23/01/2015
[36] http://www.xloudia.com/start-today/ – Último acceso: 30/08/2015
[37] http://arcrowd.com/ – Último acceso: 23/01/2015

esta anterior es la de “FaceTracking”, reconocimiento y seguimiento facial; algo interesante para algunas propuestas y proyectos que también tendremos en cuenta en nuestra comparativa.

Para hacer uso de todas las posibilidades de los dispositivos móviles, comprobaremos también si los SDKs son compatibles con las tecnologías GPS y los conocidos como “IMU Sensors”, acelerómetros y giroscopios, que en algunos momentos pueden ayudar a ubicar tanto al dispositivo como a la imagen que se quiere mostrar.

Comprobaremos si el SDK es compatible con el motor de videojuegos Unity 3D para quitar complejidad al desarrollo de la parte gráfica junto con el código y si además tiene un plugin para acceder a él desde Eclipse ADT[38] o Android Studio[39].

Comprobaremos además la licencia que tiene, distinguiremos entre “Open Source” en la que podremos acceder al código y modificarlo, “Free” licencia gratuita con funcionalidades limitadas y/o marcas de agua de la empresa y licencia “Commercial” versión completa de la licencia y de pago.

Por último, comprobaremos si el SDK nos da la posibilidad de añadirlo como librería a nuestra propia aplicación y no tener que hacerlo a una aplicación de la empresa y acceder desde ella.

Una característica que hemos desechado en la clasificación es la de “Natural Feature”, que sería el seguimiento natural del objeto 3D conforme el movimiento de la cámara, algo que suelen tener todas las herramientas.

[38] http://developer.android.com/tools/sdk/eclipse-adt.html – Último acceso: 30/08/2015
[39] http://developer.android.com/sdk/index.html – Último acceso: 30/08/2015

	Móvil	Web	PC	Tipo Marca	3D Object Tracking	Face Tracking	GPS	IMU Sensors	Unity 3D o Plugin Android	Tipo Licencia	Añadir a app propia
ARToolKit	iOS / Android	**NO**	**NO**	B/N	**NO**	**NO**	**NO**	**NO**	**NO / SI**	Open Source	**SI**
Total Immersion	iOS / Android	**SI** (Flash)	PC	B/N y **objetos reales**	**SI**	**SI**	**SI**	**SI**	**SI / SI**	Commercial	**SI**
Metaio	iOS / Android	**SI**	PC/Mac	B/N y **objetos reales**	**SI**	**SI**	**SI**	**SI**	**SI / SI**	Free + Commercial	**SI**
ARPA	iOS / Android	**NO**	**NO**	B/N y **objetos reales**	**SI**	**SI**	**SI**	**NO**	**SI / SI**	Free[40] + Commercial	**SI**
NyARToolKit	Android	**NO**	**NO**	B/N	**NO**	**NO**	**NO**	**NO**	**SI / SI**	Open Source	**SI**
Wikitude	iOS / Android	**NO**	**NO**	B/N y **objetos reales**	**SI**	**NO**	**SI**	**SI**	**NO / SI**	Free + Commercial	**SI**
ARmedia	iOS / Android	**SI**	PC / Mac / Linux	B/N y **objetos reales**	**SI**	**NO**	**SI**	**SI**	**SI / SI**	Free + Commercial	**SI**
Layar	iOS / Android	**NO**	**NO**	**NO USA**	**NO**	**NO**	**SI**	**SI**	**NO / NO**	Free + Commercial	**NO**
Vuforia	iOS / Android	**NO**	**NO**	B/N y **objetos reales**	**SI**	**SI**	**NO**	**SI**	**SI / SI**	Free + Commercial	**SI**

Tabla 2 - Comparativa de herramientas

[40] Imposible comprobarlo pese a que la web lo anuncie así, no envía el correo para completar el registro y descargar el SDK

	Móvil	Web	PC	Tipo Marca	3D Object Tracking	Face Tracking	GPS	IMU Sensors	Unity 3D o Plugin Android	Tipo Licencia	Añadir a app propia
AprilTags	iOS / Android	**NO**	**NO**	B/N	**NO**	**NO**	**NO**	**NO**	**NO** / **SI**	Open Source	**SI**
ARLab	iOS / Android	**NO**	**NO**	B/N y **objetos reales**	**SI**	**SI**	**SI**	**SI**		Commercial	
DroidAR	Android	**NO**	**NO**	B/N	**SI**	**SI**[41]	**SI**	**SI**	**NO** / **SI**	Open Source	**SI**
Aurasma	iOS / Android	**NO**	**NO**	**Objetos reales**	**NO**	**NO**	**NO**	**NO**	**NO** / **NO**	Free + Commercial	**NO**
Xloudia	iOS / Android / Windows Phone	**SI**	**SI**	**Objetos reales**	**SI**	**SI**	**SI**	**SI**	**SI** / **NO**	Temporal[42] + Commercial	**SI**
ARcrowd	**NO**	**SI** (HTML5)	**NO**	B/N	**NO**	**NO**	**NO**	**NO**	**NO** / **NO**	Free	**NO**

Tabla 2 - Comparativa de herramientas (Continuación)

[41] Se le puede añadir mediante OpenCV
[42] Prueba de 30 días desde la compra http://2.lm3labs.com/product/1806/ – Último acceso: 13/03/2015

	Móvil	Web	PC	Tipo Marca	3D Object Tracking	FaceTracking	GPS	IMU Sensors	Unity 3D o Plugin Android	Tipo Licencia	Añadir a app propia
Metaio	iOS / Android	**SI**	PC/Mac	B/N y **objetos reales**	**SI**	**SI**	**SI**	**SI**	**SI** / **SI**	Free + Commercial	**SI**
Wikitude	iOS / Android	**NO**	**NO**	B/N y **objetos reales**	**SI**	**NO**	**SI**	**SI**	**NO** / **SI**	Free + Commercial	**SI**
ARmedia	iOS / Android	**SI**	PC / Mac / Linux	B/N y **objetos reales**	**SI**	**NO**	**SI**	**SI**	**SI** / **SI**	Free + Commercial	**SI**
Vuforia	iOS / Android	**NO**	**NO**	B/N y **objetos reales**	**SI**	**SI**	**NO**	**SI**	**SI** / **SI**	Free + Commercial	**SI**
DroidAR	Android	**NO**	**NO**	B/N	**SI**	**SI**[41]	**SI**	**SI**	**NO** / **SI**	Open Source	**SI**

Tabla 3 - Comparativa de las herramientas seleccionadas

Conclusiones tablas

Como conclusión de la primera de las tablas (Tabla 2 - Comparativa de herramientas) podemos ver que las herramientas más completas y que permiten el desarrollo de una aplicación propia, por orden de aparición son: Total Immersion de D'Fusion, Metaio y ARPA, que también lo podemos considerar una buena opción pero no conseguimos acceder al SDK por lo que es descartado, Wikitude puede resultar interesante pese a que no disponga de "Face Tracking" ni plugin para Unity 3D ya que dispone de su propio entorno para hacer un desarrollo simple y de las librerías aparte para un desarrollo más complejo. ARmedia tampoco dispone de "Face Tracking" pero sí de plugin para Unity y cumple con las demás características que hemos comprobado. Vuforia que, aunque no soporte GPS, es de las que más está evolucionando y de las que más se habla en la actualidad. DroidAR también parece una alternativa interesante que estudiar y por último tenemos a Xloudia que es una opción muy interesante también.

De las herramientas que acabamos de comentar, si seleccionamos las que tienen una licencia gratuita u "open source", y a cuya descarga podamos acceder; disminuimos la lista a: Metaio, Wikitude, ARmedia, Vuforia y DroidAR las cuales compararemos de nuevo en una segunda tabla con vistas al ejemplo en el que consiste la propuesta.

Alternativas como ARToolKit, NyARToolKit, Layar, AprilTags, Aurasma y ARcrowd han sido descartadas por ser antiguas y con pocas características que puedan competir con herramientas más actuales y asentadas en el mercado moderno como ARToolKit, NyARToolKit y AprilTags. Layar ha sido descartado por no permitir el usar el SDK fuera de la propia aplicación de la empresa. Aurasma y ARcrowd no las hemos seleccionado por no ofrecer ninguna de las características que necesitamos; cubren un nicho de mercado que no es el que buscamos.

Las herramientas han sido seleccionadas de la tabla anterior (Tabla 3 - Comparativa de las herramientas seleccionadas) en función de su implantación en el mercado por su número de clientes y grandes proyectos que hemos comentado con anterioridad en Comparativa de herramientas actuales de desarrollo en AR, sus características y que sean gratuitas para el desarrollo de este proyecto. Con cualquiera de ellas se podría hacer la aplicación que se propone.

Evolución tabla comparativa

Hasta que hemos llegado a la tabla comparativa que hemos mostrado con anterioridad, esta ha sufrido varios cambios importantes.

Una de las características que al principio entraban dentro de la comparativa pero que fue descartada por no tener utilidad era la de "Natural Feature", es decir; que si se está mostrando algo de RA en este momento y te mueves alrededor vas a ver cómo se mueve y ver todas sus partes. Esto lo hacen todos los SDKs, independientemente de su antigüedad.

En las primeras versiones de la tabla nos encontramos con una comparativa escasa como podemos ver:

	Móvil	**Web**	**PC**	**Tipo Marca**	**Natural Feature**	**3D Object Tracking**	**Unity 3D**
ARToolKit	iOS/Android	**NO**	**NO**	B/N	**SI**	**NO**	**NO**

Tabla 4 - Comparativa de las herramientas en sus primeras versiones

En esta comparativa podemos ver que muchas de los SDKs iban a tener las mismas características, por lo que la comparación no iba a sernos de utilidad.

En las siguientes versiones se añadieron características que nos parecieron pertinentes, como puede ser comparar el tipo de licencia del SDK, si tiene además un plugin para añadirlo a la aplicación con otros entornos.

Con estas adiciones, ya teníamos una comparativa un poco más extensa pero aún insuficiente:

	Móvil	**Web**	**PC**	**Tipo Marca**	**Natural Feature**	**3D Object Tracking**	**Unity 3D**	**Licencia**	**Plugin Android**
AR ToolKit	iOS/ Android	**NO**	**NO**	B/N	**SI**	**NO**	**NO**	Open Source	**SI**

Tabla 5 - Comparativa de las herramientas en sus versiones intermedias

Tras comenzar la comparativa e ir encontrando información, nos dimos cuenta de que no todos los SDKs nos permitían usarlos en nuestra propia aplicación sin tener que utilizar la propia de una marca o compañía y en otros casos, era confuso saberlo hasta que no comenzaras a usar el SDK, por lo que también lo añadimos a la tabla como dato importante.

Además, en esta última versión previa a comenzar a usar los SDKs, se le añadieron también otras características como pueden ser el "Face Tracking" (seguimiento de la cara) y "GPS e IMU Sensors" que nos permiten utilizar todos los sensores nuevos que

traen actualmente los dispositivos. Añadiendo también un toque de color para hacer más fácil su comprensión.

A la hora de localizar la información para rellenar la tabla, no ha sido suficiente con las páginas oficiales de cada una de las empresas, ya que no te dan toda la información hasta que comienzas a usarla o directamente te dicen que su SDK es gratuito, pero no te dan acceso a la descarga del mismo si no pagas una licencia, por lo que ya no es gratuito.

Principales problemas del desarrollo de la tabla

Los mayores problemas para localizar la información han estado de la mano del tipo de licencia, de si se puede añadir a nuestra propia aplicación, los sensores que permite utilizar, si disponen de un plugin para Unity 3D o una librería para añadirla aparte. Para localizar todos estos datos localizamos una web bastante útil[43] en la que se hace una comparativa de muchas herramientas, aunque nos hemos servido de ella con cautela, ya que manejaba numerosos datos que consideramos erróneos, como la licencia de Total Immersion que no es gratuita, así como SDKs que no se encontraban. Podemos explicar estos errores si consideramos que esta página se crea mediante contribuciones de los usuarios; debido a ello, se han modificado los datos pertinentes (y añadido otros) para mostrar una información lo más completa y veraz posible.

[43] http://socialcompare.com/en/comparison/augmented-reality-sdks – Último acceso: 30/08/2015

Diseño centrado en el usuario

Introducción a la metodología de desarrollo

Para el desarrollo de la aplicación, como se comentó en la introducción, se va a utilizar un diseño centrado en el usuario, que se basa en la metodología clásica de desarrollo del software, pero a la que se le añaden otros procesos entre los que se incluye el diseño de la interfaz mediante la construcción de prototipos y la evaluación de estos. La metodología clásica –o desarrollo en fases— comprenden: Análisis con especificación de requisitos, Diseño, Diseño detallado, Implementación e Integración y pruebas. En el diseño centrado en el usuario, conforme se realizan esas fases se va realizando un prototipo y su evaluación, permitiendo repetir alguna de las fases del diseño hasta que el prototipo se pueda considerar aceptable. Una vez se realizan todas las fases y el prototipo, se llega a una nueva fase que es la de mantenimiento. Este diseño en su conjunto, se conoce como "Diseño centrado en el usuario". En la metodología clásica, una vez se realiza una fase, no se vuelve atrás para revisarla. El diseño centrado en el usuario, utiliza en las 5 primeras fases la creación de prototipos y la evaluación de estos contando con la interacción de los usuarios que se han seleccionado principales y secundarios para la aplicación y de sus usuarios principales de uso, en función de sus respuestas se vuelven a realizar dichas 5 fases y finalmente cuando el prototipo llega a su fase final, se lleva a cabo la fase de Mantenimiento.

Un breve resumen de este proceso se puede observar en la siguiente imagen:

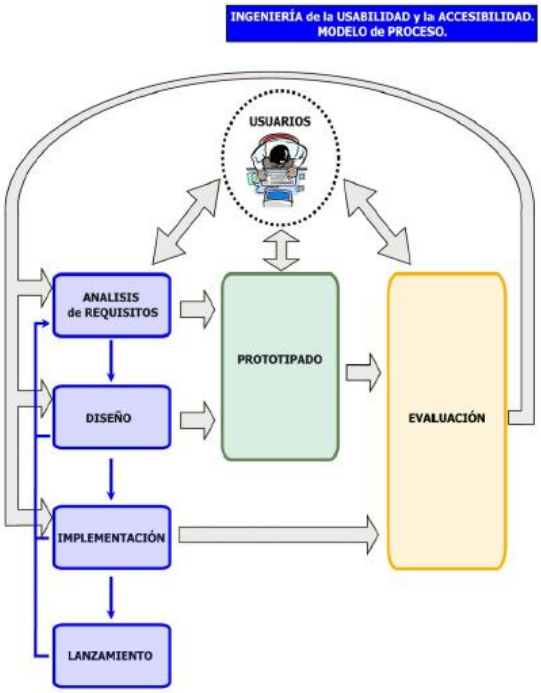

Ilustración 1 - Resumen del Diseño Centrado en el Usuario

Una de las partes más importantes del diseño centrado en el usuario es el propio usuario para el que se desarrolla la aplicación. Es necesario saber quiénes son los destinatarios y conocer sus inquietudes, una forma de hacerlo sería hablando con ellos mediante entrevistas por ejemplo u observar su día a día mediante estudios de campo. La técnica de "Personajes y Escenarios"[44] es un mecanismo para definir los objetivos de una aplicación y desarrollar sólo aquellas características que ayuden a cumplir esos objetivos. Los personajes no son reales, pero se tratan como si lo fueran. Esos personajes tienen que ser específicos, con las características que tendría cualquier persona. Dentro de los personajes, lo normal es que sólo exista un personaje primario al que hay que satisfacer al 100%, aunque también pueden existir personajes secundarios que son aquellos que la usarán y sacarán partido de la interfaz así como de cualquier elemento que pueda beneficiarlos. Hay que tener siempre en cuenta que en esta metodología nunca se debe anteponer la necesidad de un personaje secundario a la necesidad del primario. Los escenarios de cada usuario se encargarán de incorporar las tareas al diseño y sobre ellos se sustentará también el prototipo.

El prototipo que deberán evaluar, es una representación limitada del diseño y que permite a los usuarios interaccionar con él y explorar todas sus posibilidades. Es muy útil para hacer partícipe al usuario en el desarrollo y poder comprobar de una forma muy rápida y barata si la evolución del proyecto es adecuada, y en el caso de que no sea así, nos permite cambiarlo rápidamente. En función de las dimensiones del prototipo, tenemos dos tipos:

- Prototipado vertical: tiene implementadas pocas características, pero sus funcionalidades están totalmente operativas. Permite probar una parte limitada del sistema, pero se puede probar en profundidad bajo circunstancias reales.
- Prototipado horizontal: contiene toda la interfaz, pero no contiene funcionalidad. Es una simulación de la interfaz en la que no se puede realizar ningún trabajo real.

En función de la fidelidad que tenga el prototipo con el producto final también tenemos dos categorías:

- Prototipado de Baja Fidelidad: Se suele diseñar con lápiz y papel junto con algunas explicaciones durante su evaluación. No tiene mucho que ver su aspecto con el producto final, pero es rápido de hacer y barato de modificar.

[44] Cooper, A. (1999). *The Inmates Are Running the Asylum*. SAMS, Division of Macmillan Computer Publishing.

- Prototipado de Alta Fidelidad: Se diseña de la forma más parecida al producto final en términos de aspecto, interacción y tiempos.

Dentro de los tipos de prototipo nos podemos encontrar con los prototipos en papel que simulan la funcionalidad, los "storyboards" que funcionan como guión con viñetas que refleja la interacción y el recorrido de las distintas pantallas y simulaciones con algo de funcionalidad, entre otros. También existen los prototipos realizados con software, entre los que están las maquetas para tirar, ya que son prototipos similares a los de papel en los que su función es para la evaluación del usuario; incrementales, en los que se realiza por componentes separados y se van probando uno a uno hasta que se juntan y realizan una prueba final; y los evolutivos, en los que se usa como base para una próxima iteración en el diseño.

Otra de las partes más importantes es la evaluación del prototipo que debe estar presente durante todo el proceso. Gracias a ello, podremos constatar el alcance de la funcionalidad del sistema, comprobar el efecto de la interfaz e identificar los problemas potenciales del diseño. Hay muchos métodos de evaluación diferentes y se clasifican en función del lugar de evaluación, del estilo de esta, de cómo se recaba la información y de las personas que intervienen.

Según el estilo de evaluación, nos encontramos con que los métodos que se usan son varios, como los métodos de indagación, en los que se trata de averiguar información de los usuarios mediante observaciones de campo, entrevistas o cuestionarios; métodos de inspección, se examina la usabilidad de la interfaz mediante la evaluación heurística, recorrido de la usabilidad plural en el que se van tomando notas de los pasos que seguirías y se hace un debate; recorrido cognitivo, comprobando la facilidad de aprendizaje e inspección de estándares; y los test, en los que los usuarios trabajan con el sistema y los evaluadores van analizando los resultados.

Durante estas evaluaciones se va a ir viendo modificado el prototipo para adaptarse a las peticiones y gustos de los usuarios que se han seleccionado. En este paso, no podemos olvidar que siempre hemos de satisfacer por completo las demandas del usuario principal, ya que es el destinatario primario del proyecto. En cambio, las peticiones y opiniones que tengan los usuarios secundarios de la aplicación se tienen que intentar satisfacer todas también, pero con la condición de que esa modificación no entre en conflicto con ninguna de las demandas del usuario principal.

Las propuestas de las aplicaciones seguirán esta metodología de diseño y desarrollo para que sea lo más intuitiva posible para el usuario y lo más cercana posible.

Propuesta de aplicación: Aplicación de apoyo al estudio

La primera propuesta de aplicación que se va a llevar a cabo, consiste en el diseño y desarrollo de una aplicación Android en la que gracias a la Realidad Aumentada, profesores y alumnos podrán disponer de una manera más sencilla y visual ejemplos visuales de algunas partes de las enseñanzas de asignaturas de programación en la carrera de Informática. Esta aplicación permitiría a un alumno que esté estudiando enfocar con el móvil a los apuntes y mediante una pequeña imagen o patrón que la aplicación reconocerá, mostrar una animación con un ejemplo de lo que estudia como por ejemplo las estructuras de datos básicas pilas, colas y listas pasando también por cómo funciona un algoritmo recursivo entre otras muchas posibilidades. Hay muchísimo material tanto en la carrera de Informática como fuera de ella en la que se podría aplicar y no solo en el ámbito educativo, sino también en el ámbito de las conferencias.

Definición de usuarios de la aplicación

Para la definición de los usuarios los volveremos a subdividir en las categorías de primarios y secundarios.

Usuario primario: Álvaro Rodríguez

Datos personales:

- Nombre: Álvaro Rodríguez.
- Edad: 18 años.
- Estudios: bachillerato tecnológico.
- Profesión: estudiante del grado de Ingeniería Informática en Ingeniería del Software.

Contexto de uso:

- Dónde: en clase y en su casa cuando estudia.
- Cuándo: mañanas y tardes durante la época de exámenes.
- Tipo de dispositivo: teléfono inteligente de última generación.

Misión:

- Objetivos: entender las asignaturas de programación, ya que nunca ha dado programación antes y desconoce lo que es un bucle, una variable, un módulo o una estructura de datos. Desea solucionar el problema por el interés de superar el reto que le supone el comienzo de la carrera y el tener una buena base de cara al futuro.

- Expectativas:
    - Entender cómo funciona la programación sin tener que recurrir a clases particulares que le suponga un gran gasto.
    - Poder solucionar todas sus dudas conforme le vayan surgiendo mientras estudia las asignaturas y comienzas las prácticas y proyectos.

Motivación:

- Urgencia: alta.
- Deseo: alcanzar sus objetivos y no tener grandes dificultades durante su curso.
- Actitud hacia la tecnología: agresiva, siempre intenta tener lo último que haya en el mercado.

Escenarios de uso diario:

Mostrar los gráficos relacionados con los apuntes en cuestión para ayudar en el estudio.

Álvaro quiere intentar llevar la asignatura lo más al día posible, ya que considera la programación como una de las bases más importantes de la carrera. Además, la profesora suele mandarles para casa algunas páginas con ejercicios con pequeñas partes de código y tipos test en los que tienen que resolver pequeños problemas o implementar pequeños módulos. Para ello, Álvaro suele repasar un poco los apuntes de la clase del día y trata de hacerlos. Cuando enfoca con el móvil a algunas de las diapositivas que tienen una pequeña imagen o icono en la parte superior, puede ver pequeños gráficos y ejemplos que están relacionados con la explicación de esa diapositiva, lo cual le ayuda bastante a entender de una forma gráfica qué es realmente lo que está ocurriendo en la diapositiva. Gracias a ello, puede entender lo que hace las operaciones apilar y desapilar, cómo funciona una pila correctamente y por qué no puede acceder al último dato directamente.

Usuario secundario: Ana María Garrido

Datos personales:

- Nombre: Ana María Garrido.
- Edad: 32 años.
- Estudios: universitarios.
- Profesión: profesora titular en la Universidad de Extremadura en el departamento de Ingeniería de Sistemas Informáticos y Telemática en el área de lenguajes y sistemas informáticos.

Contexto de uso:

- Dónde: en clase principalmente y en el despacho.
- Cuándo: en horario de trabajo, mañana y tarde.
- Tipo de dispositivo: teléfono inteligente.

Misión:

- Objetivos: ayudar a los alumnos de su asignatura a entender las partes que puedan resultar más complejas para alguien que no tenga nociones de informática ni conocimientos de programación.
- Expectativas:
    - Mostrar animaciones gráficas simples con el contenido de las diapositivas.
    - Tratar de resolver todas las dudas que los alumnos puedan tener a la hora de estudiar la asignatura.

Motivación:

- Urgencia: media.
- Actitud hacia la tecnología: precavida, si funciona no lo toques.

Escenarios de uso diario:

Mostrar los gráficos relacionados con los apuntes en cuestión para ayudar en el estudio.

Ana María, en sus clases, para ayudar a la comprensión de sus diapositivas está probando cómo los alumnos se acostumbran poco a poco el uso del móvil para poder estudiar de una forma más dinámica y pudiendo ver los gráficos que ella incluye en sus explicaciones. Para ello, cuando hay alguna parte que considera compleja de entender, adjunta una pequeña imagen que sus alumnos puedan capturar para poder acceder al gráfico desde su propio móvil y sin complicaciones.

Para probar las animaciones antes de las clases, Ana María utiliza su teléfono y la aplicación que tiene instalada para tal función. La aplicación es muy simple, solo con abrirla ya se activa la cámara y, al enfocar a una de las diapositivas que tenga vinculada una animación, esta se reproduce automáticamente. Mientras está la animación en pantalla aparece también un botón de “pausa” para poder parar la animación en el momento que se desee. Y si pulsas sobre “pausa” este botón es automáticamente sustituido por el de “continuar”.

Escenarios de uso necesario:

Añadir los gráficos a la aplicación

Ana María se ha registrado en la aplicación con su cuenta de profesora, que le permite añadir información a la aplicación. Para ello ha tenido que conectarse a la versión web de la aplicación y adjuntar la animación que ya había elaborado y vincularla a una imagen. Automáticamente se añadirá a la lista de patrones reconocibles por la aplicación y podrá acceder a la información con sólo abrir la aplicación y conectarse a internet.

Modificar datos de una animación

Ana María desde su cuenta puede acceder a todas las animaciones que ha subido y a todos los patrones que ha vinculado con dichas animaciones. Para poder modificar alguna de ellas simplemente tendrá que seleccionarla y le saldrá un cuadro de diálogo en el que tendrá que elegir "Editar" y adjuntar o bien el nuevo patrón o la nueva animación para actualizar los datos.

Escenarios de casos límites

Eliminar los datos de una animación

En el caso de querer eliminar la información de una de las animaciones, los pasos son similares, ya que la diferencia es pulsar sobre la opción de "Eliminar" e introducir las credenciales para confirmar que es ella y proceder a la eliminación.

Desarrollo de prototipo

Para el desarrollo de prototipo, se realizará un prototipo de tipo horizontal y de baja fidelidad, uno de los varios tipos posibles que definimos anteriormente. Para su desarrollo, se ha utilizará la herramienta Pencil Project[45]. La herramienta funciona a modo de "drag and drop" (arrastrar y soltar). Una vez arrastrado el objeto que quieres, hay que configurar los enlaces para que cuando pulses sobre el objeto que quieras vaya a una vista o a otra. Con Pencil los prototipos se encuentran en formato HTML; es una página que incluye todas las vistas y la interacción funciona a través de los enlaces que hemos puesto previamente.

Durante el desarrollo se utilizará la metodología de desarrollo de "Diseño Centrado en el Usuario" (DCU, cuyos usos y ventajas con respecto a la metodología tradicional ya han sido explicados en este mismo trabajo.

[45] http://pencil.evolus.vn/ – Último acceso: 11/04/2015

Dentro del diseño, surgieron diferentes problemas que se han intentado resolver en el primer prototipo en lo referente al diseño de la aplicación y a la eficiencia de esta. Algunos de estos problemas serán consultados a los usuarios cuando se les haga la entrevista del prototipo para validarlo y son los siguientes:

- ¿Cómo deberíamos diseñar la aplicación? Realmente lo que nos interesa de la aplicación es que se puedan ver con claridad tanto los apuntes que están siendo enfocados y usados como base para mostrar las animaciones así como las propias animaciones con el suficiente tamaño como para que puedan entenderse. Por lo que no debería ser una interfaz muy sobrecargada ni con demasiadas opciones o botones en la pantalla principal.
- ¿Cómo tener la base de datos de la aplicación? Permitir que el usuario pueda acceder a la base de datos completa en cualquier momento con la consiguiente complejidad de la aplicación al reconocer un patrón u otro para mostrar la animación y el tamaño de la aplicación con todos los patrones guardados. Otra opción consistiría en permitir que el usuario seleccione en un momento dado, al abrir la aplicación por ejemplo; la asignatura que desea consultar y tener solo sus datos almacenados en ese momento. Una tercera posibilidad sería la de que el usuario pudiera subir su horario de clases para que la aplicación sólo accediera a la parte de la base de datos correspondiente a ese momento, el problema en esta opción radica en que el usuario no podría cambiarse de clase ni consultar la aplicación en su casa fuera del horario establecido.
- Añadir información en tiempo real para los usuarios que estén en ese momento en el aula. Otra opción que podría ser interesante en una aplicación de este tipo sería dar la posibilidad de que el profesor o un conferenciante quisiera compartir un dato en un momento dado bien porque se le ha olvidado incluir el código en las transparencias o porque quiera que solo pudieran acceder en ese momento en concreto.
- ¿Cómo implementar la transferencia de datos? En una clase normal hay una media de unos 70 alumnos por lo que habría que permitir que el servidor pudiera recibir más de 70 peticiones a la vez y enviar todos los datos a cada de una de ellas. En el caso de que esos datos fueran todas las animaciones de una asignatura, utilizaría bastante ancho de banda además de que la conexión Wi-Fi en las aulas no siempre funciona correctamente.

- ¿Cómo debería ser el patrón que tiene que reconocer la aplicación? Al principio de la idea, llegamos a pensar en que el propio SDK reconociera la diapositiva o el contenido, pero es algo que seguramente diera problemas con la gran semejanza que puede haber en las diapositivas. Dentro de los patrones que se pueden añadir, debemos tener en cuenta que no todas las personas imprimen sus apuntes en color, por lo que seguramente el patrón debiera ser en blanco y negro para asegurar que todo el mundo pudiera usarlo sin problemas. Y el debate entra cuando se tiene que elegir entre un patrón o usar un código QR que contenga la URL de la animación en el servidor para aligerar carga en el proceso de reconocimiento.

Evolución del prototipo

Primera versión

Se crea la primera versión del prototipo, con todos los enlaces y diseño como un menú lateral y un botón circular para abrir ese menú lateral. Por motivos de diseño, se considerará que es siempre el primer uso, por lo que tendrás que elegir siempre una asignatura al abrir la aplicación.

Este prototipo podemos verlo a continuación:

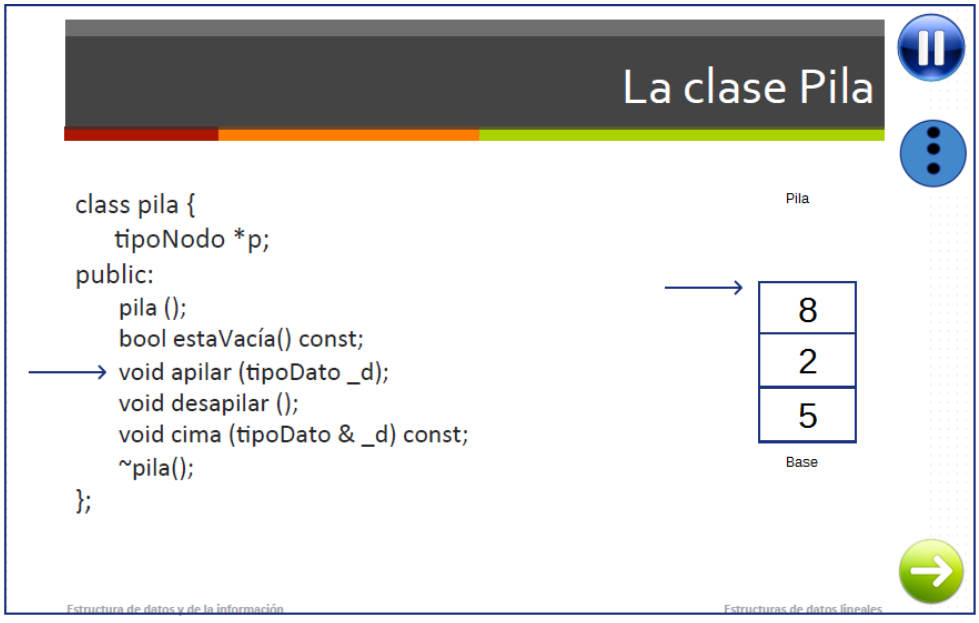

Ilustración 2 - Prototipo 1 de aplicación 1 operación apilar (8)

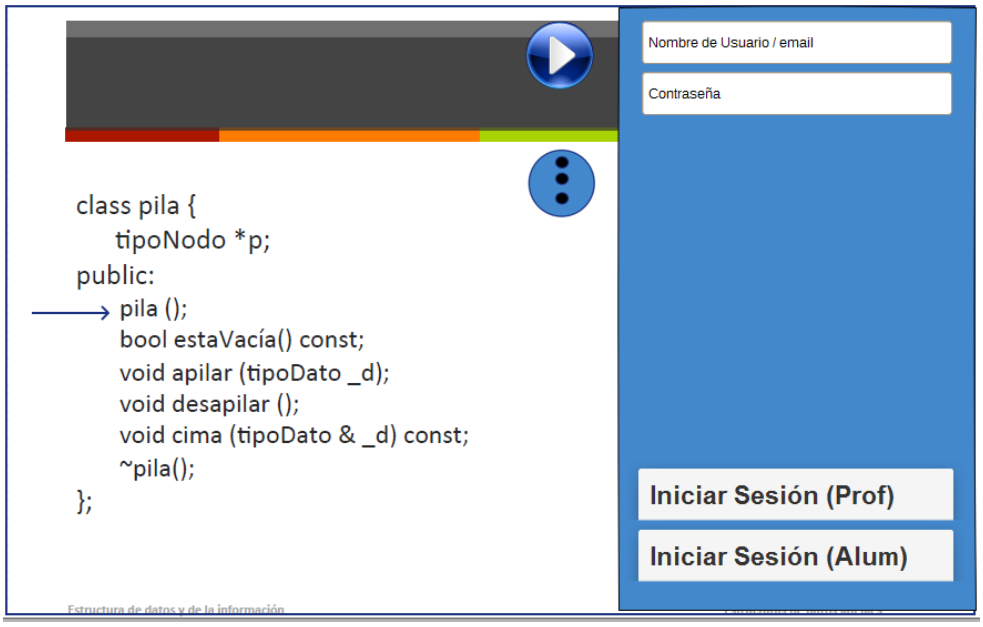

Ilustración 3 - Prototipo 1 de aplicación 1 al pulsar en iniciar sesión

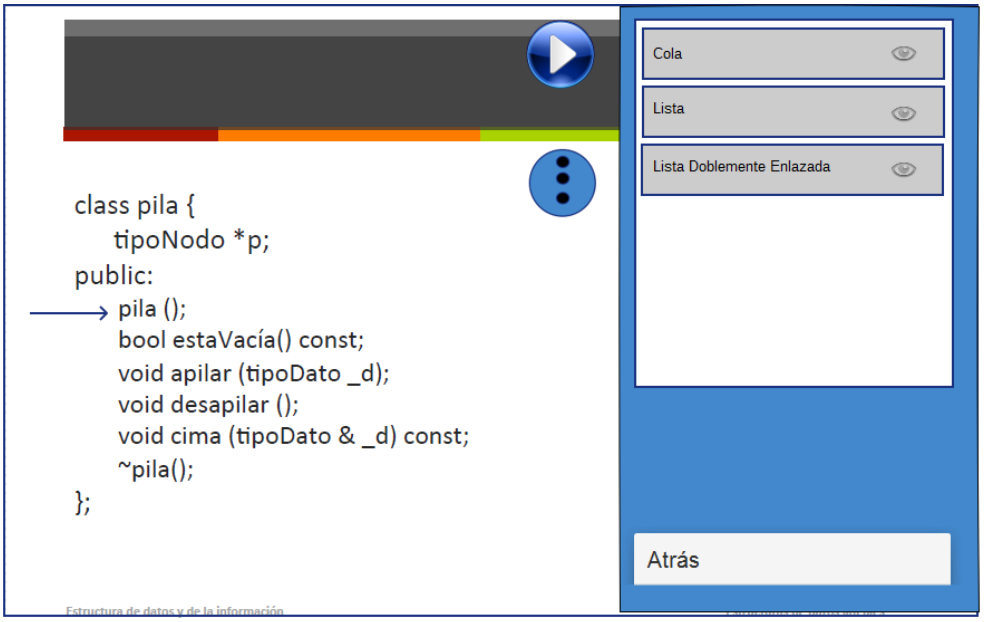

Ilustración 4 - Prototipo 1 de aplicación 1 al seleccionar activar animación oculta

Segunda versión

Se modifica el botón para abrir el menú lateral por otro cuya metáfora implique una barra de herramientas u opciones. Ese botón pasa de albergar una imagen con tres puntos a tener una imagen con un engranaje con la que el usuario está más familiarizado.

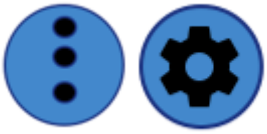

Ilustración 5 - Evolución Botón Opciones. Antiguo (izq.) y Nuevo (drcha.).

Además, dentro del menú lateral se añade un nuevo botón para poder volver al paso anterior sin tener que cerrar y abrir la barra, lo que hasta ahora implicaba comenzar de nuevo los pasos ya realizados.

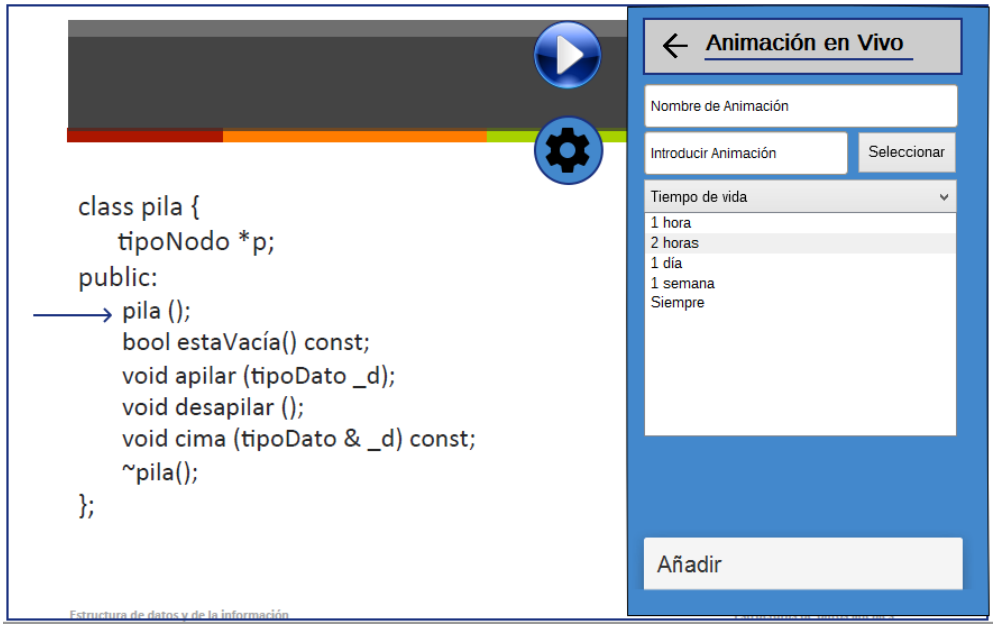

Ilustración 6 - Prototipo 2 de aplicación 1 con botón atrás en la parte superior derecha

Tercera versión

En la tercera versión del prototipo ya se ha realizado sobre un dispositivo real mediante Android Studio.

Hemos tenido que hacer un cambio en el diseño debido a que en la barra lateral o "Navigation Drawer" que teníamos previsto emplear y en la que concentrar toda la interacción del usuario con la aplicación, no hemos podido aplicar las modificaciones del contenido sin tener que recurrir a la creación de nuevas barras o a un tedioso funcionamiento para el usuario que implicaría abrir y cerrar la barra constantemente. Por lo anterior, hemos decidido hacer el diseño mediante una actividad diferente y con un diseño en el centro de la pantalla del dispositivo y creciendo o decreciendo en función del contenido. Es posible que cambiando la distribución de la nueva actividad que hemos creado podamos ponerla a la derecha como estaba pensado en un principio, pero eso deben decidirlo los usuarios en la entrevista.

Dentro del nuevo menú se han modificado algunos botones de "Configuración" y otros para mandar la navegación hacia atrás, por estar los primeros en demasiados menús y los de "atrás" por un error en el anterior diseño.

Para abrir este menú de interacción, tendremos que darle al botón azul con el engranaje del mismo modo que para la anterior propuesta, unas capturas de esta versión la podemos ver a continuación:

Ilustración 7 - Versión 3 de la propuesta 1. Menú sin iniciar sesión

Ilustración 8 - Versión 3 de la propuesta 1. Menú con sesión iniciada de profesor

Ilustración 9 - Versión 3 de la propuesta 1. Menú configuración de la cuenta

Cuarta versión

Tras una nueva evaluación por parte de los usuarios, se ha modificado el diseño de varias partes, como el botón "Atrás" que tiene la mayoría de las vistas, como podemos ver en la "Ilustración 9 - Versión 3 de la propuesta 1. Menú configuración de la cuenta" en la parte superior en el botón "Cuenta" que tiene el mismo tono que los demás botones y cuya función puede ser pasada por alto. Ahora ha pasado a tener un tono más claro para que resalte sobre el diseño.

Ilustración 10 - Versión 4 de la propuesta 1. Menú configuración de la cuenta

Además, después de una de las entrevistas, se ha modificado la interacción del usuario secundario a la hora de llegar a las acciones de dentro de una asignatura, pasando el proceso de seleccionar una asignatura y tener ya sus acciones a tener que seleccionar una asignatura, un tema o recursos concreto de esa asignatura que se haya añadido a la hora de crearla y ya después las acciones de esa asignatura y de ese tema o recurso en concreto.

Quinta versión

Después de haber realizado de nuevo las evaluaciones por parte de los usuarios, se ha modificado un submenú de configuración de un tema concreto, ya que tenía opciones tanto del tema en cuestión y de la asignatura en la que está ese tema. Se ha movido el botón de la configuración de la asignatura a la vista en la que salen todas las asignaturas. Estos cambios podemos ver en las siguientes imágenes:

Ilustración 11 - Versión 5 de la propuesta 1. Vista de la lista de asignaturas y el nuevo botón de configuración

Ilustración 12 - Versión 5 de la propuesta 1. Menú de configuración de un tema

También, se ha añadido un nuevo paso a la hora de eliminar una asignatura, teniendo que seleccionar la asignatura que deseas eliminar y tener que hacer una doble confirmación si realmente la deseas eliminar.

Ilustración 13 - Versión 5 de la propuesta 1. Vista de la lista de selección de la asignatura a borrar

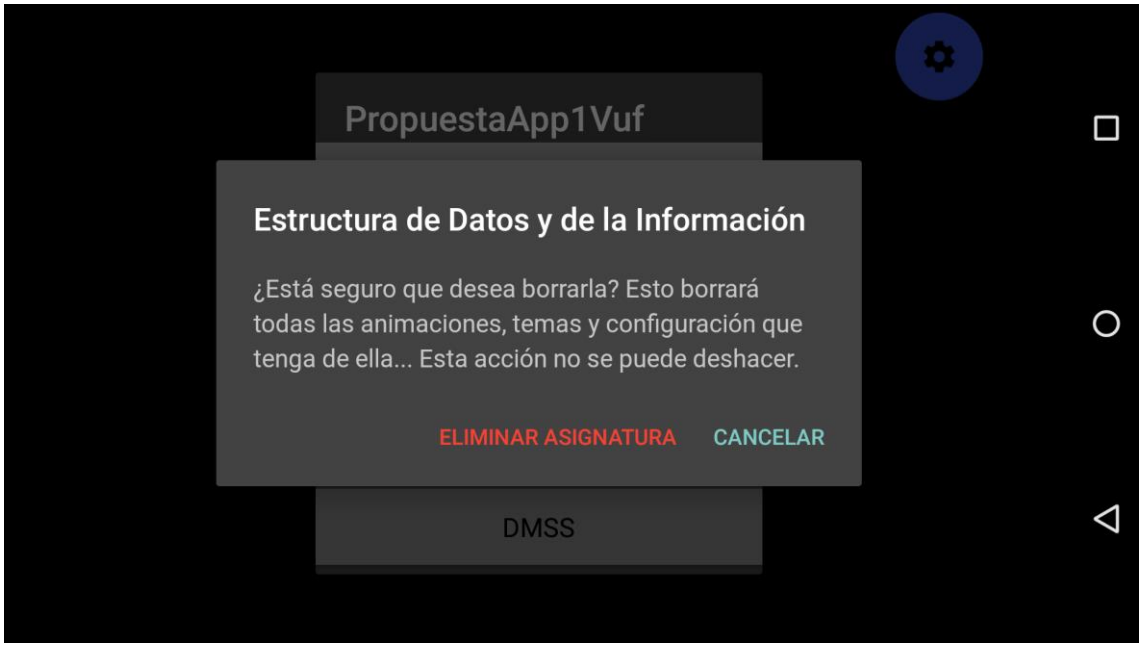

Ilustración 14 - Versión 5 de la propuesta 1. Vista del diálogo con la segunda confirmación al borrar.

Tareas a realizar en las entrevistas

Las tareas, al igual que en el prototipo anterior, se realizarán de manera esquemática con el software "Task Architect".

Las tareas que se han desarrollado para esta versión del prototipo son las siguientes:

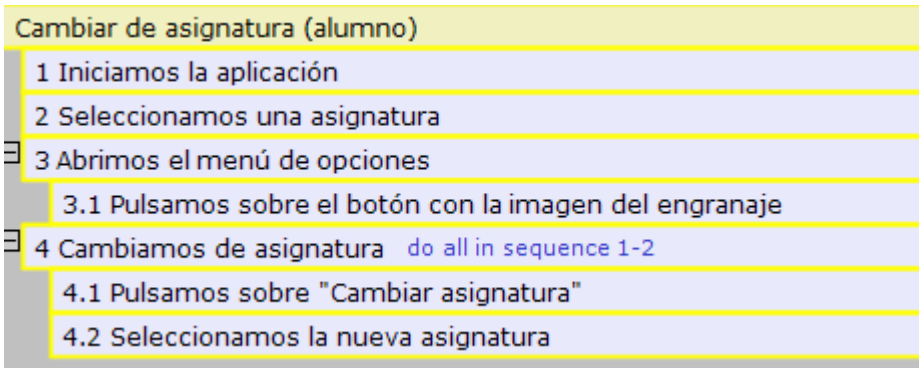

Ilustración 15 - Tarea de la aplicación 1 referente a "Cambiar de asignatura" para el usuario "alumno"

En la tarea correspondiente a la Ilustración 15 - Tarea de la aplicación 1 referente a "Cambiar de asignatura" para el usuario "alumno", podemos comprobar que es muy simple de realizar. Para el usuario, cambiar de asignatura es un proceso muy simple, tras haber iniciado la aplicación sólo tendría que seleccionar una asignatura para poder continuar y abrir la barra de opciones pulsando sobre el engranaje. Por último, para poder cambiar de asignatura, tendría que pulsar sobre el botón que pone "Cambiar asignatura" y seleccionar la asignatura deseada.

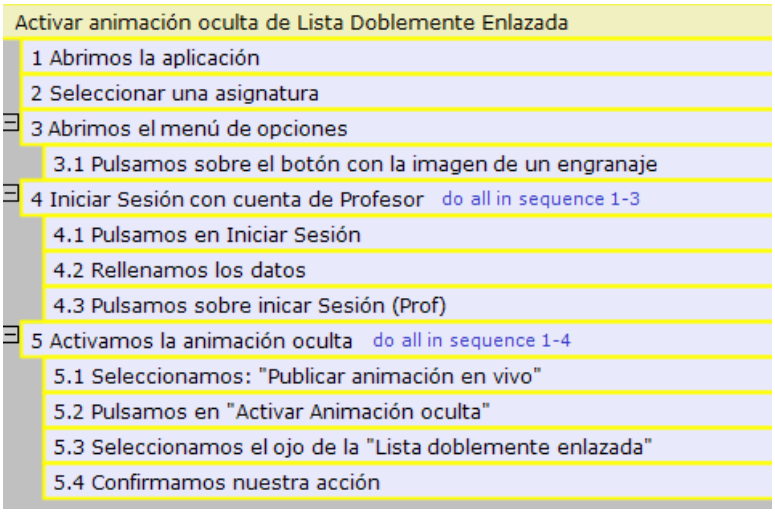

Ilustración 16 - Tarea de la aplicación 1 referente a "Activar animación oculta de Lista Doblemente Enlazada" para el usuario secundario

En la tarea descrita en la Ilustración 16 - Tarea de la aplicación 1 referente a "Activar animación oculta de Lista Doblemente Enlazada" para el usuario secundario, vemos que solamente puede realizarla un usuario secundario o profesor, que son los que alimentan de información a la aplicación. Para poder activar una animación oculta, primero necesitaremos abrir la aplicación y seleccionar una asignatura cualquiera, ya que como hemos comentado el prototipo está ideado como si fuera el primer uso de la aplicación. Después abriríamos el menú de opciones pulsando sobre el icono con el engranaje iniciaríamos ahora sesión como profesor, para ello realizaremos las subtareas por orden que en este caso, serían pulsar sobre el botón de iniciar sesión para que se nos cambie de vista, rellenar los datos correctamente y pulsar sobre el botón de “Iniciar Sesión”. Ahora, para activar la animación oculta, se debería que seleccionar la opción de “Activar animación en vivo” y “Activar animación oculta” después. En la nueva vista, seleccionaremos la “Lista doblemente enlazada” para que se active y confirmaremos la acción. Con este proceso, ya tendremos nuestra animación disponible para todo el público.

Resultados de entrevistas para validar el diseño del prototipo

A la hora de realizar las entrevistas nos hemos servido de las técnicas de “Thinking Aloud”, práctica que consiste en pensamiento en voz alta en el que el usuario entrevistado va diciendo en voz alta todo lo que se le ocurre mientras usa la aplicación y realiza las tareas encomendadas. Otro tipo de evaluación que se ha llevado a cabo junto con la anterior, es la de “Coaching Method” es decir, el método del conductor; en el que el experimentador trata de conducir al usuario en la dirección correcta mientras se usa el

sistema y así no se pierda. Este método choca un poco con la mayor parte de los métodos de evaluación en los que el experimentador trata de interferir lo menos posible con la persona que realiza el test.

Antes de la evaluación, se informó a los usuarios de por qué se requería su atención a la hora de evaluar los prototipos y cuál era su importancia en ellos. Además, también se les informó de las tareas que tenían que realizar y de cómo tenían que realizarlas siguiendo las dos técnicas de evaluación que hemos comentado anteriormente. Este proceso se realiza con un guión de orientación u "Orientation Script" que ha de hacerse antes de cada entrevista para conseguir que el usuario esté tranquilo. Dicho guión nos ofrece garantías de que es en realidad la persona "adecuada" en el lugar "adecuado". También se le dice que estaremos a su lado tomando algunas notas sobre las acciones que está realizando y sobre los pensamientos que pudieran decir en voz alta y las dudas que le surgiera, además de que estamos disponibles por si tiene alguna duda o necesita hacer alguna pregunta en cualquier momento.

Primera versión

Entrevista 1 – Usuario secundario

Usuario en el rol de profesor. No ha tenido problemas a la hora de completar la tarea para mostrar una animación en vivo y seguir el orden de la ejecución sin problemas excepto en el penúltimo paso, que en lugar seleccionar la imagen con el "ojo" para hacerlo visible, pulsó sobre el "botón" que correspondía al objeto que quería mostrar, es decir al cuadro o celda que contenía toda la información de la "Lista" y la imagen del "ojo".

El usuario comentó también que, pese a que el icono de los 3 puntos le llamaba la atención para pulsar sobre él pensando que habría algo más detrás como un etcétera, él usaría algo como un engranaje para que se entendiera que te refieres a "opciones". Además, que la imagen al cargar la aplicación de seleccionar una asignatura no debería de aparecer o aparecer al menos solo la primera vez.

Entrevista 2 – Usuario primario

Usuario en el rol de alumno. No ha tenido problemas para cambiar de asignaturas una vez tenía seleccionada la primera al iniciar la aplicación.

Los únicos comentarios que ha ido realizando durante la evaluación eran sobre cómo funcionaba realmente la animación porque no la entendía.

Segunda versión

Entrevista 1 - Usuario secundario

Usuario en el rol de profesor especializado en el ámbito de la Realidad Aumentada y aplicaciones con ella. No ha tenido dudas al realizar las tareas encomendadas, como iniciar sesión, activar una animación oculta o añadir una nueva animación.

Ha planteado la posibilidad de añadir un pequeño tutorial al inicio con muy poco texto sobre el uso de algunas partes, ya que podría resultar confuso, pero en cambio, si hay mucho texto por lo general no se suele leer. Otra de sus dudas fue sobre la creación de las animaciones y del formato soportado, pero eso en una fase tan temprana del diseño de la aplicación aún no se ha comprobado en profundidad.

Entrevista 2 – Usuario secundario

El usuario ha realizado correctamente las tareas de registrarse e iniciar sesión sin ningún tipo de problema. En la última tarea que tenía que realizar, activar una animación oculta; tuvo dudas al pulsar sobre el botón "publicar animación en vivo" y preguntaba a qué se refería ese botón y a qué asignatura. "Si es para una diapositiva concreta, de un tema concreto de una asignatura concreta, debería poderse dar la opción y si es para la diapositiva actual, debería indicarse" comentaba durante el test.

Durante la traza de la animación, también comentó que debería verse en la misma animación cuál es el paso siguiente que va a tener lugar antes de que lo tenga, a modo de vista previa, como sucede en el juego de "Tetris". Si no parece que los datos se generan de forma espontánea. También mencionó que en un ejemplo en el que hubiera más código y no solo los módulos principales de la clase, se entendería mejor sobre todo esta parte de qué es lo siguiente que va a ocurrir.

También, sugirió que, al activar una animación oculta debería salir sobre qué asignatura, o que te de la opción de la asignatura actual; te pida el número del tema al que está asociado esa animación y o bien que salga una lista de las animaciones de ese tema o la aplicación tenga conocimiento también de la diapositiva a la que está vinculada.

En lo referente al diseño de la diapositiva, para diferenciar cuál tiene una animación de Realidad Aumentada vinculada y cuál no, debería tener un icono, patrón o código QR en la diapositiva para que ni el alumno ni el profesor estuvieran con el móvil en todas las diapositivas para ir comprobando. Se le comentó que eso iba a ser modificado en la siguiente fase de desarrollo, cuando se empezara el desarrollo de la aplicación para que pudiera reconocer la diapositiva.

El usuario también se interesó por la parte servidora de la aplicación, de cómo obtendría los datos. Dio la idea del funcionamiento de la aplicación "Spotify" que descarga la canción conforme la vas escuchando en bloques de unos 30 segundos para no sobrecargar el servidor ni descargar completa la canción. Aplicar este funcionamiento a nuestro entorno, consistiría en que la aplicación se descargara la información de la animación solicitada y quizás las 3 posteriores, para agilizar la consulta, a no ser que descarguemos todas las animaciones de un tema completo.

Entrevista 3 – Usuario secundario

Al igual que el usuario anterior, al abrir la aplicación del prototipo, comentó sus inquietudes acerca de dónde salían los datos. "Debería verse algo más de código para ver realmente lo que se está haciendo en ese momento y así intuir el siguiente paso por la siguiente línea de código, o mostrar también una vista previa del siguiente paso".

Ya dentro de la barra de opciones, en el apartado "Mis Asignaturas" se extrañó un poco por cómo funcionaba, ya que se esperaba que al pulsar, además de darle una lista de las asignaturas que tiene, entendía que al pulsar sobre una de ellas saldría una lista de los temas y/o diapositivas con animaciones asociadas.

Al realizar la tarea de "Activar la animación oculta de la Lista Doblemente enlazada" en el último paso, el de activar la animación, ha pulsado sobre la celda completa en la que se encuentra el nombre de la animación, pensando que si pulsaba ahí el ojo cambiaría de un color gris a verde y eso significaría que ahora estaría visible.

Mientras realizaba la tarea de "Añadir una nueva animación", comentó que sobre qué diapositiva o imagen se realizaría. Ya que él suponía que sería sobre la actual, y como solución decía: "además de darte la opción de si la quieres añadir a esa, debería salir una lista con las diapositivas que tiene la asignatura y que la aplicación tenga conocimiento de ellas, quizás por el número de diapositiva de cada tema".

Entrevista 4 – Usuario primario

Realizó las tareas de registrar, iniciar sesión y cambiar de asignatura correctamente. El usuario, al pulsar sobre el botón de "Asignaturas Favoritas" preguntó cómo se añadían las asignaturas favoritas, y la verdad es que aún no está puesta la opción; falta esa parte del diseño.

Entrevista 5 – Usuario secundario

Durante el test no mostró ningún tipo de problemas al realizar las tareas de registrarse, iniciar sesión y activar una animación oculta.

Respectó a esta última tarea, comentó que las dos opciones que hay dentro de "Añadir animación en vivo" son muy parecidas, que igual habría que modificar un poco el texto porque leyéndolo rápido es fácil equivocarse al seleccionar una opción. Otra opción también sería añadirle alguna imagen a modo de metáfora, como al de añadir una nueva animación ponerle un símbolo "+" y a la opción de "Activar una animación oculta" ponerle una de un botón de "ON/OFF" o algo que rápidamente te lleve a pensar que es para activar sin tener que leer nada.

Tercera versión

Entrevista 1 – Usuario secundario

Durante el test no tuvo problemas a la hora de realizar las tareas ya en un dispositivo real y con la interfaz cambiada de la versión anterior a esta. Expresó que le resultaba más cómoda esta interfaz al estar el menú en el medio y era más fácil verlo y centrarte en ello, ya que la acción de la cámara permanece en pausa mientras este menú está abierto.

Volviendo a la evaluación, como hemos dicho en el proceso de realizar las tareas no tuvo problemas al completar las acciones solicitadas haciendo el recorrido por los menús en el orden correcto. Excepto en la tarea de "cerrar sesión del usuario", donde en vez de pulsar atrás, ya que tenía en el menú abierto, cerró y abrió el menú para empezar de nuevo el proceso.

Entrevista 2 – Usuario secundario

Respecto a la evaluación del prototipo realizando las tareas solicitadas, no tuvo ningún problema a la hora de llegar a los menús y acciones. Ahora, cada vez que tenía que volver atrás en un menú, en vez de pulsar sobre el botón "Atrás" de cada vista, cerraba y abría cuadro de diálogo para comenzar de nuevo el proceso. Por lo que para la siguiente versión del prototipo se modificarán estos botones para cambiarle el color a uno que resalte más y se use en la interacción.

Al comenzar la evaluación y abrir la aplicación por primera vez, comentó que no le desagradaba la nueva versión del diseño del menú de opciones. Durante el proceso también comentó que la forma de ir concretando en una asignatura a la hora de acceder a sus animaciones y opciones, le parecía un acierto y le gustaba la forma en que funcionaba. Por otro lado, comentó si sería posible modificarlo para que en vez de pulsar una asignatura y que te salgan ya sus acciones con todas las animaciones de esa asignatura, que pueden llegar a ser muchas; si se podría ir concretando en el tema cada vez, pulsando sobre una asignatura, a continuación, sobre un tema o recurso concreto y

por último que te saliera el menú de interacción que ya tenemos, pero ahora con las acciones y animaciones concretas de un solo tema y no de todos. Este proceso sería menos tedioso para enfrentarse a la aplicación ya que sería semejante a la plataforma Moddle que tienen los profesores en la web del Campus Virtual de la Universidad de Extremadura.

Cuarta versión

Entrevista 1 – Usuario secundario

Mientras se realizaba la evaluación de las tareas que se encomendaron no hubo problemas a la hora de su realización siguiendo un orden correcto por los menús. El único problema que encontró fue a la hora de "cerrar sesión" que no encontraba la opción y comentó que no le veía mucho sentido dentro del apartado "Configuración" pero que una vez lo habías hecho ya sabías que estaba ahí y no en ningún otro apartado, pero no entraría ahí como primera opción.

Durante el desarrollo de la entrevista, comentó que el menú de configuración de la asignatura, debería ir dentro de la vista en la que salen las asignaturas y no mezclarse con las del tema. Y también que no sabía la utilidad de la opción "añadir animación en vivo" fuera del apartado de las asignaturas y se le recordó que esa opción sería para añadir un recurso de AR que no tiene por qué estar asociada a una asignatura o tema.

Propuesta de aplicación: Tarjetas de visita

La propuesta de la aplicación que se va a llevar a cabo, consiste en el diseño y desarrollo de una aplicación para el sistema operativo Android que mediante Realidad Aumentada sea capaz de reconocer "tarjetas de visita" y permitir que estas muestren una información mucho más completa de la que disponen en la actualidad. Además de que permitiría mediante un servidor web mantenerlas actualizadas en el caso de que no queramos cambiar el diseño de las mismas. Esta aplicación permitiría desde cosas básicas como que al enfocar a la tarjeta te salga un teléfono o un sobre y al pulsar sobre ellos pudieras llamar o mandar un correo electrónico para poder contactar con el dueño, hasta que al enfocar a la tarjeta se abriera un vídeo de YouTube en el que la persona dueña o la empresa te explican quiénes son, a qué se dedican. Hay muchísimo sobre lo que innovar en esta materia.

Para el desarrollo de la aplicación se utilizan las mismas herramientas que para la primera propuesta y se realizará también un diseño centrado en el usuario.

Definición de usuarios de la aplicación

Para definir a los usuarios que van a utilizar la aplicación vamos a catalogarlos en primarios y secundarios. Al usuario primario, que por lo general suele ser uno; hay que satisfacerlo completamente tanto en usabilidad y características de la aplicación como en la interfaz. Las demandas de los usuarios secundarios serán satisfechas siempre y cuando no interfieran con las peticiones del usuario prioritario, el principal.

Los escenarios son los encargados de incorporar tareas al diseño de la aplicación. Nos encontraremos con tres tipos de escenarios, diferenciados por el nivel de uso que le dará el usuario y su prioridad de cara al desarrollo de la aplicación. Diferenciamos escenarios de uso diario, de uso necesario y de casos límite.

Usuario primario: Juan Rodríguez

Datos personales:

- Nombre: Juan Rodríguez.
- Edad: 37 años.
- Estudios: universitarios.
- Profesión: representante de una empresa de soluciones tecnológicas.

Contexto de uso:

- Dónde: en su despacho y en charlas.
- Cuándo: mañana y tarde.
- Tipo de dispositivo: teléfono inteligente de última generación.

Misión:

- Objetivos: conseguir información de posibles clientes y ponerse en contacto con ellos.
- Expectativas:
  - Poder acceder con facilidad a la web pública del cliente.
  - Poder acceder a perfiles públicos en redes sociales.
  - Reproducir contenido audiovisual.
  - Obtener forma de contacto como correo electrónico o teléfono.

Motivación:

- Urgencia: quiere una aplicación que realice ese trabajo para poder utilizarla ya.
- Deseo: ahorrarse tiempo en el trabajo y poder rendir más.
- Actitud hacia la tecnología: agresivo, siempre a la última.

Escenario de uso diario:

Mostrar datos y ponerse en contacto con la persona dueña de la tarjeta.

Juan ha reunido varias tarjetas en la última conferencia a la que fue sobre Realidad Aumentada y quiere separar las tarjetas en función de los proyectos y clientes que han tenido cada una de las empresas para ver si han hecho proyectos parecidos a los que él está interesado. Cuando enfoca a la tarjeta observa que le salen varios iconos de las redes sociales de la empresa, un acceso a la página web oficial, otro acceso, pero con la dirección de correo de contacto; otro con el teléfono del dueño de la tarjeta y, por último, el canal en YouTube en el que se pueden ver ejemplos de los proyectos que han realizado con anterioridad.

Escenario de uso necesario:

Registrarse en la aplicación y seleccionar las opciones que deseas que se muestren

Juan suele estar al día de la actualidad tecnológica, le gusta saber cómo avanza cada día la tecnología. En la última charla que fue, le interesó mucho un "stand" que había sobre soluciones con Realidad Aumentada aplicada a cosas tan simples como pueden ser tarjetas de visita para ofrecer más datos de los que se puede mostrar en un pequeño trozo de cartulina. Allí conoció a Manuela Jiménez que le dio una de sus tarjetas y le explicó cómo registrarse y añadir algo de información para enseñárselo a sus alumnos y que se interesasen por este mundo que está aún en auge.

Aprovechando los ratos libres que tenía, Juan se aventuró a registrarse en la aplicación. Era un proceso muy simple, solamente se le pedía un nombre de usuario, un correo electrónico para validar su cuenta y estar al tanto de ofertas y noticias y una contraseña. Tras haber creado su cuenta y confirmarla mediante el correo que recibió en su cuenta, accede a la interfaz básica de la aplicación. En ella puede ver que tiene una barra con diferentes componentes en la parte derecha entre los que se encuentran componentes de YouTube, Twitter, Facebook, añadir imagen, una página web, entre otros. Arriba a la derecha ve que hay un botón verde en el que pone Mi Cuenta para modificar su información o sus tarjetas. En la parte superior tiene una notificación para que añada una imagen sobre la que se mostrará la realidad aumentada.

Al seleccionar la notificación le saldrá un menú con dos opciones, una de ellas será para seleccionar una imagen de su equipo y la segunda, si no tiene una imagen, la aplicación le da la opción de crear su propia tarjeta y enviarle un lote con ellas a una dirección previo pago. Juan selecciona una imagen que tenía preparada ya en su equipo para probar la aplicación. Una vez subida se le cierra la notificación.

En la pantalla anterior, aparece en el fondo la imagen que ella acaba de subir y en la barra superior un nuevo botón de finalizar edición. Sobre la imagen empieza a poner los componentes. Arrastra el de ver una imagen y le sale un menú en el que seleccionar la imagen desde su equipo o desde la web. Al seleccionar una imagen desde su equipo quiere finalizar el proceso, pulsa sobre el botón de finalizar y aparece una notificación que le avisa de que ya puede utilizar su dispositivo móvil para enfocar a su tarjeta desde la aplicación para ver cómo ha quedado su diseño.

Escenarios de casos límites

Borrar una tarjeta

Juan quiere borrar una tarjeta que tenía en la que tenía que cambiar varios de los componentes para tenerla actualizada y prefiere borrarlos y crearla de nuevo. Para borrar una tarjeta inicia sesión con sus datos en la aplicación, observa todas las tarjetas que tiene y selecciona la que quiere borrar. Se le abrirá la interfaz de edición de la tarjeta y pulsará sobre el botón de eliminar tarjeta. Se le abrirá un cuadro de diálogo preguntándole que si está seguro de lo que va a hacer y al seleccionar sí se le habrá borrado la tarjeta.

Borrar cuenta

Juan desea borrar su cuenta completa porque quiere borrar todas las tarjetas que tiene vinculadas. Para ello inicia sesión en la aplicación con sus datos y al pulsar sobre la opción “Mi Cuenta” accede a la pantalla en la que se encuentran sus datos y debajo un botón hay un botón modificar datos y al lado uno de eliminar cuenta. Al pulsar sobre eliminar cuenta le sale un cuadro de diálogo que le advierte de que si borra la cuenta se borrarán todas las tarjetas asociadas junto con todos sus datos. Junto con otros dos botones de cancelar y otro en rojo de borrar cuenta. Al pulsar sobre este último, automáticamente se cierra la sesión de la aplicación y se borran todos los datos relacionados con su cuenta.

Usuario secundario: Aseret Ibáñez

Datos personales:

- Nombre: Aseret Ibáñez.
- Edad: 33 años.
- Estudios: universitarios.
- Profesión: profesora de universidad.

Contexto de uso:

- Dónde: en su despacho.
- Cuándo: mañana.
- Tipo de dispositivo: teléfono inteligente.

Misión:

- Objetivos: que los alumnos accedan a su horario de tutorías y a contactar con ella; y seleccionar alumnos solicitantes de prácticas en su departamento en función de sus conocimientos mostrados públicamente en redes como LinkedIn.
- Expectativas:
  - Permitir el contacto con ella de forma rápida y cómoda.
  - Obtener información de forma rápida y correcta de personas de interés.

Motivación:

- Urgencia: media.
- Deseo: conseguir una forma de mantener la información de sus contactos actualizada y acceder a esa información rápidamente. Además, quiere poder superar la barrera de los problemas de visión al recibir una tarjeta convencional.
- Actitud hacia la tecnología: precavida.

Deficiencias:

- Astigmatismo, problemas de visión de cerca.

Escenario de uso diario:

Mostrar datos y ponerse en contacto con la persona dueña de la tarjeta.

Aseret desea que sus alumnos tengan un acceso rápido a su horario de tutorías y a su horario de clases para que puedan plantear las dudas que tengan dentro de ese horario y así poder rendir más el resto del día en otras tareas. Por eso ha puesto una imagen tanto en la puerta del despacho como en los apuntes de clase que pueden “capturar” los alumnos con sus móviles y acceder a los datos con facilidad. Además, le gustaría también que mostrara su dirección de correo para agilizar la forma de contacto con ella abriendo directamente la aplicación del correo del dispositivo.

Cuando los alumnos enfoquen la tarjeta, les aparecerá en la interfaz 3 iconos sobre los que poder pulsar, los accesos a las imágenes o documentos con los horarios de clase y

tutorías y el último un sobre en el que al pulsar serían redireccionados a la aplicación de correo.

Usuario secundario: Enrique Meléndez

Datos personales:

- Nombre: Enrique Meléndez.
- Edad: 25 años.
- Estudios: universitarios y bellas artes.
- Profesión: profesor de arte y pintor vocacional.

Contexto de uso:

- Dónde: convenciones y galerías de arte.
- Cuándo: mañana y tarde.
- Tipo de dispositivo: teléfono inteligente.

Misión:

- Objetivos: posibles clientes que a través de su tarjeta de visita puedan acceder a una galería con parte de sus obras desde la misma aplicación o a una dirección web donde se puedan ver. Además, desea que también se puedan poner en contacto con él con facilidad y quizás ver dónde será su próxima exposición o charla.
- Expectativas:
    - Espera encontrar una experiencia atractiva y simple para que el usuario desee utilizarla y encontrar sus obras.

Motivación:

- Urgencia: media.
- Deseo: desea alcanzar esos objetivos para ampliar el público que puede acceder a sus obras
- Actitud hacia la tecnología: precavida.

Escenario de uso diario:

Mostrar datos y ponerse en contacto con la persona dueña de la tarjeta.

Enrique tiene una página web en la que tiene subidas parte de sus obras para que el público general pueda contemplarlas. En las tarjetas que reparte en la entrada de las galerías en las que expone, al enfocarla con el móvil puedes observar un enlace a su página web, junto a un widget de una galería en la que aparecen sus obras justo al lado. También un teléfono y dirección de correo de contacto.

Modificar los datos de la tarjeta

Enrique quiere actualizar los datos que tiene publicados en su tarjeta para introducir la dirección de su próxima exposición. Desde la propia aplicación hay un menú en el que, tras iniciar sesión, puedes modificar las tarjetas que tengas vinculadas a tu cuenta.

Enrique busca la tarjeta que quiere actualizar de las que tiene registradas y a la derecha aparece una barra en la que puede seleccionar los componentes que desea añadir. Bajo esta barra hay un botón rojo para eliminarla. A la izquierda puede ver los componentes que ya tiene añadidos la tarjeta y cómo se distribuyen cuando enfoque a la tarjeta. Pulsa sobre el componente de texto que tiene con los datos de la galería y los modifica. Si guarda los cambios, se actualizarían en el servidor y ya estarían disponibles para todos los que tengan su tarjeta.

Desarrollo de prototipo

El desarrollo de este prototipo será también de tipo horizontal y de baja fidelidad, y su desarrollo lo realizaremos de nuevo con la herramienta Pencil Project.

En lo referente al diseño del prototipo y de la aplicación, surgieron varias dudas y problemas sobre cómo debería ser la interfaz de la aplicación y cómo debería de funcionar. Algunos de esos problemas fueron los siguientes:

- ¿Cómo añadir una tarjeta nueva? Es una parte compleja de solucionar. El prototipo está desarrollado de tal modo que “escanea” la tarjeta y a partir de ahí te permite añadirle información.
- ¿Cómo reconocerá la aplicación la tarjeta? Aquí surgen varias opciones. Una de ellas, la que más memoria cuesta, sería que la aplicación tuviera en memoria todas y cada una de las tarjetas para comprobar si el patrón de esta está en su base de datos. La segunda opción, sería que la aplicación reconociera que es una tarjeta, la captura y la manda al servidor, el cuál comprobaría la tarjeta con su base de datos y le devolvería como respuesta a la aplicación lo que debe mostrar o que no está en su base de datos.

Evolución del prototipo

Primera versión

Algunas imágenes de primer prototipo realizado con Pencil, sin haber tenido lugar las entrevistas aún; las podemos encontrar a continuación:

Ilustración 17 - Ejemplo pantalla principal prototipo 2

En esta primera ilustración del prototipo, podemos ver cómo sería la pantalla principal de la aplicación cuando se está enfocando a una tarjeta de visita. En ella hay vinculadas una página web, en la que se utiliza la metáfora del mundo conectado en la aplicación; un número de teléfono y un correo, con las correspondientes metáforas del altavoz de un teléfono y un sobre; y unas cuentas de Facebook, Twitter y LinkedIn. Además, en la parte superior derecha hay un botón con un diseño que incita a pensar que si lo pulsamos, encontraremos algo más.

Ilustración 18 - Captura del prototipo: "Crear nueva tarjeta"

Esta nueva imagen del prototipo correspondiente a la sección de crear una nueva tarjeta, podemos ver cómo la interfaz está dividida en dos partes: la parte izquierda en la que estamos viendo cómo va a quedar actualmente la información de la tarjeta con los datos virtuales que se le han añadido y una parte izquierda, en la que se encuentran las opciones disponibles para añadir información a la tarjeta.

Segunda versión

Se modifica el botón para abrir el menú lateral por otro cuya metáfora implique una barra de herramientas u opciones, ese botón pasa de albergar una imagen con tres puntos a tener una imagen con un engranaje con la que el usuario está más familiarizado.

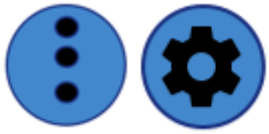

Ilustración 19 - Evolución Botón Opciones. Antiguo (izq.) y Nuevo (drcha.).

Además, dentro del menú lateral, se añade un nuevo botón para poder volver al paso anterior sin tener que cerrar y abrir la barra, teniendo que comenzar de nuevo los pasos realizados.

Ilustración 20 - Prototipo 2 con botón atrás

Se añaden también más imágenes a la vista de "detalles de la tarjeta" para que sea más intuitivo para el usuario de que esa es una tarjeta suya ya realizada y con los componentes de Realidad Aumentada ya configurados.

Por último, se añade también un botón para cerrar sesión, olvidado en la primera versión.

Tercera versión

La tercera versión del prototipo ya se ha realizado sobre un dispositivo real mediante Android Studio, al igual que la tercera versión del prototipo de la primera propuesta.

El diseño se ha visto alterado también del mismo modo que en la anterior propuesta de aplicación con un diseño en el centro de la pantalla del dispositivo y creciendo o decreciendo en función del contenido. Es posible que cambiando la distribución de la nueva actividad que hemos creado podamos ponerla a la derecha como estaba pensado en un principio, pero eso deben decidirlo los usuarios en la entrevista.

Para abrir este menú de interacción, tendremos que darle al botón azul con el engranaje como veremos en las capturas a continuación:

Ilustración 21 - Versión 3 de la propuesta 2. Menú sin iniciar sesión

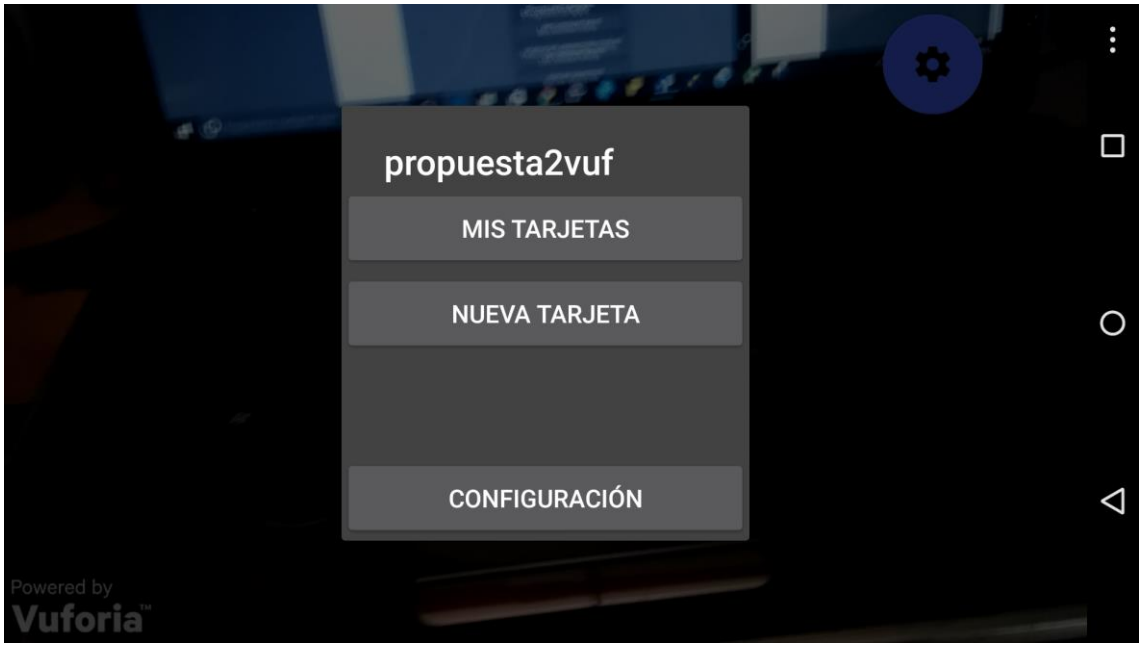

Ilustración 22 - Versión 3 de la propuesta 2. Menú con sesión iniciada

Ilustración 23 - Versión 3 de la propuesta 2. Menú de acciones de una tarjeta

Como se puede apreciar en la Ilustración 23 - Versión 3 de la propuesta 2. Menú de acciones de una tarjeta, el botón superior sale con un color grisáceo para que se note que es diferente a los demás, esto se ha obtenido de la cuarta versión del prototipo de la segunda propuesta de la aplicación tras las entrevistas.

Tareas a realizar en las entrevistas

A la hora de hacer un esquema de las tareas que deberán ejecutar los usuarios seleccionados para evaluar la interfaz y funcionalidad de la aplicación se ha utilizado el

software “Task Architect” [46] en su versión gratuita que nos permite las funciones que deseamos.

Las tareas que se han desarrollado para este prototipo son:

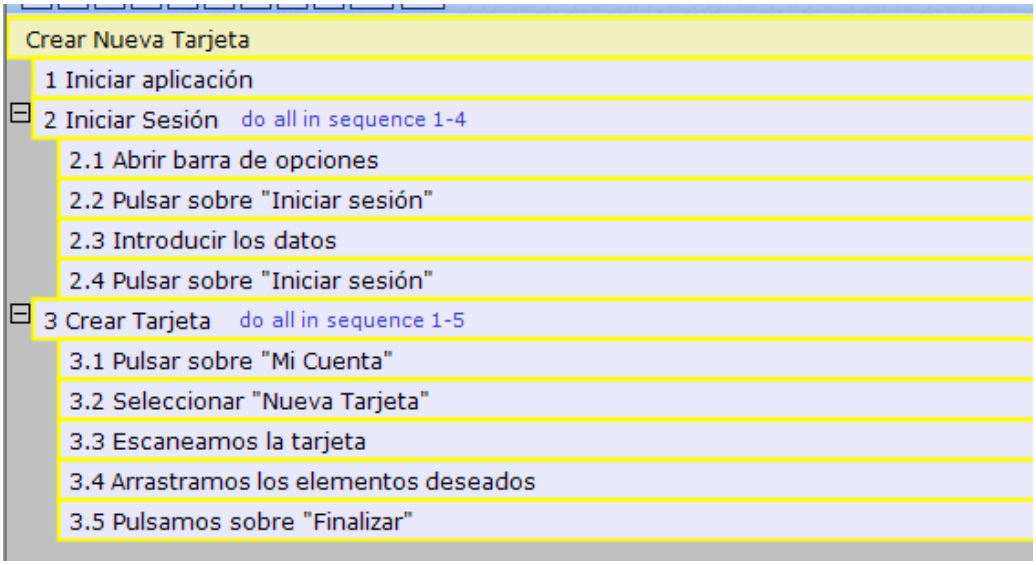

Ilustración 24 - Tarea de la aplicación 2 referente a "Crear Nueva Tarjeta"

En la tarea de la Ilustración 24 - Tarea de la aplicación 2 referente a "Crear Nueva Tarjeta", podemos ver el proceso para realizarla. Paso por paso, tendríamos que iniciar la aplicación. A continuación, iniciaremos sesión y para ello habría que hacer en secuencia los subpasos “abrir barra de opciones”, pulsar sobre “Iniciar Sesión”, introducir los datos del usuario y por último pulsar sobre el botón inferior “Iniciar Sesión”. Tras haber iniciado sesión, el siguiente paso sería crear una tarjeta, para ello tendríamos que seguir los subpasos siguientes y, ya que tenemos la barra de opciones abierta; pulsaríamos sobre “Mi Cuenta”, seleccionaríamos la opción de crear una nueva tarjeta y tras esto nos pediría ahora la aplicación que escaneemos la tarjeta que queremos añadir colocándola para que se vea bien y pulsando sobre el botón “Escanear”. En el siguiente paso tendríamos que arrastrar sobre la zona que queramos de la pantalla los elementos que queremos que se muestren con la Realidad Aumentada, y en función del que arrastremos nos indicará que le añadamos una información aparte. Por último, pulsamos sobre “Finalizar” y terminaríamos el proceso.

[46] http://taskarchitect.com/ – Último acceso: 30/08/2015

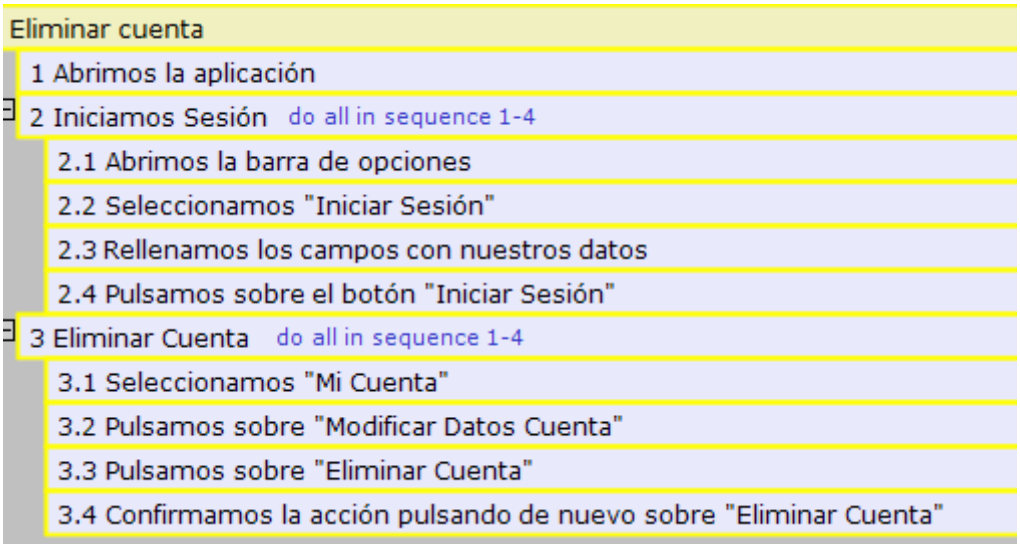

Ilustración 25 - Tarea de la aplicación 2 referente a "Eliminar Cuenta"

En la tarea de la Ilustración 25 - Tarea de la aplicación 2 referente a "Eliminar Cuenta", vemos que los pasos que hay que seguir son muy simples. Para comenzar abriríamos la aplicación, iniciaríamos sesión abriendo primero la barra de opciones y seleccionando "Iniciar Sesión", iniciaríamos los datos del usuario y por último pulsaríamos sobre el botón inferior "Iniciar Sesión". Y para borrar la cuenta, pulsaríamos sobre "Mi Cuenta" aprovechando que ya tenemos la barra de opciones abierta, seguiríamos con "Modificar Datos de la Cuenta", seleccionaríamos "Eliminar Cuenta" y confirmaríamos la acción pulsando de nuevo sobre "Eliminar Cuenta". Con esto, ya tendríamos todos nuestros datos borrados.

Resultados de entrevistas para validar el diseño del prototipo

A la hora de realizar las entrevistas, hemos seguido la misma metodología que en la propuesta anterior. Se han utilizado los métodos "Thinking Aloud", pensando en voz alta; y "Coaching Method", método del conductor; para obtener información útil del usuario que estaba siendo evaluado.

Antes de comenzar con las evaluaciones del prototipo por parte de los usuarios, se les ha leído también el guión de orientación, "The Orientation Script"; para que supieran las tareas que debían realizar, estuvieran más tranquilos y así que pudieran realizar la evaluación de una manera más fiel sabiendo que eran las personas "adecuadas" para ello. Además, también se les comentó que estaríamos allí en todo momento tomando algunas notas sobre las acciones que iban realizando y lo que iban diciendo en voz alta y que estaríamos disponibles para cualquier duda que pudieran tener.

Primera versión

Entrevista 1

El usuario entrevistado para realizar la primera evaluación de prototipo ha sido el usuario secundario “Enrique Meléndez” el cuál no ha tenido ningún problema para realizar ninguna de las dos tareas. No ha dudado a la hora de seleccionar una opción u otra.

Como apunte, solo ha comentado la posibilidad de cambiar el icono de opciones por algo más cómodo para el usuario y que pueda entenderlo antes, como un engranaje, ya que es más habitual. También como apunte para la tarea de “Eliminar Cuenta” ha comentado que debería solicitarse una confirmación de los datos a la hora de borrar una cuenta, la cual está hecha, pero había un error con los enlaces del prototipo en Pencil.

Segunda versión

Entrevista 1

El usuario entrevistado para realizar la evaluación del segundo prototipo ha sido el usuario primario “Juan Rodríguez” el cuál ha realizado las diferentes tareas sin haber tenido ningún problema ni duda que haya comentado a la hora de seleccionar una opción u otra al realizarlas.

Tercera versión

Entrevista 1

El usuario con el cuál se ha llevado a cabo la evaluación del nuevo prototipo en el dispositivo real, no tuvo problemas a la hora de realizar las tareas impuestas. Comentó que le gustaba el nuevo diseño y que no tenía problemas en que estuviera en el centro, así directamente lo tenía delante de los ojos y no podía distraerse. Además le gustó que algunos botones tuvieran otra tonalidad para diferenciar las funcionalidades de cada uno.

Entrevista 2

El usuario mostró también su satisfacción por el nuevo diseño de la aplicación, comentó también que aparte del diseño le gustaba que al pulsar en cualquier parte fuera del menú de opciones este se cerrara.

Pulsó algunos botones para probar su funcionalidad como el de crear una nueva tarjeta, pero no está implementada su funcionalidad. Quitando este hecho, no tuvo problemas a la hora de realizar una correcta evaluación del prototipo sin perderse en ningún momento del recorrido lógico de interacción.

Desarrollo de las aplicaciones y comparativa de los SDK seleccionados

En este apartado expondremos explicaciones acerca del desarrollo de las aplicaciones con los recursos AR en las diferentes herramientas que nos proporcionan las empresas dueñas del SDK y cómo se exporta su código para utilizarlo en Android Studio y modificarlo.

Desarrollo primera aplicación – Aplicación de apoyo al estudio

Preguntas previas

Antes de comenzar el desarrollo de la aplicación, nos hemos encontrado con diferentes problemas, entre los que podemos destacar:

- Actualizar información de la aplicación sin tener que recompilar la aplicación. Esta es una tarea que consideramos necesaria de cara al buen funcionamiento de la aplicación y el poder actualizar su información sin tener que actualizar la aplicación, ya que así está planteada desde el principio.
- Cuando se descargue la información del servidor, ¿pedimos los datos como hace "Spotify" o nos descargamos todas las relacionadas?
- ¿Cómo hacer los modelos 3D para mostrarlos a una diapositiva concreta? Este inconveniente lo solucionamos con la aplicación "Blender"[47] ya que es una de las mejores herramientas del mercado, además de ser gratuita es compatible con todos los Sistemas Operativos de ordenador y de código abierto. Además de esa herramienta, para los modelos 3D hemos utilizado también "SketchUp"[48] que es bastante más simple y para los modelos que queremos realizar son más que suficientes y por último el propio Unity3D que también nos permite hacer modelos.

Desarrollo en Unity 3D con Vuforia

Para descargar Unity, nos vamos a su página oficial[49], en la parte superior le damos a "Obtener Unity" y seleccionamos la descarga gratuita. Nosotros trabajaremos con la versión 4.6.5 ya que empezamos con la 4.6.4 y algunos de los SDKs aún no habían sido probados para la versión 5 de Unity, la versión 4 la podemos descargar del archivo de Unity[50].

[47] https://www.blender.org/ – Último acceso: 30/08/2015
[48] http://www.sketchup.com/es – Último acceso: 30/08/2015
[49] https://unity3d.com/ – Último acceso: 30/08/2015
[50] https://unity3d.com/es/get-unity/download/archive – Último acceso: 30/08/2015

La instalación de Unity es muy simple, con abrir el archivo que nos hemos descargado y seleccionar la ruta de instalación, ya haría todo el proceso él automáticamente.

Para instalar Vuforia, nos vamos a su página web oficial y a la sección de descargas[51] y veremos los SDKs de Android, iOS y el plugin de Unity. Para poder descargarlos, hay que iniciar sesión, por lo que deberíamos crearnos una si no disponemos ya de ella. Una vez nos hayamos descargado el plugin y el SDK de Android por si acaso lo utilizamos, la instalación se hará sobre un proyecto ya abierto en Unity como el que vemos a continuación.

Una vez tenemos instalada la herramienta, no es muy compleja de utilizar. La primera pantalla que nos saldrá será la de creación de un proyecto nuevo y los paquetes que queremos que tenga, como podemos ver a continuación:

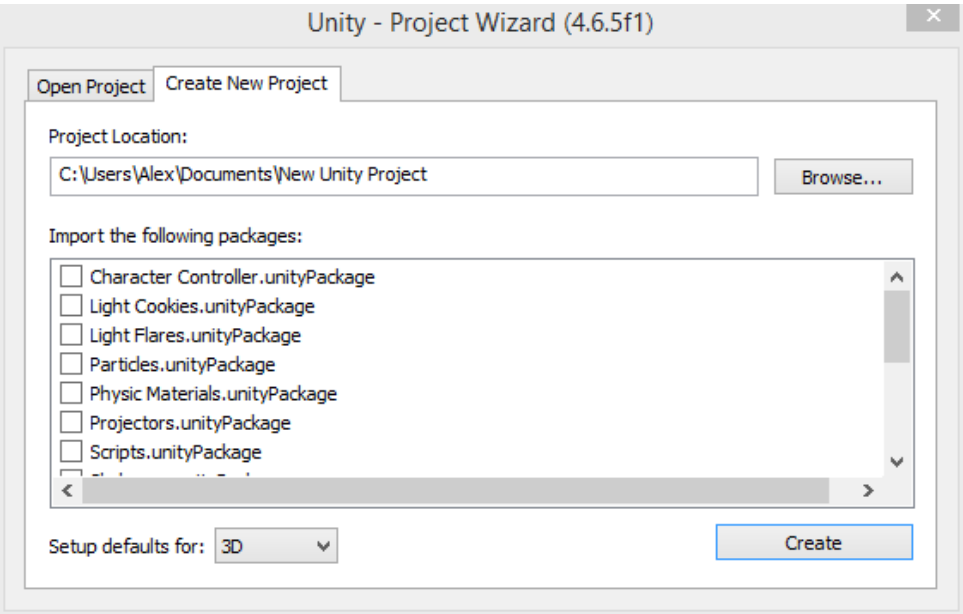

Ilustración 26 - Nuevo Proyecto Unity

Una vez creado el proyecto, nos dirigimos al plugin que nos hemos descargado de la web de Vuforia en su versión de Unity y con hacer doble clic automáticamente nos dirá Unity para importarlo al proyecto actual.

Una vez importado el paquete, ya podemos utilizar Vuforia dentro de Unity, para ello simplemente nos vamos a la vista de "Project" y abrimos la carpeta de "Qualcomm Augmented Reality", "Prefabs" y dentro nos encontraremos con los recursos de Vuforia, como la cámara de AR llamada "ARCamera" que es lo primero que debemos añadir a la vista de jerarquía. También añadiremos un "ImageTarget" sobre la misma vista.

[51] https://developer.vuforia.com/downloads/sdk – Último acceso: 30/08/2015

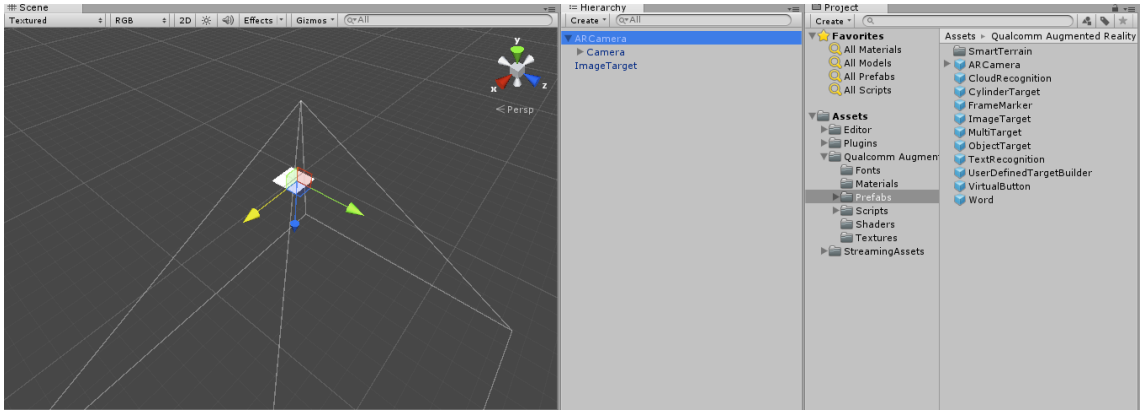

Ilustración 27 - Inicio Unity y Vuforia

Para continuar, tendremos que añadir la "Key" o clave de la licencia del SDK de Vuforia y crearnos los "Image Targets" en la web. Para obtener la "key", nos vamos a la web de "Vuforia Developers"[52] y accedemos. Ahora nos iremos a la pestaña de "Develop" y en "License Manager" crearemos una nueva licencia.

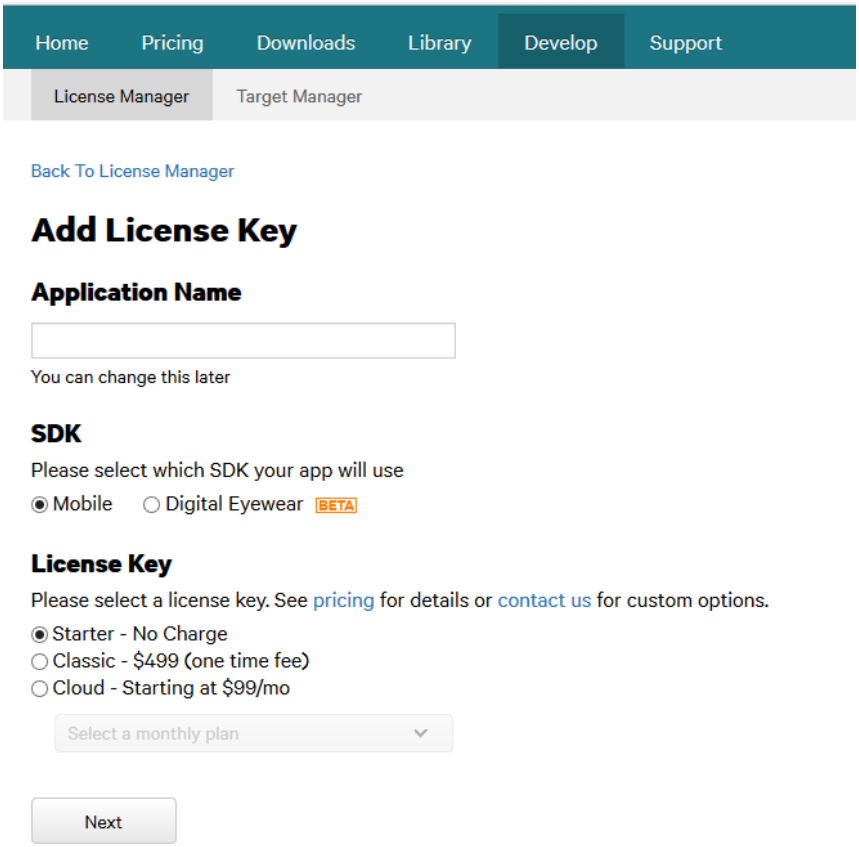

Ilustración 28 - Nueva Clave de Licencia Vuforia

A continuación, tendremos nuestra "Key License" y simplemente tendremos que introducirla en Unity, teniendo seleccionada la "ARCamera" que hemos añadido antes y

[52] https://developer.vuforia.com/ – Último acceso: 30/08/2015

dentro de la vista “Inspector” veremos que hay un cuadro de texto dentro de “QCARBehaviour (Script)” en el que hay que pegarla:

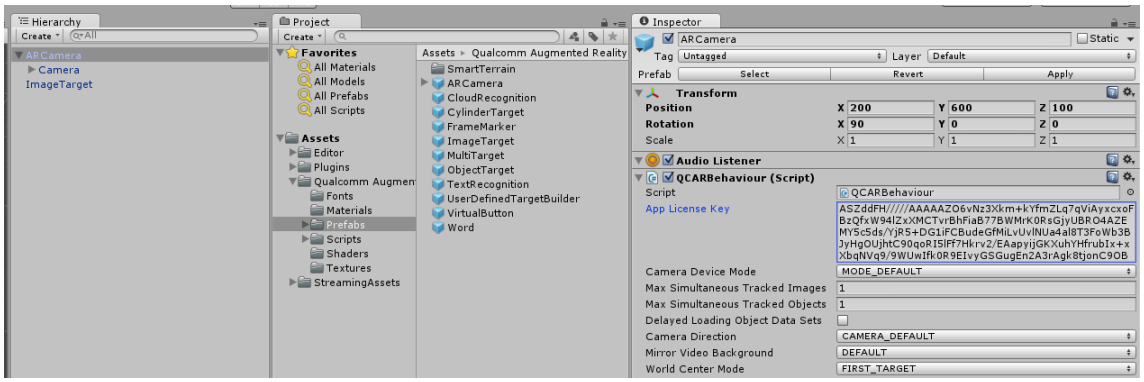

Ilustración 29 - Key License en Unity

Ahora, para que la aplicación pueda reconocer patrones, se los tenemos que añadir. Para ello, de nuevo en la web anterior, hacemos el mismo recorrido pero ahora en vez de irnos a “License Manager” nos iremos a “Target Manager”.

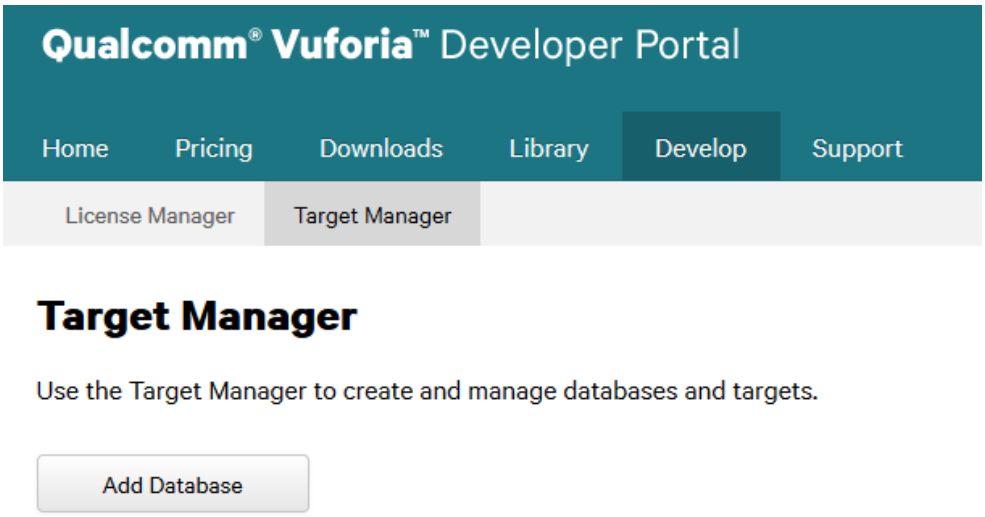

Ilustración 30 - Usar Target Manager de Vuforia

Crearemos una nueva base de datos y cuando nos de la opción seleccionaremos que por ahora queremos que esté en el dispositivo, no en el Cloud. Una vez creada, accedemos a ella y añadiremos un nuevo “Target”. Nos dará varias opciones como son:

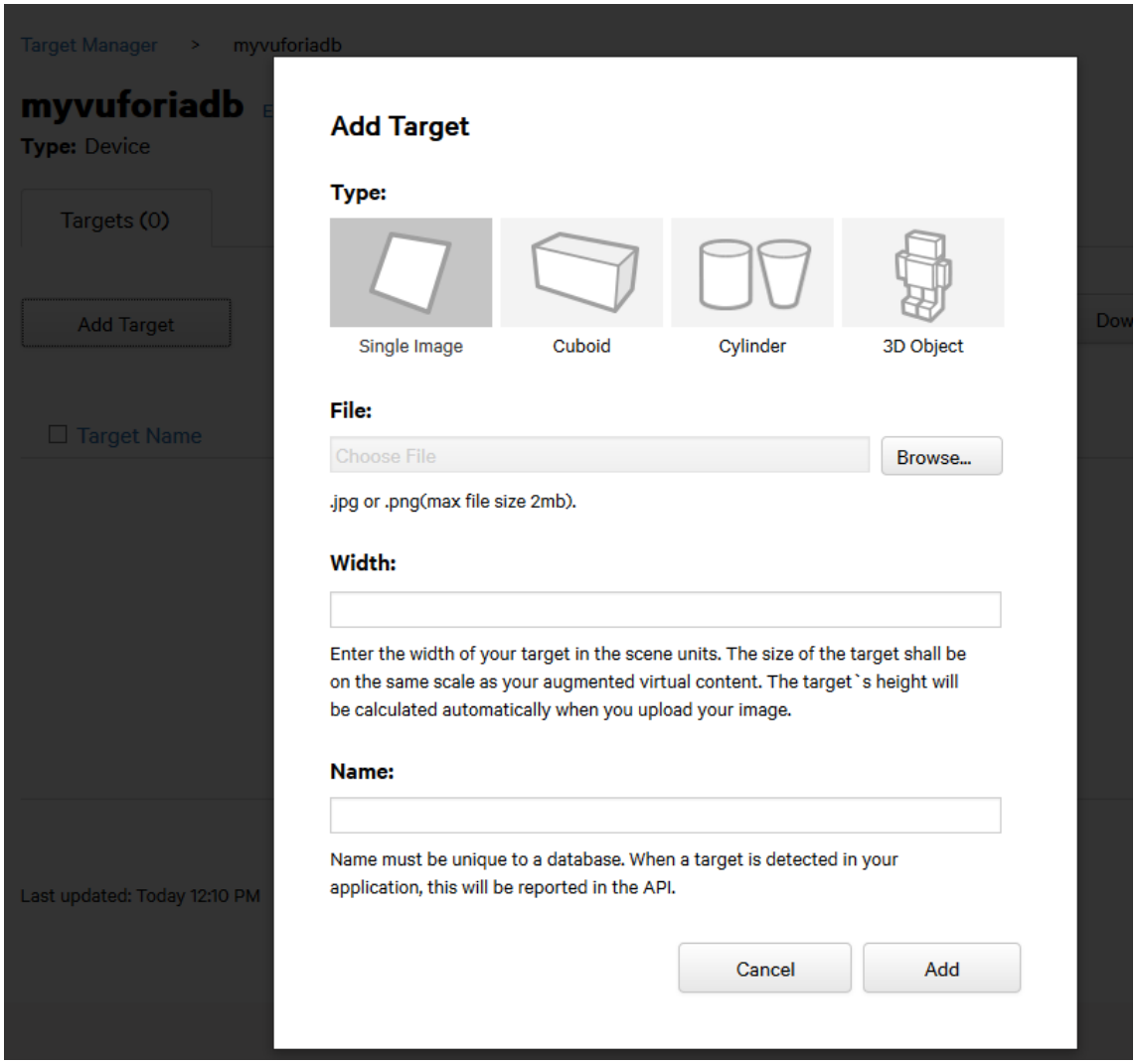

Ilustración 31 - Tipos de Target en Vuforia

Entre los tipos de "Targets" soportados por Vuforia observamos:

- **Simple Image**: el más común, ya que es una imagen que tiene que reconocer y si lo hace te mostrará los recursos AR que tenga asociados a esa imagen. Solo tendríamos que añadir una imagen y darle su anchura y ya estaría creado.
- **Cuboid**: es un objeto en forma de cubo que puede reconocer si le añadimos las caras una vez creado. Al seleccionarlo sólo hay que añadirle el tamaño. Una vez hecho al seleccionarlo te pedirá las imágenes que forman las caras.

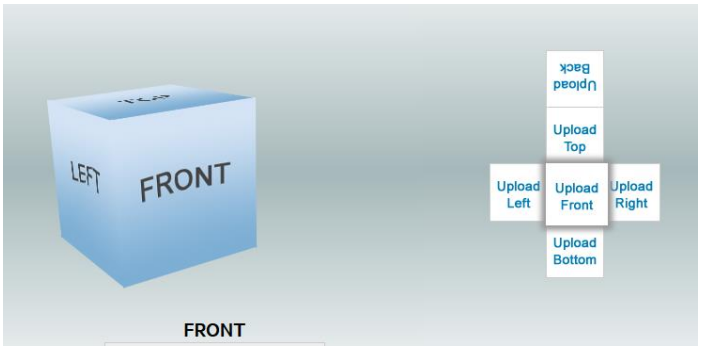

Ilustración 32 - Cuboid en Vuforia

- **Cylinder**: es un objeto con forma de cilindro que podrá reconocer si le añadimos las imágenes que lo forman después de haberlo creado.

Ilustración 33 - Cylinder en Vuforia

- 3D Object: es un objeto 3D que puede reconocer Vuforia. Para añadirlo hay que utilizar la herramienta "Vuforia Object Scanner" y añadir luego el archivo que nos cree. Podemos reconocer objetos que tienen formas, como un juguete y nos mostraría el contenido AR que nosotros le asociáramos, como en los demás tipos.

De entre todos los tipos posibles, nosotros seleccionaremos "Single Image". Una vez lo hayamos creado, veremos que nos da una clasificación con estrellas en función de las diferencias que tenga la imagen que hemos añadido para que la reconozca sin problemas. Para descargar ese target solo, la seleccionaremos y le daremos a "Download Dataset" y si no la seleccionamos nos descargará todos los targets que tenga ahora mismo la base de datos.

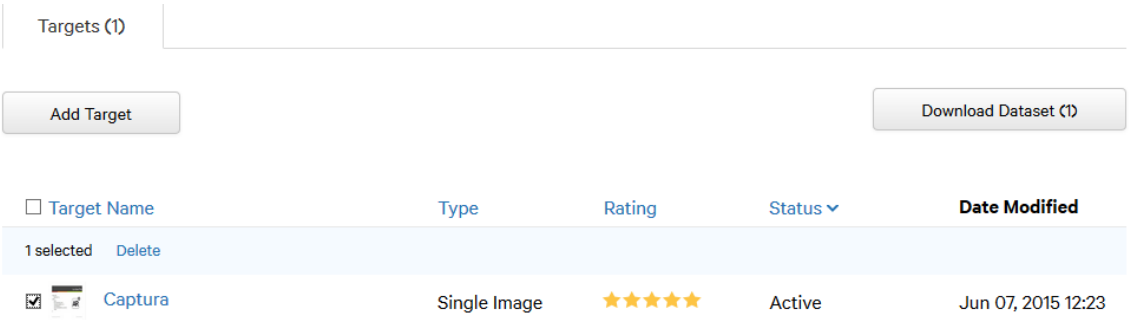

Ilustración 34 - Single Image Target en Vuforia

Cuando le demos a descargar, nos dará la opción de descargar para un proyecto Eclipse, Ant, Xcode o para Unity Editor. Nosotros ahora seleccionaremos Unity y lo añadiremos a nuestro proyecto actual. Para ello, simplemente ejecutaremos el archivo que nos acabamos de descargar de nuestra que tiene el mismo nombre que el "Target Manager" que creamos antes. Y se nos añadirán automáticamente los datos de configuración en la carpeta "StreamingAssets" y la imagen que adjuntaremos después a nuestro "ImageTarget" se añade dentro de la carpeta "Editor". Todo eso lo podemos ver a continuación más claramente:

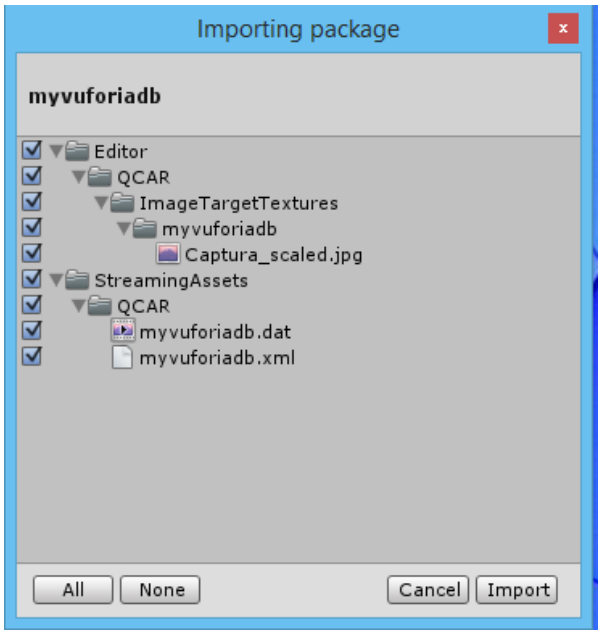

Ilustración 35 - Importar Target Manager a Unity

Para que Vuforia pueda reconocer el "Target" que añadimos antes, tendremos que arrastrar la imagen de "Captura_scaled.jpg" sobre el ImageTarget que añadimos antes a nuestra jerarquía, si no nos saliera en el sentido que queremos la imagen, tan sólo tendríamos que seleccionar el "Image Target" y en la vista de "Inspector" ir modificando los valores de la rotación hasta encontrar el que deseamos. Tras eso, para que el "Image Target" sepa de dónde tiene que sacar todos los datos del "Target" que

hemos añadido, se lo tendremos que indicar, para ello teniéndolo seleccionado nos vamos a la vista de "Inspector" y dentro de "Image Target Behaviour (Script)" rellenamos los datos del "Data Set" y del "Image Target".

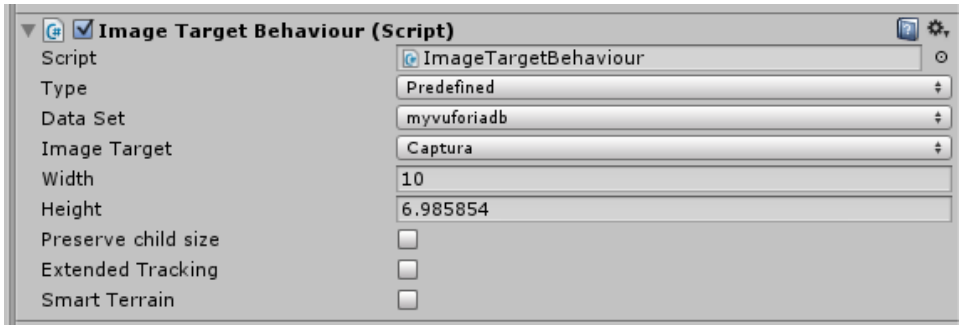

Ilustración 36 - Configuración de Image Target y Data Set

También debemos decirle a la cámara, el "Data Set" que debe cargar, para ello el procedimiento es el mismo, la seleccionamos y dentro de "Data Set Load Behaviour (Script)" activamos las casillas de los que queremos que muestre.

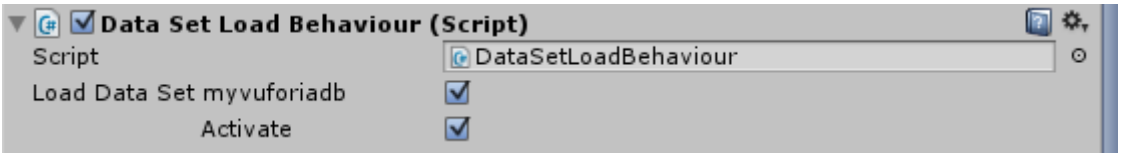

Ilustración 37 - Configuración de ARCamera y Data Set

Ahora tendremos que añadir los recursos de AR que queremos que se muestren, para ello podemos utilizar los propios de Unity bajo la pestaña de "GameObject" en la barra superior o los modelos que hemos realizado en SketchUp y Blender, que son los que utilizaremos. Para ello y para mantener un orden, crearemos una nueva carpeta llamada "Models" bajo la carpeta de "Assets" simplemente con el botón derecho. Dentro de ella crearemos una para cada uno de los programas Blender y SketchUp y añadiremos dentro de cada una sus modelos para no equivocarlos arrastrándolos desde el explorador de archivos de nuestro equipo.

Una vez los tenemos ya incluidos en el proyecto, los arrastramos sobre el nuestro "ImageTarget" y los colocamos como queremos bien moviéndolos con las flechas que salen sobre él o bien poniendo las coordenadas a mano. Es posible que al importarlos, se pierda el "Material" o "Textura" de los mismos, por lo que tendremos que incluir la imagen que habíamos usado dentro de la carpeta y luego arrastrarla sobre el cubo o sobre las caras del cubo dependiendo del modelo que sea.

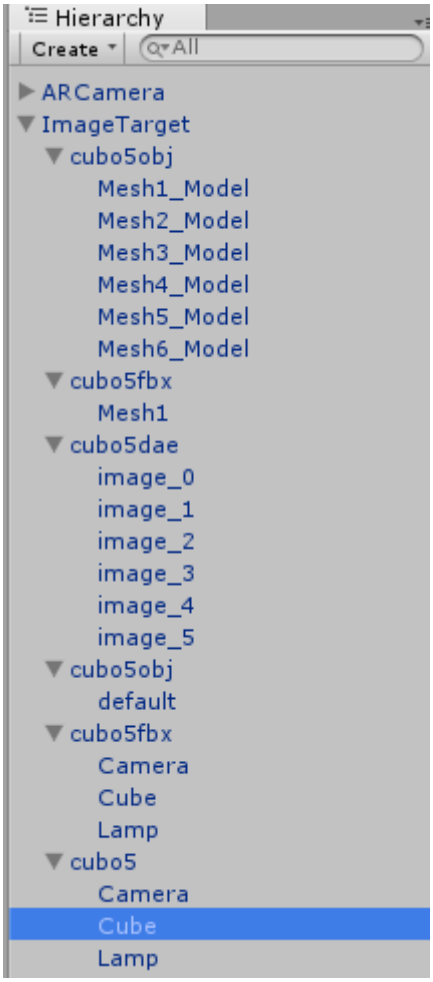

Ilustración 38 - Modelos añadidos en Unity con Vuforia

Como vemos en la imagen superior, algunos modelos directamente salen un cubo y en otros salen las 6 caras del mismo. Tal y como se muestran en la imagen, nosotros tenemos que los 3 primeros cubos pertenecen a los creados con “SketchUp” y se han colocado en la parte derecha de la “ImageTarget” y los 3 siguientes pertenecen a los creados con “Blender” que se encuentran en la parte izquierda como podemos ver aquí

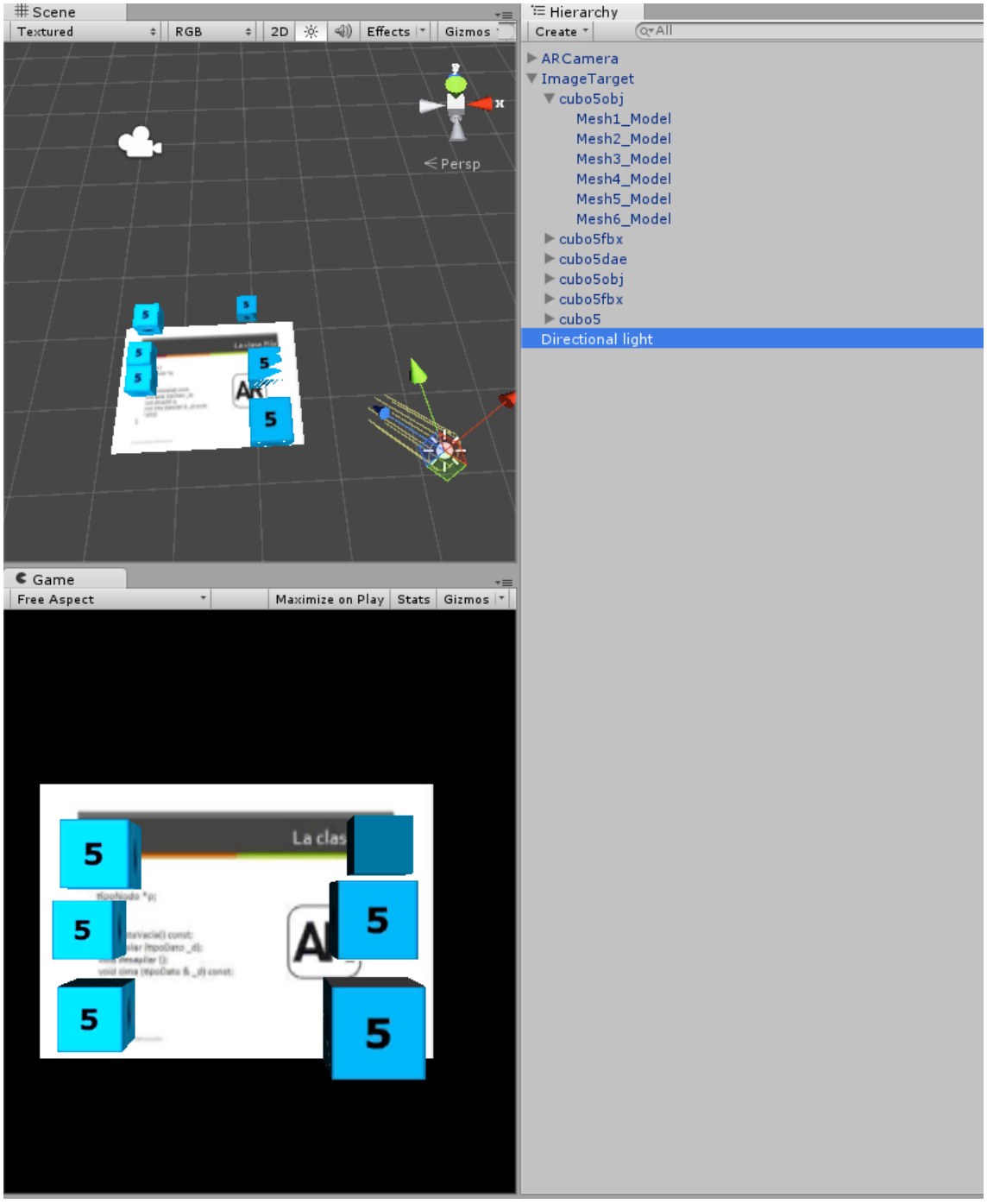

Ilustración 39 - Resultado primera aplicación con Vuforia en Unity

Como podemos ver, también hemos añadido unas cuantas luces para que se pudieran ver los modelos, si no estarían oscuros. Para añadirlas, es necesario crearlas desde la parte superior en "GameObject", seleccionar "Light". En el ejemplo, nosotros hemos añadido una "Directional Light" apuntando a los modelos.

Ahora, para comprobarlo en nuestro dispositivo móvil o exportarlo como proyecto Android, nos iríamos a "File" y "Build Settings", seleccionamos "Android", pulsamos

sobre el botón “Add Current” para que añada nuestra escena actual al proyecto que queremos exportar; y le damos a “Player Settings”.

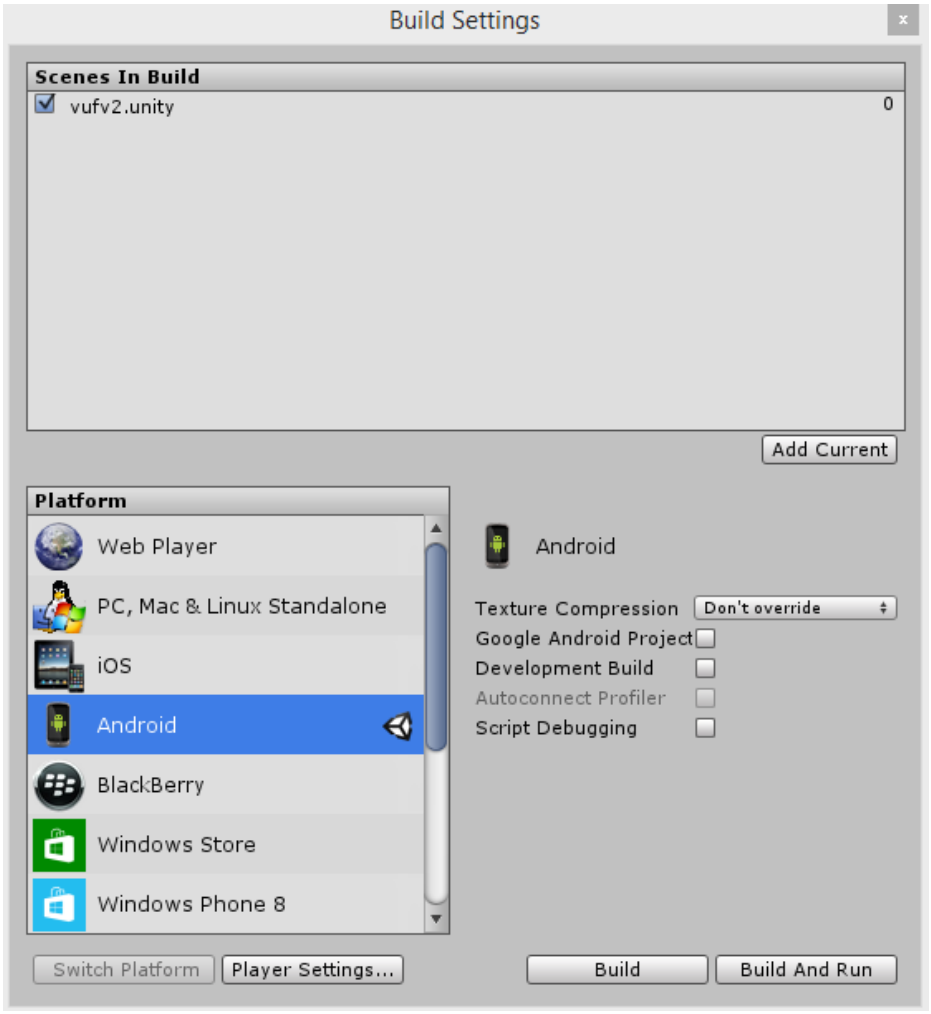

Ilustración 40 - Exportar proyecto a Android

Para que nos salga el icono de Unity en Android, tendremos que pulsar sobre “Switch Platform” ya que por defecto Unity está configurado para exportar el proyecto y aplicación para el propio PC.

Ya en Player Settings, nos interesa la parte de la vista del “Inspector” en “Other Settings” y concretamente “Bundle Identifier*” tendremos que poner el nombre del id del paquete que lo identifica y en la parte superior modificamos el nombre de nuestra compañía para poner el que queramos. También podemos cambiar otros campos como el nombre de la aplicación.

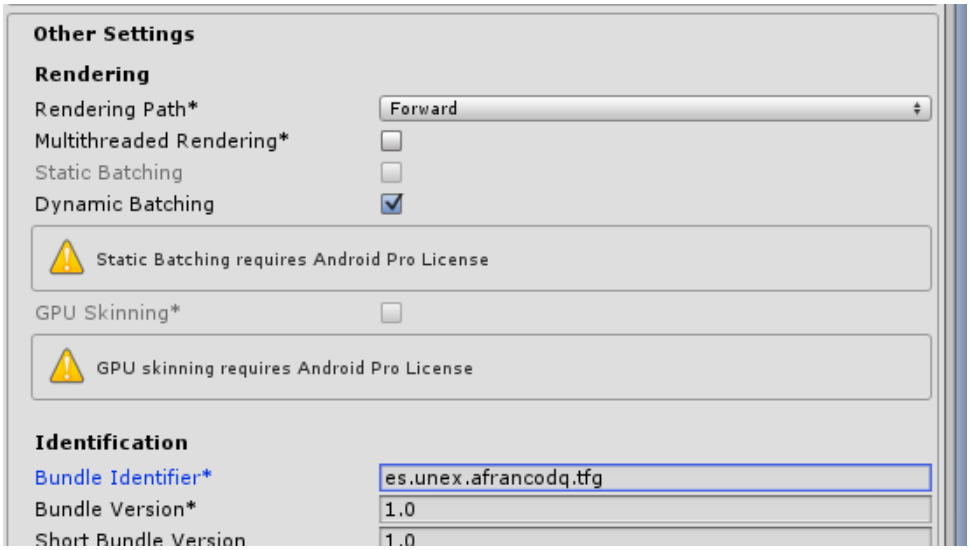

*Ilustración 41 - Modificando el Bundle Identifier**

Una vez hemos configurado lo anterior (la "Key License" añadida a la "ARCamera" y la "ImageTarget" añadidas con sus modelos) podemos exportar el proyecto sin problema. Para ello dentro de "Build Settings" de nuevo, podemos o exportar el proyecto a Eclipse ADT o Android Studio con marcar la casilla, con lo que no podremos crear el archivo instalable desde Unity, pero sí podremos en Android Studio.

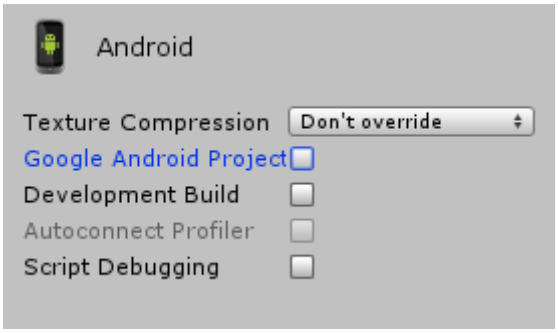

Ilustración 42 - Exportar Android Project

Si queremos instalarla directamente en el dispositivo, basta que con seleccionar "Build and Run" mientras tengamos un dispositivo conectado[53].

Para hacer las pruebas más rápidas, podemos darle al botón "Play" directamente arriba y nos ahorramos el tener que compilarlo todo y crear la aplicación para el móvil. Mientras está en ejecución nos permite modificar cualquier campo, pero cuando paremos la ejecución, todo volverá a su estado anterior.

[53] Damos por hecho con los drivers o controladores del teléfono están instalados correctamente y el SDK de Android también.

Por último, al probar la aplicación en un dispositivo Android real para comprobar si funcionaban los modelos correctamente y ver la velocidad de Vuforia al reconocer un Target y lo estable que es obtenemos las siguientes imágenes:

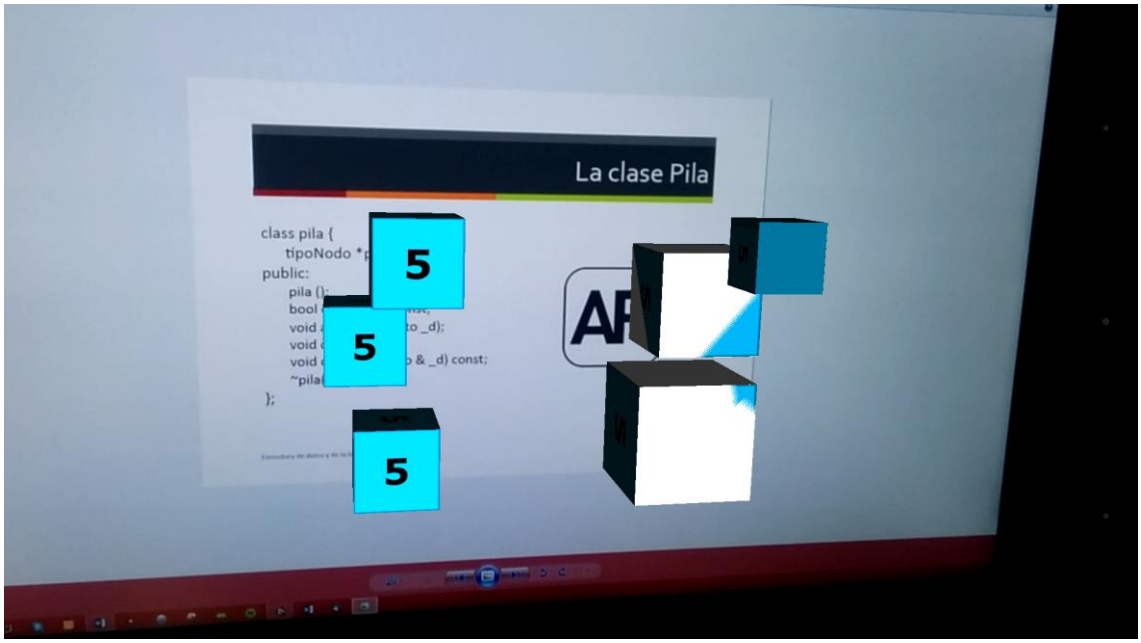

Ilustración 43 - Vuforia en dispositivo Android real 1

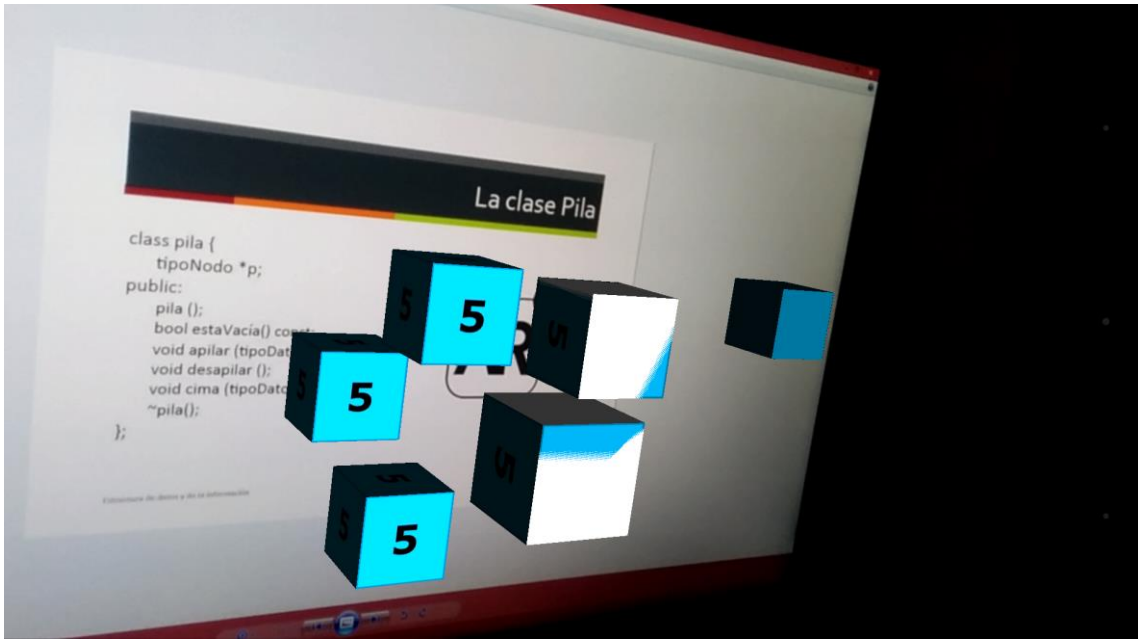

Ilustración 44 - Vuforia en dispositivo Android real 2

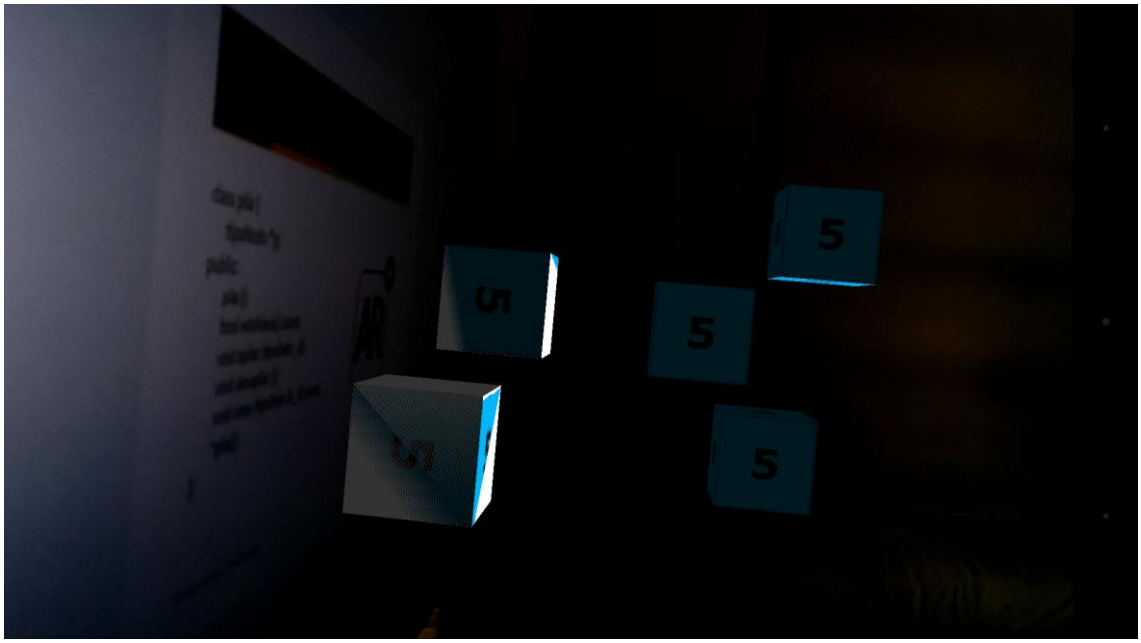

Ilustración 45 - Vuforia en dispositivo Android real 3

En la primera imagen podemos ver que hay 6 recursos AR colocados en 2 columnas. La columna de la izquierda presenta los objetos que han sido creados y exportados en "Blender", mientras que la columna de la derecha son de "SketchUp". Además, cada fila se corresponde también con el mismo formato exportado por ambos programas, siendo la primera fila los exportados en formato "dae", la segunda "fbx" y la tercera "obj".

Como podemos ver también en la primera de las imágenes, los que han sido creados en "Blender" no dan error de ningún tipo, mientras los que se crearon con "SketchUp" o no se ve bien la textura o material asociado, o la luz da problemas porque se refleja y quedan blancos, por lo que no nos sirven por problemas de compatibilidad.

En la segunda imagen, observamos exactamente lo mismo, pero introduciendo un nuevo ángulo de cámara para comprobar que no era problema de ese punto concreto.

Por último, en la tercera de las imágenes, podemos observar que el ángulo de visión de Vuforia es bastante amplio, ya que casi no se ve el "Target" y los recursos AR se siguen mostrando en 3D, aunque algunos más adelantados que otros.

Múltiples Targets

Para permitir que la aplicación pueda reconocer varios "targets" u objetivos, tendríamos que repetir el proceso de subir las imágenes que queramos en el formato que deseemos al servidor de Vuforia como vimos anteriormente, en función de que el "target" deseado sea una imagen o un objeto.

Una vez tenemos completado ese paso y descargado la nueva base de datos de los targets, bastará con que la importamos de nuevo a Unity 3D y añadamos un nuevo "ImageTarget" a nuestra escena, junto al anterior que ya teníamos. Posteriormente lo configuramos con la nueva imagen que hemos añadido, como podemos ver a continuación en la siguiente imagen:

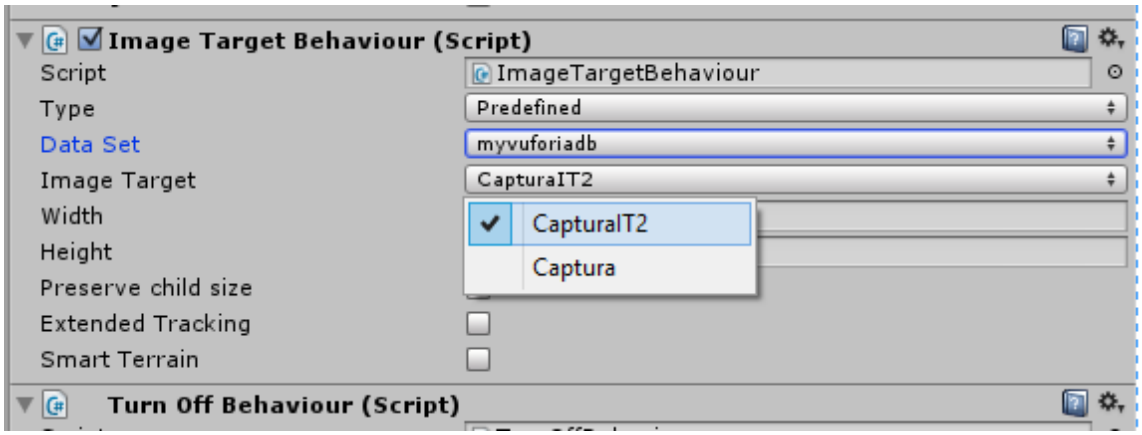

Ilustración 46 - Configurando un segundo "target" en Vuforia

Tras haberlo añadido y configurado con la información de la segunda imagen que queremos que pueda ser reconocida, le añadiremos los recursos AR que deseemos y ya funcionará correctamente. No es necesario que ese "ImageTarget" que acabamos de añadir esté ubicado bajo la cámara de Vuforia; es suficiente con que se encuentre en la misma escena, como vemos en la siguiente imagen:

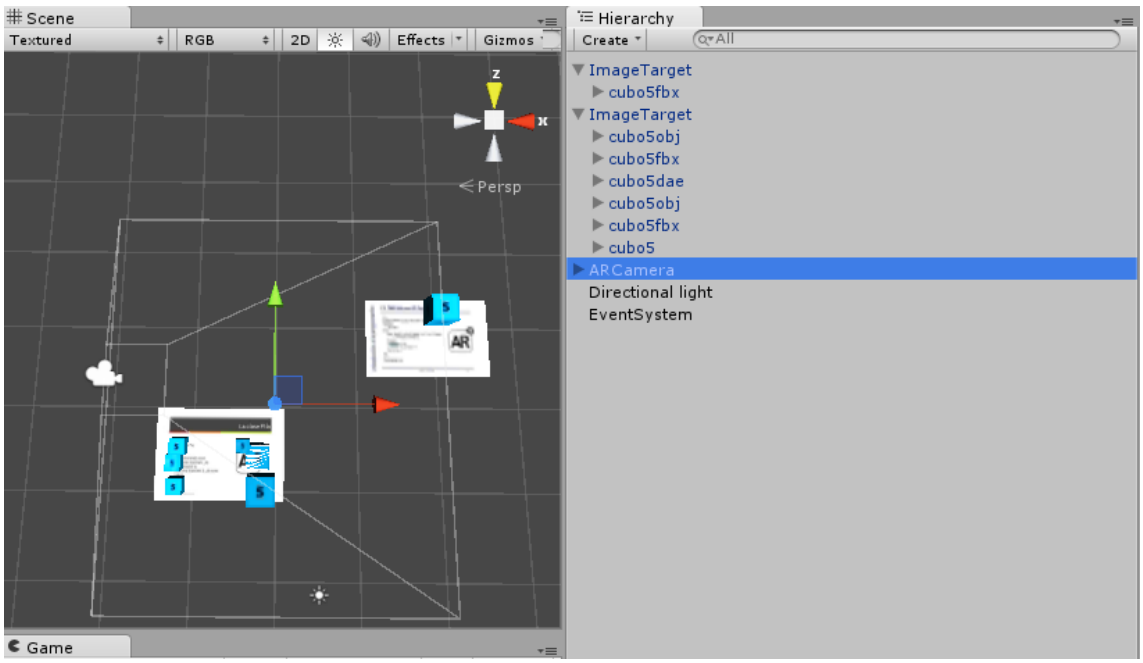

Ilustración 47 - Escena de la propuesta dos en Unity 3D y Vuforia con dos "targets"

Además, existe la posibilidad de que deseemos que puedan reconocerse dos "targets" a la vez (aunque para nuestra propuesta no nos resulte demasiado útil). Para poder llevar esto a cabo, bastaría con seleccionar la cámara de Vuforia, "ARCamera" y buscar en la vista del inspector el script "QCARBehaviour", el mismo al que le añadimos al principio la clave de licencia de Vuforia, e indicarle el número máximo de imágenes que puede rastrear, que por defecto se encuentra con un valor de 1. Con la siguiente imagen podremos ver exactamente dónde se encuentra esa información:

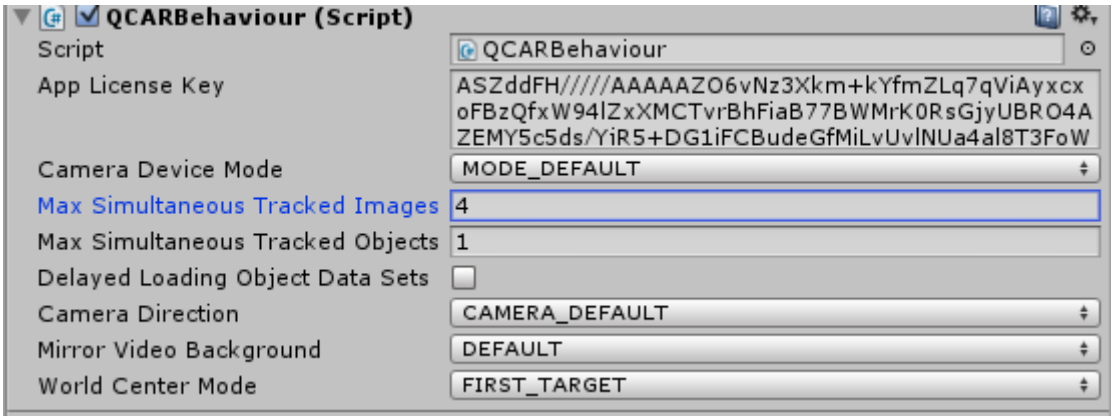

Ilustración 48 - Rastreo de múltiples imágenes a la vez en Vuforia y Unity 3D

Animaciones en Unity

Las animaciones o "Animation" en Unity, es una de las funciones que incluye Unity por defecto y que se puede utilizar con la versión gratuita de la herramienta.

Para ello basta con que seleccionemos un objeto, en nuestro caso haremos el ejemplo con un proyecto diferente en el cuál tenemos varios objetos cubo simulando los datos que vamos a insertar en una pila y textos que irán explicando el proceso en la pantalla del dispositivo:

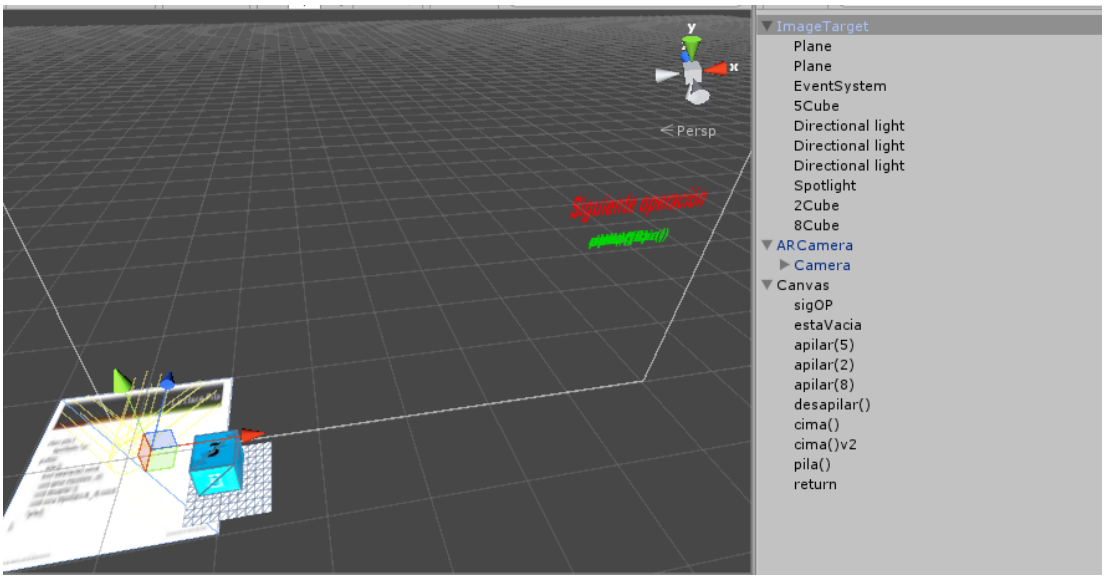

Ilustración 49 - Ejemplo Vuforia y Unity para añadir animaciones con "Animation"

Seleccionamos uno de los objetos, como por ejemplo el “5Cube” que desciende de nuestro “ImageTarget”, y, en la barra de herramientas pulsamos sobre “Window” y posteriormente en “Animation”. Al pulsarla, se nos abrirá una nueva ventana en la cual se nos pedirá añadir una nueva curva de animación. La crearemos y guardaremos en memoria para poder configurarla. Cuando lo hayamos hecho seleccionaremos “Position”, ya que ahora nos interesa mover el cubo y pulsaremos sobre el signo “+” que hay a su derecha.

También hemos añadido una nueva curva para modificar su tamaño o escala, para que no se vea en el modelo hasta que nosotros se lo pongamos.

Una vez hayamos realizado esos pasos, tendríamos algo similar a lo que aparece en la siguiente imagen que adjuntamos:

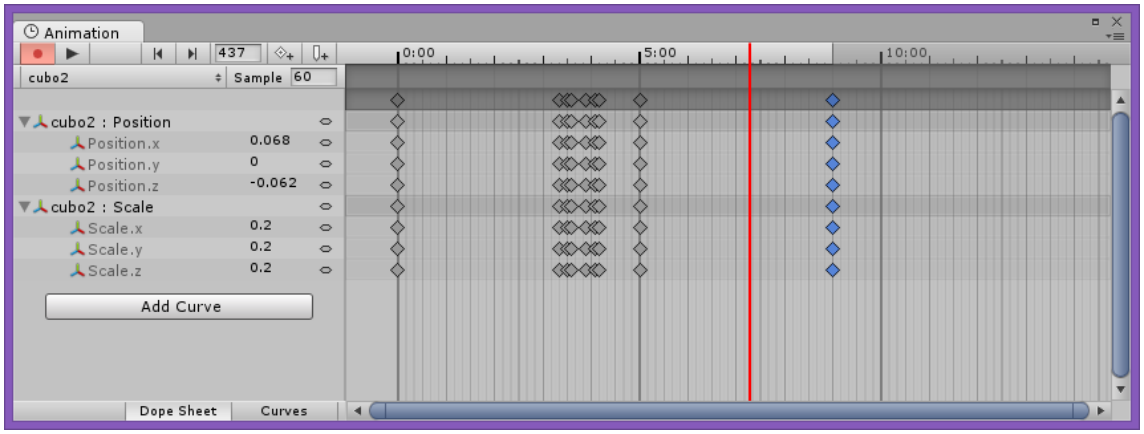

Ilustración 50 - Ejemplo animación en Unity propuesta de aplicación 2

Para comprobar cómo está quedando la animación que estamos creando, tendremos que pulsar el botón “play” que se encuentra en la parte superior izquierda de la imagen anterior y lo veremos en las vistas de escena y “game” de Unity en tiempo real. Podemos ver a continuación una muestra del estado de las animaciones del proyecto antes de ser terminadas:

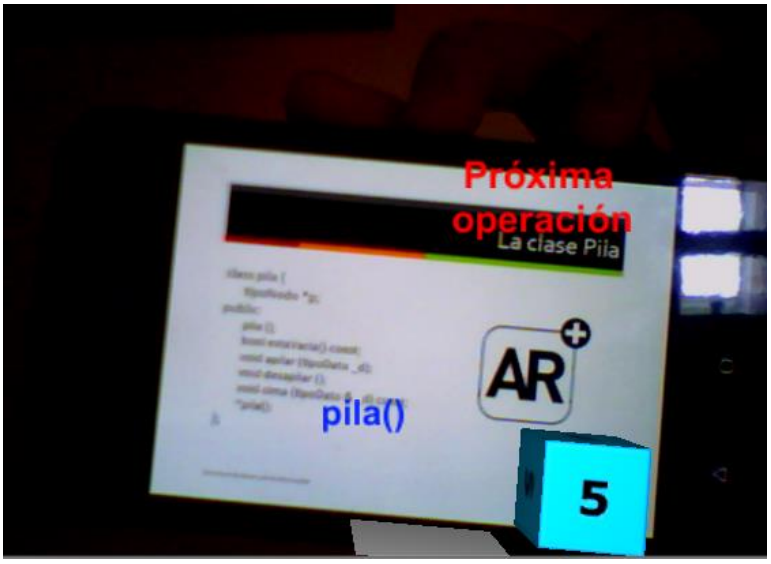

Ilustración 51 - Prueba de animación de la propuesta 2 desde Unity en PC

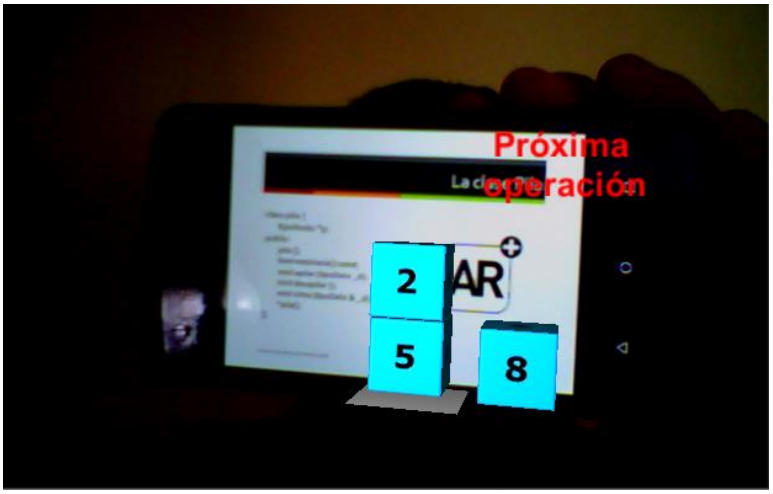

Ilustración 52 - Prueba de animación de la propuesta 2 desde Unity en PC

Como podemos ver en las imágenes, las letras que hemos añadido no se reconocen en el movimiento de la cámara al igual que los recursos AR que tenemos añadidos, las letras se ven siempre de frente, lo cual en un primer momento podemos considerar como un fallo en su funcionamiento.

Principales problemas encontrados

Los principales problemas que nos hemos encontrado a la hora de trabajar con Vuforia y Unity 3D han sido los siguientes:

- Mostrar texto. La forma en la que lo mostramos actualmente, tiene problemas de compatibilidad con Android desde Unity, ya que en Unity en su vista previa en el equipo funciona bien, pero en Android, al ejecutarlo, se ven los recursos AR con sus animaciones pero no el texto.

- Las animaciones de los recursos. Hemos tenido varios problemas con ellas para que quedaran bien al final tanto haciendo las animaciones mediante código en los scripts de Unity como también contando con la ayuda del plugin “iTween” pero había problemas con las coordenadas de los recursos, ya que en la jerarquía los recursos AR son hijos de los “ImageTargets” y las coordenadas “0, 0, 0” corresponden al centro del padre o “ImageTarget” que a su vez tenía otras coordenadas “0, 0, 0” que corresponden a la escena de Unity, que eran las que usaba “iTween” y era complejo obtener todas las coordenadas bien de cada recurso. En cambio, con el gestor de animaciones de Unity, con mover nosotros el recurso a la posición que queríamos en el segundo que queríamos, ya se calculaba sola la animación. Aunque esto último también tenía el problema de que en algunas ocasiones el recurso AR se movía alrededor de esa posición que le habías indicado y había que duplicar los datos de esa posición que queríamos inmediatamente después para impedir esa animación que generaba Unity automáticamente.

Desarrollo con Metaio SDK y Creator

En nuestro ejemplo realizado con Metaio Creator, hemos añadido una imagen como “Image Tracking” y le hemos puesto como recurso de AR un modelo realizado con “SketchUp” y exportado con formato de objeto “.obj”, quedando algo similar a lo siguiente:

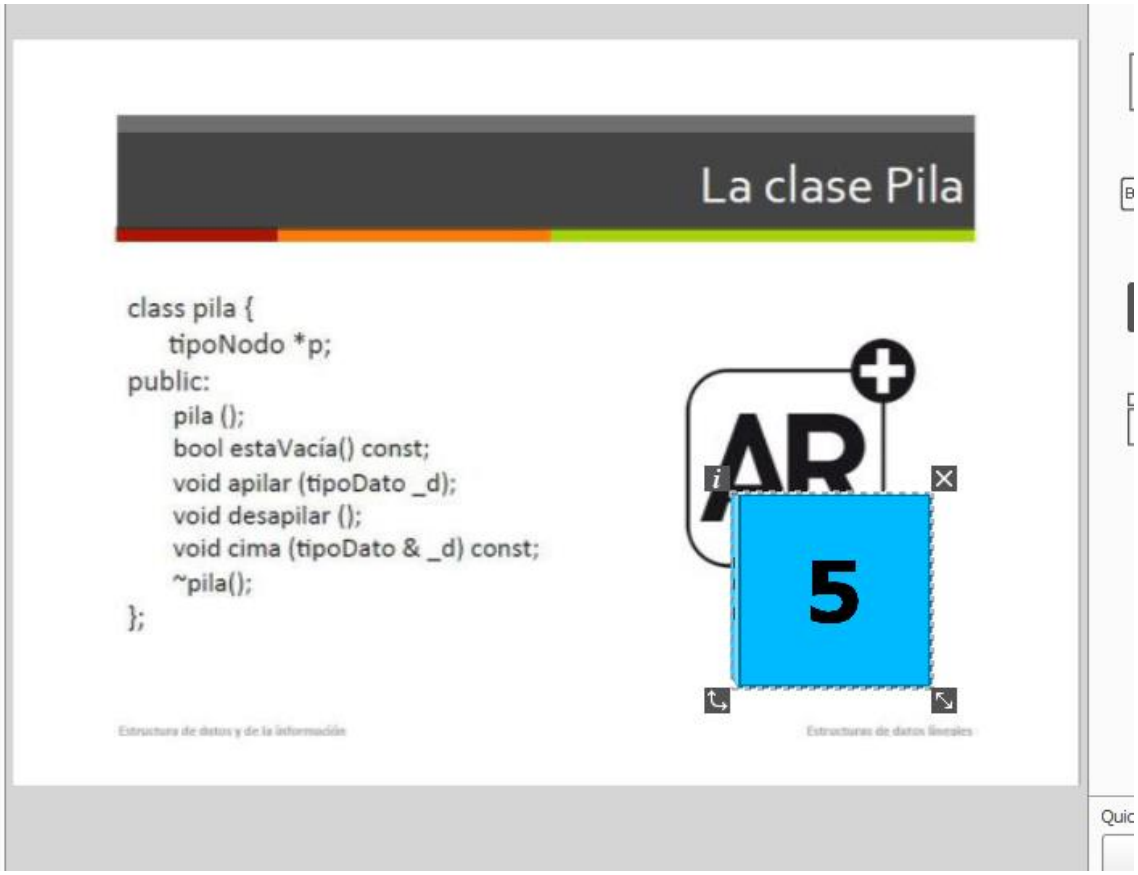

Ilustración 53 - Ejemplo Metaio Creator

A los recursos que vamos añadiendo les podemos añadir animaciones para que o bien se muevan por la pantalla, roten sobre sí mismos o cambien de tamaño. Para lograr esto, tendremos que hacer clic con el botón derecho sobre el recurso que acabamos de importar y entrar en “Properties”; de ahí, pasamos a la pestaña “Animation” y veremos algo similar a la siguiente imagen:

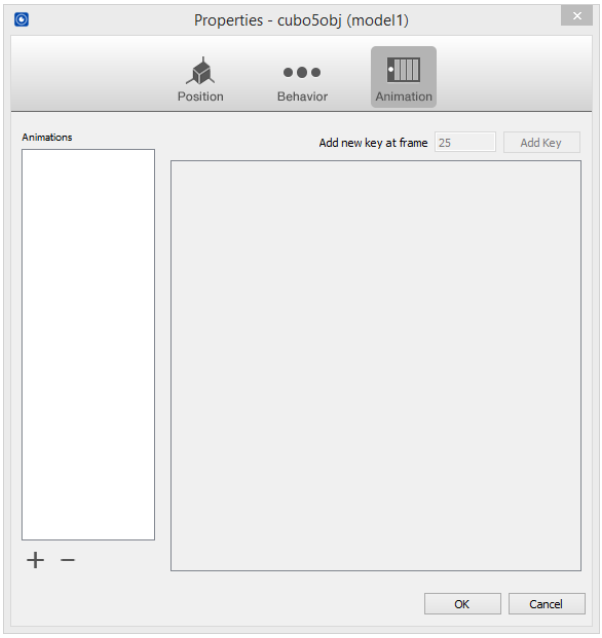

Ilustración 54 - Añadir animación 1

En esa pantalla, pulsaremos sobre el botón con el símbolo "+" para añadir una nueva animación y elegiremos el "frame" o imagen en el que queremos que haga esa animación y las coordenadas de la nueva posición, grados que queremos que gire o tamaño que queremos que tenga al terminar la animación. Podemos añadir cuantas animaciones queramos, pero realmente luego podremos mostrar solamente dos, una puede configurarse para que se reproduzca cuando aparezca el recurso y otra si se pulsa.

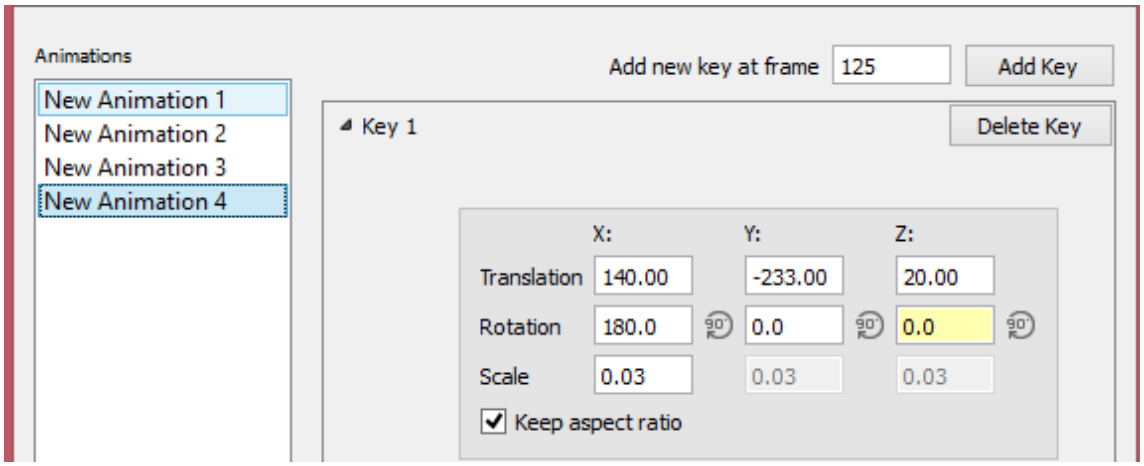

Ilustración 55 – Añadir animación 2 – Creación animación

A continuación, para que se reproduzca la animación; en la pestaña "Behaviour", podremos configurar varias opciones para el modelo, desde cambiar su código "JavaScript" asociado, hasta asociarle acciones cuando se pulse o cuando aparezca. Ahora solo configuraremos las animaciones como comentamos antes para cuando aparezca y para cuando sea pulsado.

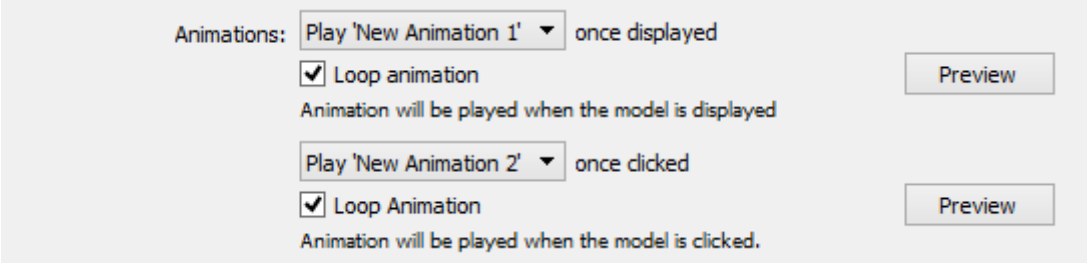

Ilustración 56 – Añadir animación 3 – Configuración reproducción

Para poder probar lo que llevamos hecho hasta ahora, tenemos diferentes opciones. Una de ellas es directamente pulsando sobre el botón "Preview" que se encuentra debajo de los recursos AR y usará nuestra Webcam en el caso de que tengamos una. Por otro lado, si lo queremos probarlo en el móvil, tendremos que exportar la aplicación, pero no nos exportará una aplicación que se instala directamente en un dispositivo, sino que nos creará un proyecto que en el caso de Android será compatible con Eclipse y Android Studio y, en el caso de iOS nos creará un proyecto iOS Xcode compatible.

Una vez hemos compilado el proyecto y lo ejecutamos en un dispositivo Android veremos algo similar a las siguientes capturas:

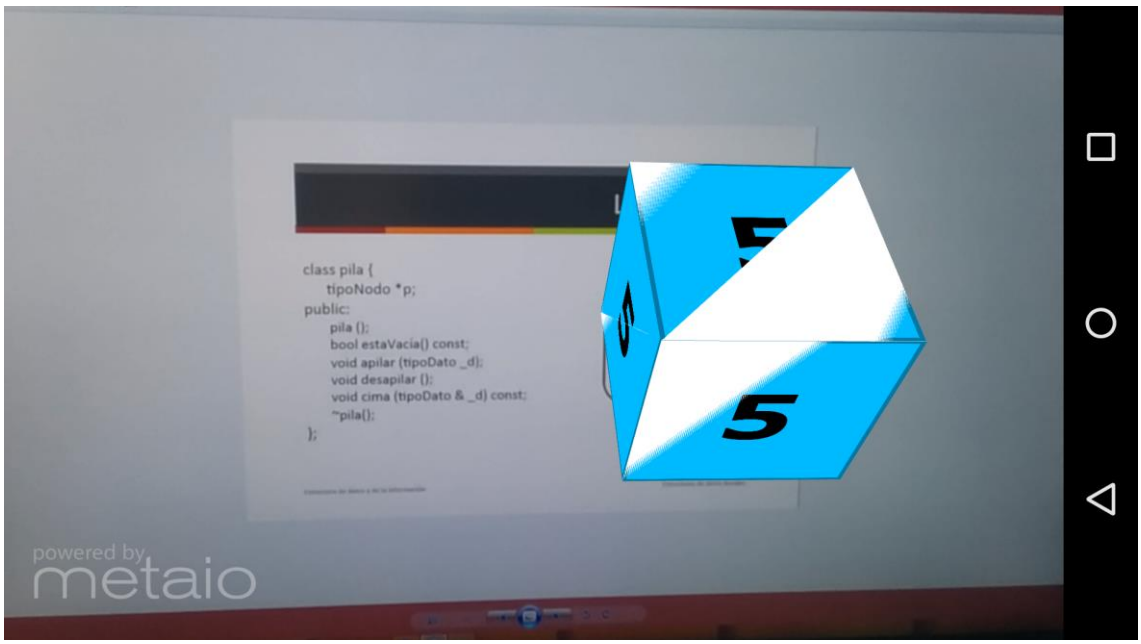

Ilustración 57 - Demo de aplicación al reconocer el patrón

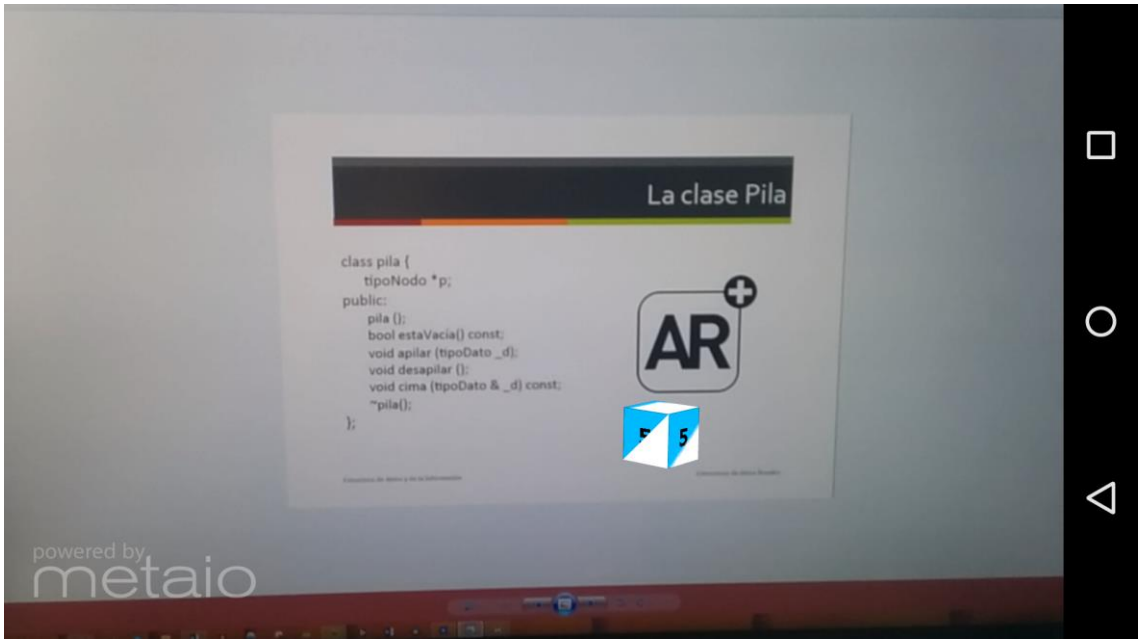

Ilustración 58 - Demo de aplicación al pulsar sobre el recurso AR

Como podemos comprobar, la aplicación no se lleva bien con los modelos de SketchUp en formato de objeto ".obj", pero tampoco con el formato de Collada o FBX, que son los compatibles con Metaio que se pueden exportar desde SketchUp.

Tras probar con ese modelo de SketchUp y ver que daba problemas, decidimos probar con los demás formatos compatibles y también con los modelos creados con Blender sin ninguna animación vinculada a los modelos.

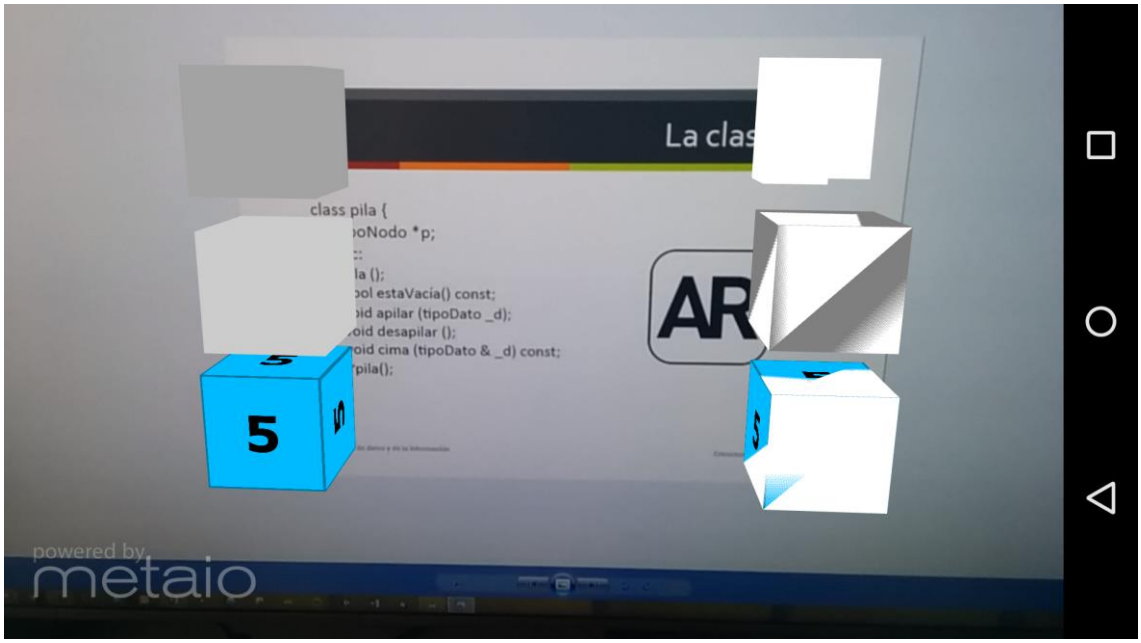

Ilustración 59 - Modelos Metaio Blender Vs SketchUp 1

Ilustración 60 - Modelos Metaio Blender Vs SketchUp 2

En las imágenes anteriores, podemos ver 6 recursos AR diferentes distribuidos en dos columnas. La columna de la izquierda se corresponde a los modelos exportados con el programa Blender, mientras que los de la derecha son aquellos creados con el programa SketchUp. Cada fila se corresponde también al mismo formato exportado en ambos programas, siendo la primera file los exportados en formato "dae", la segunda "fbx" y la tercera "obj" que, como podemos ver hasta ahora; es la que mejor funciona.

Observamos que los modelos de SketchUp dan problemas a la hora de mostrarse, mientras que los de Blender no tienen ese problema con las luces, aunque no muestran sus imágenes o texturas en 2 de los 3 formatos que hemos probado.

Para poder probar lo que llevamos hecho hasta ahora, tenemos diferentes opciones. Una de ellas es directamente pulsando sobre el botón "Preview" que se encuentra debajo de los recursos AR y usará nuestra Webcam en el caso de que tengamos una. Por otro lado, si lo queremos probar en el móvil tendremos que exportar la aplicación, pero no nos exportará una aplicación que se instala directamente en un dispositivo, nos creará un proyecto que en el caso de Android será compatible con Eclipse y Android Studio y en el caso de iOS nos creará un proyecto iOS Xcode compatible. Para ello tendremos que haber iniciado sesión previamente en la herramienta con nuestra cuenta de Metaio ya sea cuando nos lo pregunta al ejecutarla o pulsando sobre la opción "Settings".

Principales problemas encontrados durante el desarrollo

- No se pueden añadir más de dos animaciones y cada una solo se puede configurar para cuando aparece el recurso o para cuando se pulsa. Es una limitación de la versión gratuita.
- No se pueden mostrar más de 6 recursos AR a la vez. Es una limitación de la versión gratuita.
- No se pueden poner más de 2 objetos de "tracking" diferentes, por lo que no nos va a servir para el desarrollo de nuestra aplicación. Es una limitación de la versión gratuita.
- Actualmente no se puede comprar una versión de pago porque la empresa lo ha restringido al ser comprada por Apple[54].

Metaio SDK en Unity 3D

Tras haber intentado iniciar un proyecto en él, sólo es compatible con Unity Pro, no con la versión gratuita.

Para poder hacer la aplicación con Unity, tenemos que exportarla como proyecto Android y luego compilarla con Android Studio.

Desarrollo con ARMedia SDK y Unity

Tras haber probado a hacer la aplicación, puesto que el SDK para Android y para Unity es gratuito, cuando vas a pedir una licencia para el SDK y obtener la clave para poderlo utilizar te piden una subscripción, y estas son de pago (1000 €, 2500 € y 3500 €).

Resumen del desarrollo de la propuesta

En esta sección haremos un breve resumen de lo que se ha comentado en los desarrollos anteriores sobre la segunda propuesta de una forma más esquemática para aclarar lo realizado.

Resumen de Unity

La aplicación a la que queríamos llegar en el desarrollo de Unity es la que podemos ver en las siguientes ilustraciones: Ilustración 51 - Prueba de animación de la propuesta 2 desde Unity en PC e Ilustración 52 - Prueba de animación de la propuesta 2 desde Unity en PC.

Para el desarrollo de Unity, de manera esquemática se ha realizado un proceso similar al indicado en la siguiente imagen:

[54] http://www.applesfera.com/apple-1/apple-quiere-entrar-de-lleno-en-la-realidad-virtual-y-adquiere-metaio – Último acceso: 02/06/2015

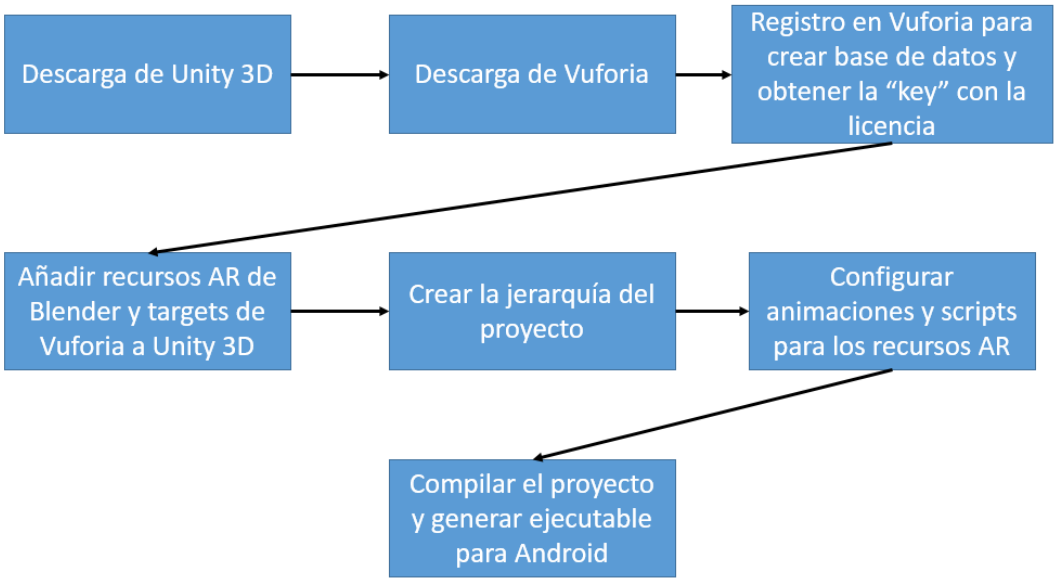

Ilustración 61 - Resumen desarrollo propuesta 1 en Unity con Vuforia

Como podemos ver en la imagen, se ha realizado una explicación acerca de la descarga e instalación de Unity 3D junto con el plugin de Vuforia y la instalación de los mismos. A continuación se ha indicado cómo obtener la clave de la licencia para poder utilizar Vuforia y cómo obtener los "targets" en el formato adecuado para utilizarlos en nuestras aplicaciones.

Una vez tenemos Unity configurado, hemos visto como añadir los modelos que habíamos creado con Blender y los "targets" que hemos obtenido de la web de Vuforia. Se ha comentado también cómo crear una "escena" de la aplicación añadiendo los componentes de Vuforia y los recursos AR que queremos que aparecieran para cada tarjeta y cómo se iba formando el árbol con la jerarquía de la aplicación.

Además, hemos realizado una guía sobre cómo utilizar las animaciones y scripts en C# en Unity.

Por último, en el desarrollo podemos ver una guía sobre cómo compilar el proyecto en Unity 3D y ejecutarlo en un dispositivo Android real o general el código para editarlo en "Android Studio", el entorno de desarrollo oficial de Android.

Resumen de Metaio

Para el resumen del desarrollo en Metaio utilizamos como ejemplo de la aplicación que queríamos crear la Ilustración 53 - Ejemplo Metaio Creator, en la que podemos ver una imagen y un recurso AR que aparecerá cuando la aplicación reconozca dicha imagen.

Un esquema de lo que se ha realizado en el apartado de Metaio sería el que podemos ver a continuación:

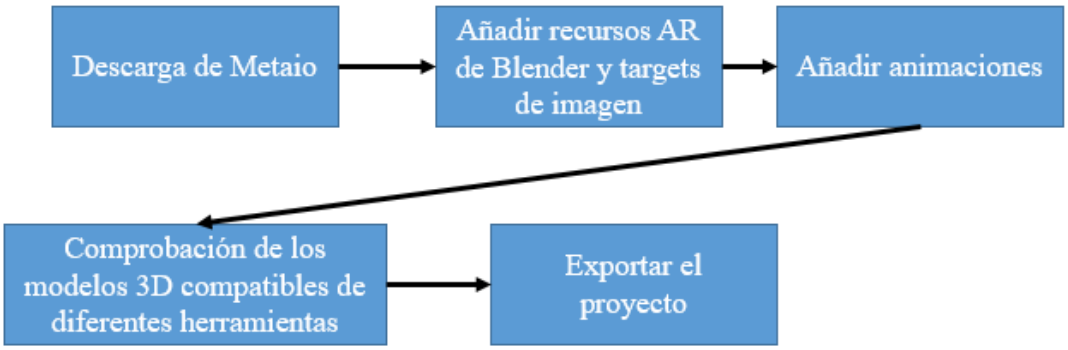

Ilustración 62 - Resumen desarrollo propuesta 1 en Metaio

Como se observa en la imagen, al principio del apartado hemos comentado cómo y de dónde descargar el SDK y el software propio de la herramienta. Una vez lo tenemos instalado, hemos creado un proyecto en el que hemos añadido un "target de imagen" y un recurso AR sobre el que trabajar con las animaciones, pero que sólo nos permite añadir dos animaciones por cada recurso, una para cuando aparezca y otra para cuando se pulse. No podemos hacer grandes animaciones de gran complejidad con la herramienta. Además, como vimos que nos dio problemas con los modelos que habíamos exportado con "SketchUp", quisimos probar los tres formatos compatibles con Metaio que podemos exportar desde "Blender" y "SketchUp", viendo que la primera da mejores resultados.

Por último hemos visto cómo exportar el proyecto para poder modificar la información en el entorno de desarrollo oficial de Android, Android Studio.

Desarrollo segunda aplicación – Tarjetas de visita

Preguntas previas

Antes de comenzar el desarrollo de la aplicación, al igual que para la otra propuesta; nos hemos enfrentado a varias cuestiones:

- ¿Dónde tener la información de la aplicación? Realmente para un par de datos, consideremos un par del orden de cientos como podría llegar a ser la primera aplicación; no habría problema con la memoria del dispositivo, ya que además puedes clasificarlos por asignaturas o temas e ir solicitando las que sean más susceptibles de ser usadas en ese momento. Pero para una aplicación como esta, se debería de tener acceso a todas las tarjetas para compararlas. Algunos SDK permiten tener una gran cantidad de "targets" en una base de datos "cloud", en la nube, y él se ocupa de ir comprobándolo, por lo que podría ser una solución.

- ¿Deberían tener las tarjetas un formato concreto en el que se muestre quizás un código QR? Ahora mismo, con lo que estamos trabajando es con reconocimiento de imágenes y con las tarjetas, por lo que es muy fácil que se parezcan unas a otras. Una ampliación podría ser usar códigos QR que se solicitaran a un servidor web para que se descargara la información a mostrar. Así solucionaríamos tanto este problema como el anterior del número de tarjetas en memoria.

Desarrollo en Unity 3D con Vuforia

El desarrollo con la herramienta Unity 3D y el SDK de Vuforia es similar al que hemos realizado para la aplicación anterior, por lo que empezaremos a explicar el proceso desde que se han añadido los dos "ImageTarget" y los recursos asociados a cada uno de ellos.

En nuestro punto de partida tendremos dos "ImageTarget", como hemos comentado, y varios recursos asociados a cada uno de ellos, entre los que podemos destacar una esfera y varios cubos, además de diversas imágenes en uno de los "ImageTarget". En la siguiente imagen ilustramos lo comentado:

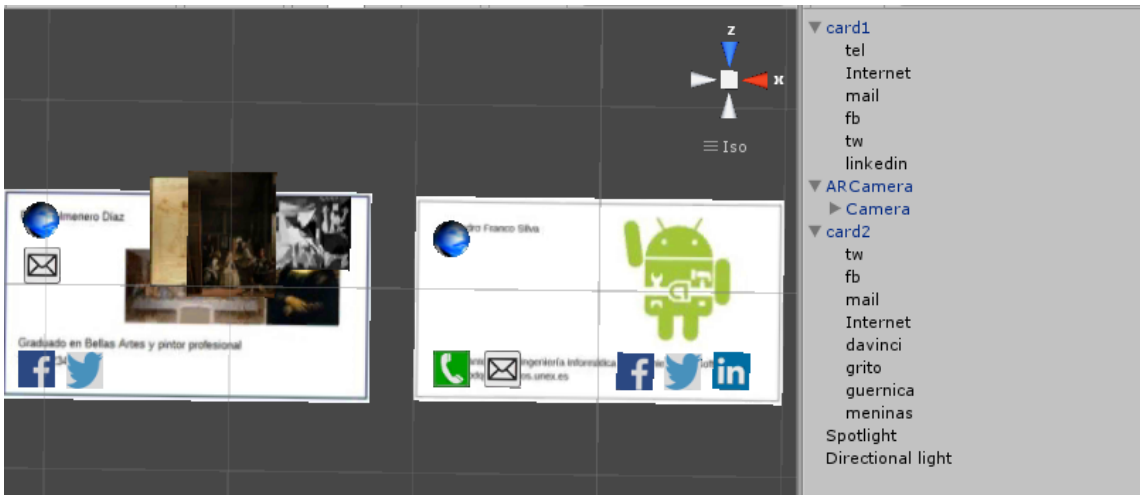

Ilustración 63 - Modelos añadidos en Unity con Vuforia en propuesta 2

Los modelos que podemos ver han sido diseñados y exportados con Blender, ya que es el único que no nos ha dado problemas.

Cada uno de los recursos AR tiene asociado un "script" el cuál nos permite ejecutar código JavaScript o C#.

Para añadirle un script a un componente concreto basta con seleccionarlo, acceder a la vista de "Inspector" situada en la parte derecha de Unity y pulsar, abajo, sobre "Add Component" y a continuación seleccionar "New Script". Dentro de este último paso,

podemos seleccionar el lenguaje que queremos utilizar en nuestro “Script” entre los que tenemos “CSharp” o “C#” y JavaScript.

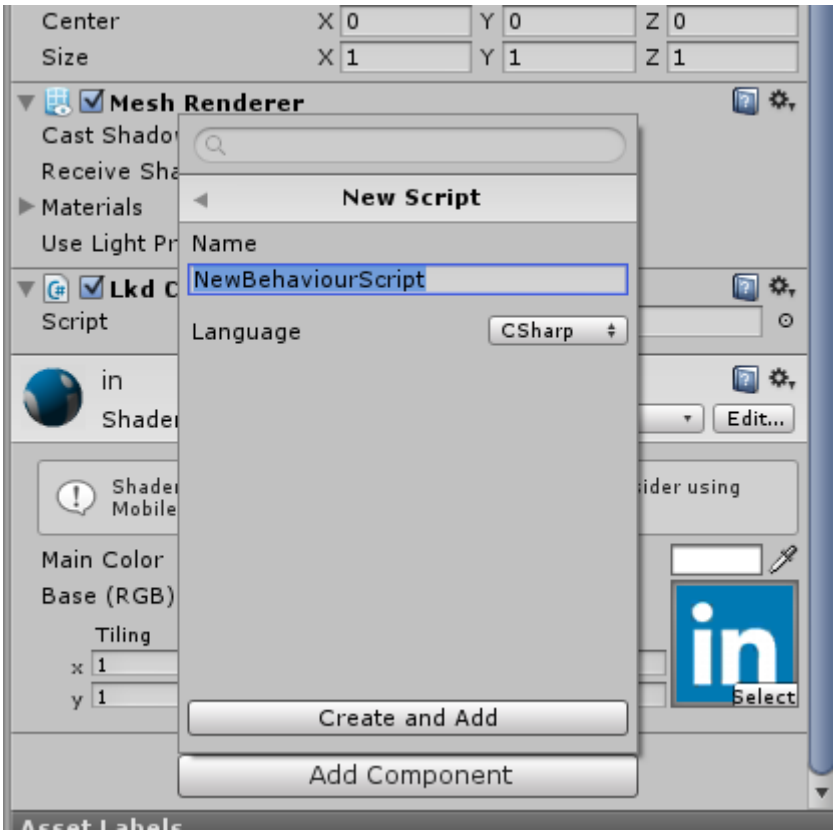

Ilustración 64 - Añadir nuevo Script a un objeto en Unity para la propuesta 2

Una vez hayamos creado el script, nos saldrá en la vista de “Inspector” como un nuevo componente.

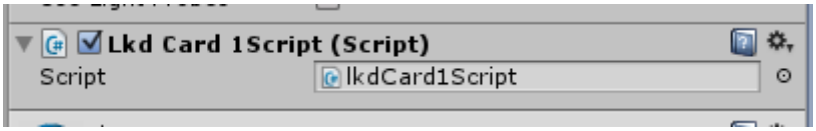

Ilustración 65 - Script en la vista de "Inspector" en Unity

Al hacer doble clic sobre él, se nos abrirá el software “MonoDevelop”, que es un entorno de desarrollo que viene integrado y es libre y gratuito.

En el fichero que se nos abre, encontramos dos módulos creados por defecto, que son “Start” y “Update”. “Start” se ejecutará en la inicialización de la aplicación, haya reconocido o no el “target” asociado al recurso al que pertenece este script. “Update” se invocará en todos los “frames” de nuestra aplicación. Con este método hay que tener cuidado, porque nuestra aplicación se ejecutará dependiendo del dispositivo en torno a los 30 “frames per second” (fps), imágenes por segundo; lo cual implica que se invocará 30 veces por segundo, lo que pongamos dentro de ese módulo.

En nuestro script, añadiremos una nueva función descendiendo directamente de la clase, como los métodos “Start” y “Update”, que tendrá el siguiente código:

```
void OnMouseDown(){
        Application.OpenURL ("https://es.linkedin.com/in/afrasilv");
}
```

Podemos ver que el código es muy simple; hace que al pulsar sobre el modelo, se ejecute el código que le hayamos puesto, que en nuestro caso le envía a la aplicación una indicación para que abra una dirección web.

En otros scripts, como el de teléfono o el mail, los códigos son similares, pero utilizando las funciones “tel:xxxxxxxxx” para llamar y que nos abrirá la aplicación asociada al teléfono de nuestro dispositivo, y “mailto:” para mandar un mail y nos abrirá la aplicación asociada al correo electrónico. El código sería respectivamente:

- Teléfono:

```
void OnMouseDown(){
        Application.OpenURL ("tel:612345678");
}
```

- Correo electrónico:

```
void OnMouseDown(){
        Application.OpenURL ("mailto:afrancodq@alumnos.unex.es");
}
```

Estos son los scripts más simples, ya que el sistema lo controla solo y no hay que indicarle nada más, además de que los navegadores actuales comprenden las urls o direcciones web “tel” y “mailto”.

Animación de las imágenes

Mediante las animaciones de Unity 3D

Como comentamos en la aplicación anterior, vamos a utilizar la herramienta integrada “Animation” dentro de Unity para hacer las animaciones.

En nuestro ejemplo, utilizaremos el objeto de la imagen con nombre “davinci”:

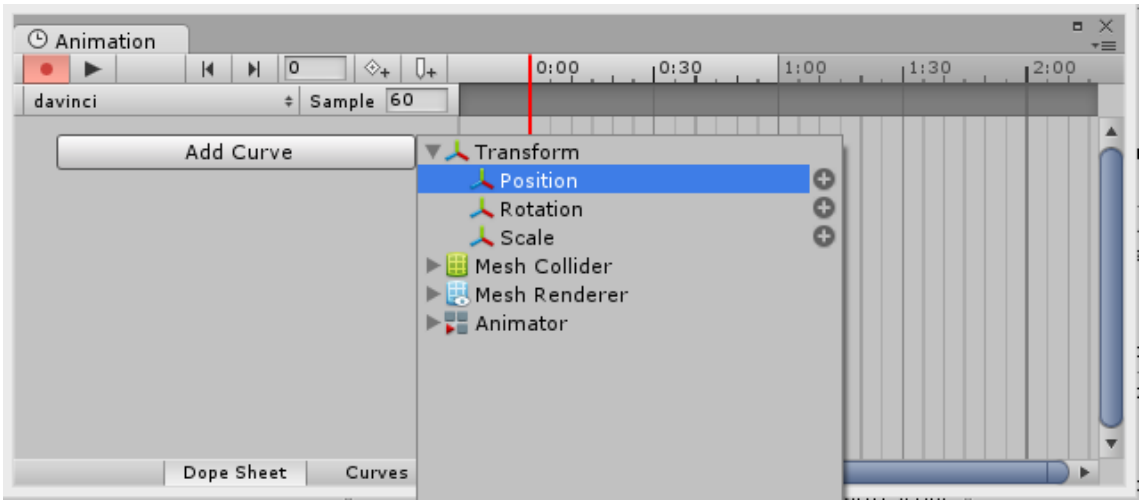

Ilustración 66 - Ventana "Animation" en Unity

Cuando lo hayamos añadido, veremos que se han introducido unas columnas de puntos en unos tiempos concretos que podemos ver en la barra superior, mientras que en la parte izquierda podemos ver los valores. Esas columnas de puntos se pueden mover para seleccionar el tiempo concreto que deseemos y, una vez esté, colocadas en el tiempo que queramos, le daremos el valor que deseemos a las coordenadas. Para añadir una nueva columna de puntos debemos hacer doble clic en el tiempo que queramos y luego se podrá modificar.

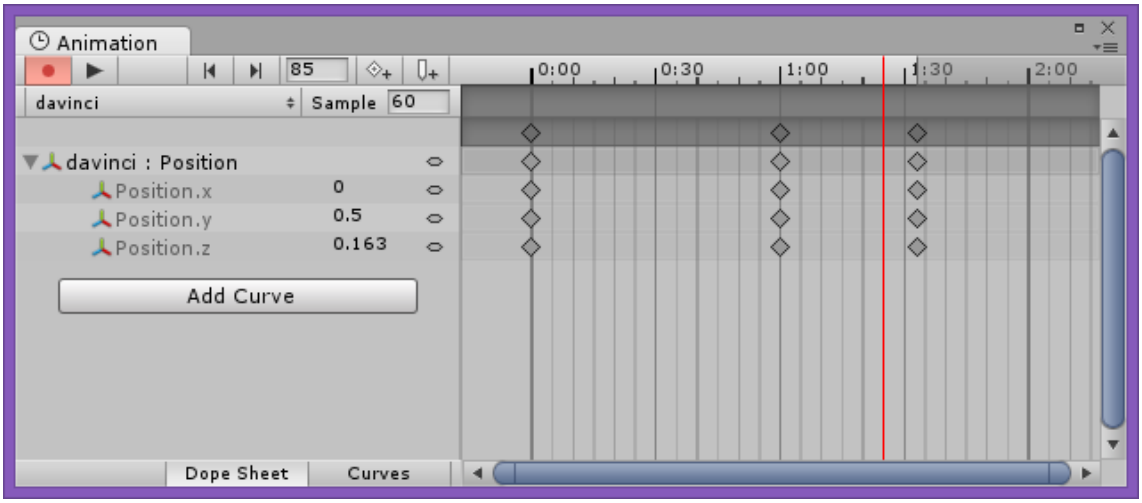

Ilustración 67 - Ventana "Animation" en Unity con varios tiempos añadidos

Para comprobar los cambios que estamos haciendo, tendremos que pulsar sobre el botón de play que se encuentra en la barra superior y en las ventanas de escena y "game" podremos ver el resultado.

Mediante script en C#

Para las 4 imágenes que tenemos en la tarjeta de la izquierda, el script es más complejo; ya que quisimos que tuvieran una pequeña animación en la que fueran rotando poco a poco.

Existe una extensión para Unity que hace esta tarea de una forma bastante estética y simple de codificar; esta extensión es "iTween"[55]. Hemos tratado de utilizarla y pese a que los objetos AR se movían, estos no lo hacían en las coordenadas correctas. Como para que funcionen con Vuforia tienen que descender en la vista de "Hierarchy" o jerarquía de un "ImageTarget" y al hacer eso pasan a tener unas coordenadas "x,y,z" respecto al punto "0,0,0", que se encuentra en el centro de dicho "ImageTarget" y "iTween" utiliza las coordenadas absolutas respecto al punto "0,0,0" real de nuestra escena en Unity. Esto no es un gran problema, ya que mediante código se pueden averiguar estas coordenadas, pero "iTween" no las reconoce bien y envía cada objeto a unas coordenadas que no tienen nada que ver con las que le hemos puesto pese a no haber nada más ejecutándose en ese momento, por lo que no será compatible con Vuforia. Sin utilizar nada de AR funciona correctamente.

Para hacer las animaciones de las 4 imágenes rotando, hemos utilizado un contador para saber en qué posición está y una variable que controla el tiempo en el que se inicializa la aplicación y ejecuta un cambio de posición cada 2 segundos.

Para ello, utilizamos también un script en "C#" y creamos las variables referentes a las 4 imágenes y 4 vectores con las posiciones iniciales de cada una de ellas.

```
GameObject meninas;
GameObject davinci;
GameObject grito;
GameObject guernica;

Vector3 vectorMeninas;
Vector3 vectorVinci;
Vector3 vectorGrito;
Vector3 vectorGuernica;

int contMov = 0;
float myTime;
```

[55] http://itween.pixelplacement.com/ - Último acceso 24/06/2015

Las variables referentes a las imágenes son las 4 primeras del tipo “GameObject” ya que en Unity todas las imágenes y modelos de Blender que tenemos ahora mismo añadidos son de ese tipo.
En el módulo “Start”, asociamos cada “GameObject” con su objeto en nuestra escena y cada vector con las posiciones iniciales de los mismos. Además, inicializamos también la variable que controla el tiempo como podemos ver a continuación:

```
meninas = GameObject.Find("meninas");
vectorMeninas = meninas.transform.position;
myTime = Time.realtimeSinceStartup + 2f;
```

En el código, podemos ver cómo se obtiene el objeto “meninas” de nuestra escena y se lo asocia a nuestro “GameObject”; para ello, tiene que tener el mismo nombre que le indicamos nosotros aquí. En la segunda línea, accedemos a la posición del objeto. Respecto al tiempo, hay varias funciones que nos devuelven el tiempo, cada una es interesante según cada situación; esta función que usamos nos devuelve el tiempo que ha pasado desde que se ejecutó la aplicación y nosotros le sumamos 2 segundos en formato “float”.
Ya en el módulo “Update”, comprobamos el tiempo que ha pasado y cuando se cumplan los 2 segundos, ejecutamos el código y actualizamos nuestro tiempo:

```
if(myTime <= Time.realtimeSinceStartup){
        switch(contMov){
                case 0:
                        contMov=1;
                        movUno();
                        break;
        […]
        myTime = Time.realtimeSinceStartup + 2f;
```

En este extracto de código del módulo “Update” comprobamos si nuestra variable de tiempo es menor que el actual y en ese caso entramos, comprobamos en qué movimiento estamos y cambiamos el contador de movimiento y movemos los objetos. Por último, actualizamos el tiempo.

El movimiento de los objetos es muy simple teniendo las coordenadas de los otros objetos, como podemos ver en nuestro módulo “movUno()” que hemos invocado en el fragmento anterior:

```
public void movUno(){
        meninas.transform.position = vectorVinci;
        davinci.transform.position = vectorGrito;
        grito.transform.position = vectorGuernica;
        guernica.transform.position = vectorMeninas;
}
```

En este fragmento, apreciamos como se modifican las posiciones de cada uno de ellos simulando un movimiento en círculo.

Por último, ya sólo tendríamos que ejecutarlo o exportarlo como proyecto para Android Studio como ya hemos visto en la aplicación anterior.

Principales problemas encontrados

Los principales problemas que nos hemos encontrado en el desarrollo de este prototipo en Vuforia y Unity 3D se han presentado a la hora de añadir una animación mediante código en los scripts por los problemas generados por las coordenadas de los objetos.

Desarrollo con Metaio SDK

En lo referente al desarrollo en Metaio, debido a las grandes limitaciones que incluye la herramienta, su uso no ha sido posible. Si añadimos 2 imágenes, una para cada tarjeta, sólo nos permite añadir 3 recursos AR. Pero como en la aplicación anterior no comentamos como se añadían las acciones al pulsar sobre el recurso, lo comentaremos con este ejemplo de nuestra propuesta de aplicación.

En la siguiente imagen podremos ver una de las tarjetas que estamos usando como “target” y dos recursos AR, un sobre y un botón para Twitter.

Ilustración 68 - Propuesta 2 en Metaio Creator

Para asociarles una acción cuando sean pulsadas, basta con darles con el botón derecho al recurso que quieres en este momento e irte a la opción de “Properties”.

Dentro de este menú, nos vamos a la pestaña “Behavior” y en la parte inferior, podremos leer “Action when clicked” y tenemos una lista con varias opciones:

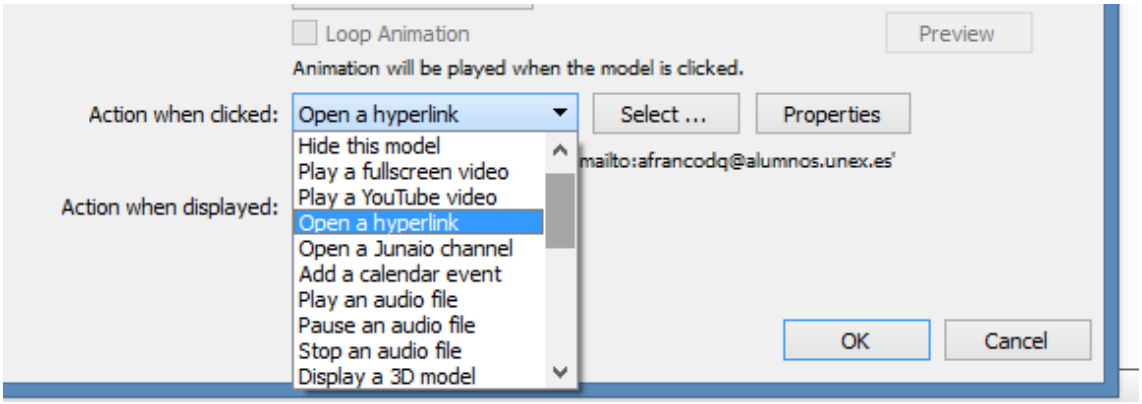

Ilustración 69 - Acciones al pulsar un recurso AR en Metaio Creator

Al seleccionar la opción que queremos, le daremos a “Select…” y añadiremos la información asociada a lo que hemos puesto. Nosotros hemos seleccionado un hipervínculo y en “Select…” hemos añadido su información: “mailto:afrancodq@alumnos.unex.es”

Principales problemas encontrados

Los problemas principales de Metaio radican en las grandes limitaciones que tiene su SDK, que no da apenas margen para probar varias acciones, animaciones o recursos AR en la misma aplicación.

Resumen del desarrollo de la propuesta

En esta sección haremos un breve resumen de lo que se ha comentado en los desarrollos anteriores sobre la segunda propuesta de una forma más esquemática para aclarar lo realizado.

Resumen de Unity

Sirviéndonos de la Ilustración 63 - Modelos añadidos en Unity con Vuforia en propuesta 2, la idea del desarrollo de la aplicaciones es que pueda reconocer dos tarjetas que nosotros le hemos configurado y varios botones a modo de recursos AR en los cuales controlaremos los eventos de pulsación para saber si quieres acceder a una web o a una de las redes sociales que salen, por ejemplo.

En el resumen de esta sección, tenemos que decir que partimos de la base de que se ha realizado primero la propuesta anterior y se sabe hacer las cosas. Aparte de ello, se ha comentado cómo realizar las siguientes tareas que vemos en el esquema:

Ilustración 70 - Resumen desarrollo propuesta 2 en Unity 3D y Vuforia

Como vemos, en esta propuesta no hemos tenido que realizar muchas más tareas que para la propuesta anterior, ya que solo hemos visto de nuevo cómo se añaden las animaciones a los recursos AR de esta propuesta y hemos comentado también cómo se añaden eventos de clic al pulsar sobre los diferentes recursos AR.

Resumen de Metaio

Como propuesta de aplicación final, podemos verla en la Ilustración 68 - Propuesta 2 en Metaio Creator, que encontramos la imagen de una tarjeta y dos botones o recursos AR a los que se le asociará un evento de clic para interactuar con la aplicación.

De nuevo, en este resumen; damos por hecho de que se ha realizado la propuesta anterior en Metaio y que se tiene ya la configuración previa del SDK y herramienta. Además de ello, durante el desarrollo de la propuesta hemos podido ver las siguientes tareas que vemos en este esquema:

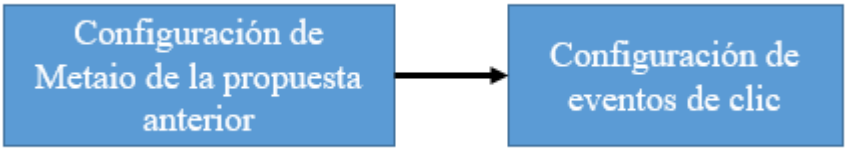

Ilustración 71 - Resumen desarrollo propuesta 2 en Metaio

Durante el desarrollo en Metaio, hemos visto cómo se añaden eventos a los recursos AR para que ejecuten diferentes tareas cuando sean seleccionados.

Desarrollo de las aplicaciones en Android Studio

El desarrollo final de las dos propuestas de aplicación se ha realizado en Android Studio por ser el entorno de desarrollo oficial para Android y, además, gratuito.

En este apartado veremos a grandes rasgos cómo es el código generado por las diferentes herramientas seleccionadas y ver qué se puede hacer con ese código y cómo podemos añadirle más datos o modificarlos.

Proyectos exportados desde Unity 3D

La estructura de los proyectos exportados usando la herramienta de Unity en Android Studio es la que podemos ver en la siguiente imagen:

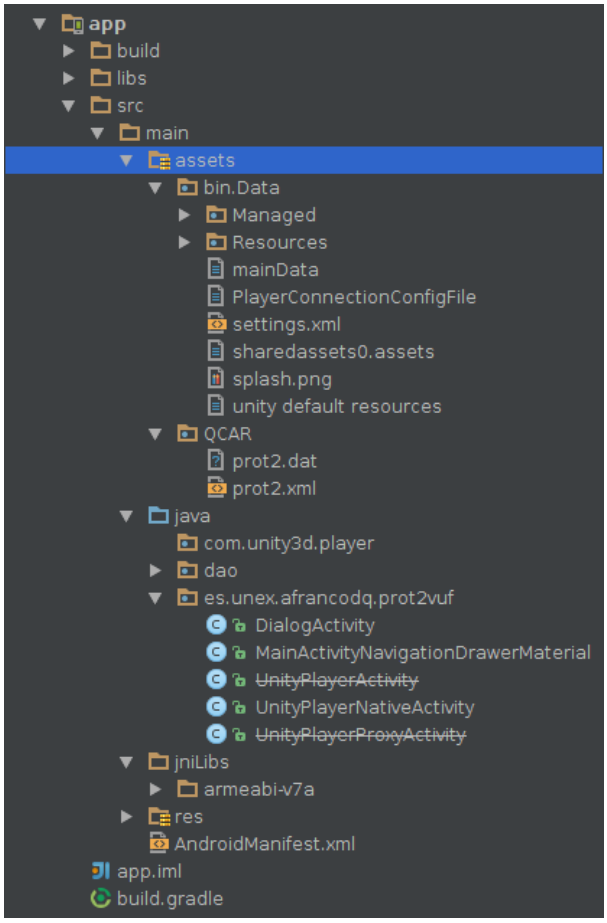

Ilustración 72 - Estructura de un proyecto exportado por Unity en Android Studio

En la Ilustración 72 - Estructura de un proyecto exportado por Unity en Android Studio, podemos ver una jerarquía de carpetas y dentro de ellas una llamada “java”. En su interior podemos encontrar otras tres, la carpeta “dao” ha sido creada por nosotros para tener una clase “User” que controle las acciones del usuario y saber el rol de la persona que usa la aplicación. Dentro de la carpeta inferior, “es.unex.afrancodq.prot2vuf”, podemos encontrar cinco ficheros java, de los que los dos primeros han sido creados por nosotros y los tres inferiores son generados por Unity.

Otra vista de la arquitectura del proyecto en Android Studio exportado por Unity la podemos ver en la siguiente imagen para ayudar a su comprensión:

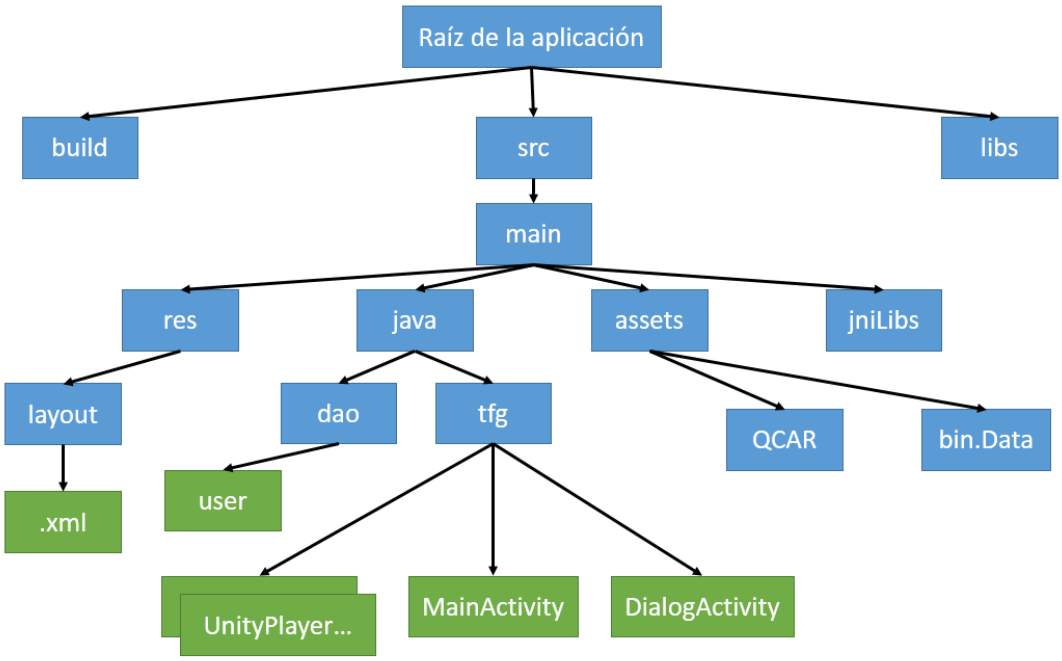

Ilustración 73 - Esquema de la estructura de un proyecto exportado por Unity en Android Studio

Como podemos ver en la imagen, la estructura es igual que la de la Ilustración 72 - Estructura de un proyecto exportado por Unity en Android Studio, sacada directamente de la aplicación. Aquí podemos ver con más claridad los archivos que más nos interesan y/o que hemos modificado. Como los ".xml" de la carpeta "layout" para las diferentes vistas del menú de opciones. El fichero "user" para un futuro control de usuarios y los ficheros que descienden de la carpeta "tfg", que son los principales de la aplicación.

En el fichero "DialogActivity" controlamos toda la interacción del menú que aparece al pulsar sobre el botón de opciones y vamos actualizando la interfaz con unos datos u otros. En el segundo fichero encontramos la clase "MainActivityNavigationDrawerMaterial", en la que controlamos la funcionalidad del botón de opciones que aparece en la esquina superior derecha y además controlamos que la cámara de Unity, la que hemos configurado anteriormente en el proyecto, arranque y ocupe todo el tamaño de la pantalla. Además, si abrimos el menú de opciones, la cámara se pone en pausa y, cuando se cierra, se reanuda su funcionamiento.

En las tres clases inferiores, dos de ellas se encuentran porque Android Studio las considera obsoletas o "deprecated" y te recomienda usar la clase "UnityPlayerNativeActivity", que tiene la misma funcionalidad que la creada por nosotros "MainActivityNavigationDrawerMaterial", pero sólo controlando la cámara, como no le puedes añadir nada más a esa clase, hemos creado una clase nueva desde cero.

También podemos ver una carpeta que se encuentra seleccionada, la carpeta “assets”; esa carpeta es la generada por Unity 3D y se encuentran en ella en binario todos los “targets” que tengamos asociados y se encuentran dentro del archivo “prot2.dat” y “prot2.xml” que podemos encontrar dentro de la carpeta “QCAR”.
Dentro de la carpeta “Managed” nos encontramos con varios ficheros de librerías “dll” y en “Resources” hay un fichero de configuración para Unity. Por último, al final de la imagen, se puede ver una carpeta llamada “jniLibs”, que contiene librerías de Unity y Vuforia para los procesadores con arquitectura “arm” en su versión 7, la más extendida.

Clase MainActivityNavigationDrawerMaterial

Como hemos comentado anteriormente, en esta clase controlamos la interacción con el botón de opciones y la cámara de Unity y Vuforia.
Para el botón flotante azul, utilizamos una librería externa de “getbase” llamada “floatingactionbutton”[56]. Para poder utilizarla, añadimos la librería al proyecto y en nuestra clase creamos un “listener” o escuchador asociado a un evento de hacerle clic. Un fragmento del código que hace esa función es el siguiente:

```
settingsButton = (FloatingActionButton) findViewById(R.id.floating_button_settings);
settingsButton.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      Intent i = new Intent(MainActivityNavigationDrawerMaterial.this,
DialogActivity.class);
      i.putExtra("sesionIniciada", sesionIniciada);
      startActivity(i);
   }
});
```

En este fragmento de código podemos apreciar que en la primera línea se le vincula la vista del botón azul con una variable global llamada “settingsButton” del tipo “FloatingActionButton”, la librería externa que añadimos antes. A continuación, le añadimos el “listener” a nuestra variable. En el método “onClick” que se invocará en el momento en el que hagamos clic a la vista o “view”, hacemos una llamada mediante un “intent” a nuestra clase diálogo y le pasamos por parámetro el valor de una variable que

[56] https://github.com/futuresimple/android-floating-action-button - Última visita: 30/07/2015

comprueba si ya hemos iniciado sesión o no. Por último, invocamos el método "startActivity" con el "intent" por parámetro, que hará la función de ejecutar la clase del diálogo o nuestro menú de opciones.

Clase DialogActivity

Como también hemos comentado anteriormente, esta clase se encarga de la vista del menú de opciones y la interacción con este.

Comentaremos a continuación un par de ejemplos de código en los que se ve por completo el funcionamiento de la clase.

Para obtener la interacción con los diferentes botones que aparecen, tan sólo tendremos que inicializar el "button" o botón con su vista asociada y añadirle un "listener" para el evento clic, como podemos ver a continuación:

```
Button deleteTopic = (Button) findViewById(R.id.deleteTopic);
deleteTopic.setOnClickListener(new View.OnClickListener() {
   public void onClick(View v){
   });
});
```

En el ejemplo que hemos usado, "deleteTopic" o borrar tema; nos saldrá un nuevo diálogo que nos avisará de que si borramos el tema se borrarán todas las animaciones asociadas a él y tendremos que seleccionar si estamos de acuerdo o no. Para ello hemos utilizado una librería externa también, llamada "MaterialDialog" que se usa como vemos aquí:

```
deleteTopic.setOnClickListener(new View.OnClickListener() {
   public void onClick(View v){
      final MaterialDialog dialogTopic = new
MaterialDialog.Builder(DialogActivity.this)
            .title(R.string.app_name)
            .content(R.string.deleteTopic)
            .positiveText("Cancelar")
            .negativeText("Eliminar Tema")
            .negativeColorRes(R.color.material_red_500)
            .show();
```

```
        View negative = dialogTopic.getActionButton(DialogAction.NEGATIVE);
        negative.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                dialogTopic.dismiss();
                setContentView(R.layout.topics_subject_list);
                topicsSubject();
            }
        });
    }
});
```

De arriba abajo, encontramos que inicializamos nuestro "MaterialDialog" dándole por parámetro la actividad a la que va a estar asociada, después le indicamos también su título, contenido, el texto que debe salir en la opción positiva, en la negativa, el color que queremos que tenga la opción negativa y, finalmente, indicamos que se muestre mediante "show()". Los parámetros que se han puesto de "R.string" son cadenas de texto que están inicializadas en un archivo concreto para los de tipo "string" y no repetir una y otra vez el mismo valor, además de ayudarnos a la hora de traducir la aplicación a diferentes idiomas.

Si pulsáramos en eliminar tema, la opción negativa; se ejecutaría la siguiente parte de código que haría que el diálogo se cerrase con la llamada al método "dismiss()" y cambiaría el contenido de la vista del menú de opciones a la lista de asignaturas; y con "topicsSubject" controlamos las acciones con los botones que se mostrarían ahí.

En la siguiente imagen podemos ver cómo queda este fragmento de código en un dispositivo real:

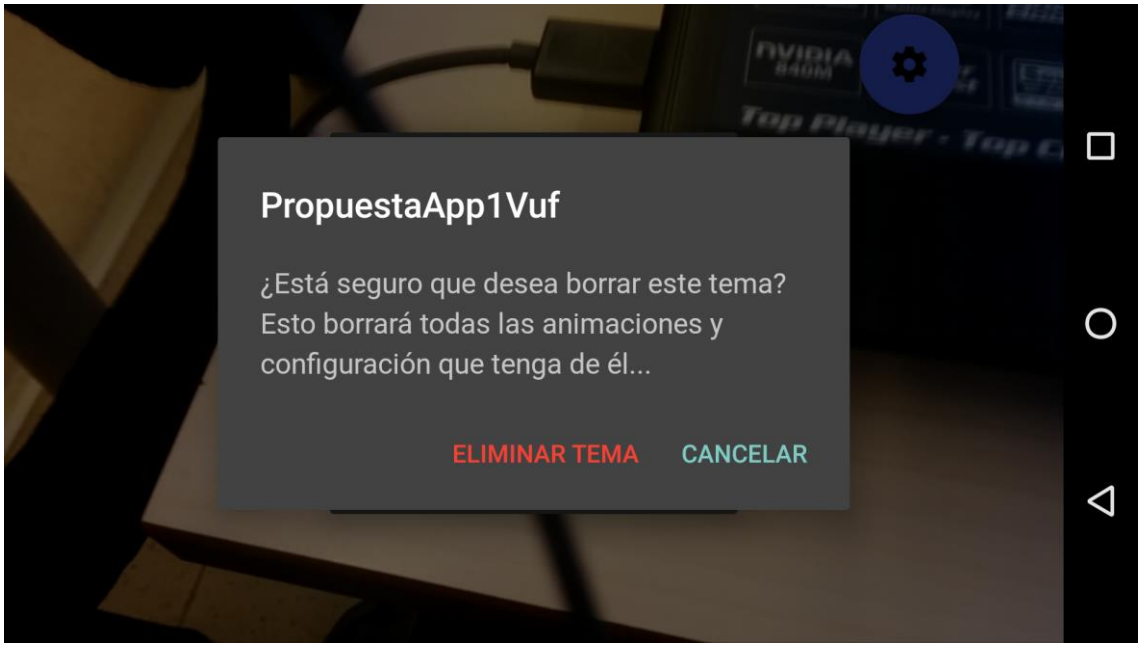

Ilustración 74 - Ejemplo de "MaterialDialog" en la aplicación Android

Principales problemas encontrados

No nos hemos encontrado grandes impedimentos a la hora de hacer el desarrollo, aunque los pocos problemas que nos han seguido en el procedimiento han sido de gran trascendencia. Podemos encontrar una pequeña lista de estos problemas a continuación:

- Para añadir el botón azul o "FloatingButton" con la librería externa, tuvimos bastantes problemas porque no podíamos modificar la clase que generaba Unity, ya que dejaba de funcionar la aplicación. Tuvimos que crear una clase nueva propia e ir encontrando los fragmentos de código con la configuración de la cámara para que funcionara correctamente. En la primera versión, la cámara solo reconocía un cuarto del tamaño real de la pantalla. Una vez tuvimos esa nueva clase creada y funcional, ya no tuvimos más problemas relacionados.
- Antes de crear el diseño del diálogo que aparece en el centro de la pantalla para el menú de opciones, intentamos hacerlo con una barra lateral típica en muchas aplicaciones de Android, llamada "Navigation Drawer", y nos percatamos de que para lo que nosotros queríamos hacer no era posible, ya que no se puede modificar el contenido sin crear una nueva barra e ir abriéndola y cerrándola en cada clic, lo que podía llegar a ser bastante tedioso y optamos por el diseño centrado de un diálogo a la espera de que en las entrevistas con los usuarios estos comentaran sus impresiones y al final les gustó el nuevo diseño y lo mantuvimos hasta la versión final con pequeñas variaciones.

- En nuestra búsqueda para ver cómo podríamos actualizar la base de datos de la aplicación sin tener que actualizar la propia aplicación, nos encontramos con que Vuforia genera sus "targets" u objetivos a rastrear en binario y que van incluidos dentro de la aplicación. A su vez Unity, con el que añadimos los recursos AR y las animaciones, también genera el código en binario y no podemos acceder a él, por lo que no es fácil hacer que la aplicación pudiera descargar esos datos aparte. Ya que el proceso completo pasaría por subir los "targets" que queremos a la web de Vuforia, añadirlos a nuestro proyecto en Unity, añadirle los recursos AR y las animaciones, exportar el proyecto para Android Studio y ahí podemos obtener la carpeta "Assets" que es en la que se incluyen todos los datos. Una vez tenemos esta carpeta, podemos o bien copiarla en nuestro proyecto con la aplicación ya hecha (funcionaría correctamente) o bien indicarle a la aplicación de alguna forma que lo descargara completo de algún servidor y encontrar la manera de que la aplicación obtuviera de ahí los datos.

Proyectos exportados con Metaio Creator

Los proyectos que exporta Metaio Creator, tienen la estructura antigua que se utilizaba con Eclipse ADT, que ya se considera obsoleto y cuyos plugins no son actualmente actualizados por Google en favor de Android Studio, el nuevo entorno oficial. Android Studio se encarga de adaptar los proyectos con la estructura antigua a la que utiliza él.

Tras convertir el proyecto exportado por Metaio Creator en Android Studio, podemos encontrarnos con la jerarquía que se vemos en la siguiente imagen:

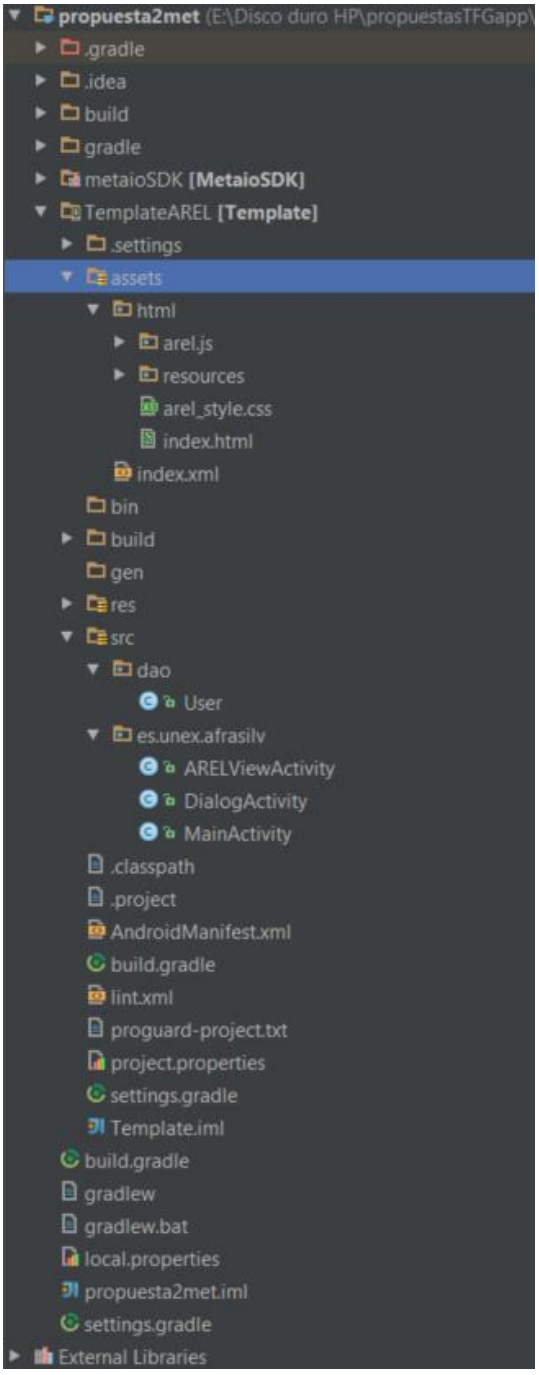

Ilustración 75 - Estructura de un proyecto exportado por Metaio en Android Studio

En la Ilustración 75 - Estructura de un proyecto exportado por Metaio en Android Studio, podemos ver la jerarquía de carpeta de Metaio. De todas las carpetas que hay, las que nos interesan son las que descienden de "TemplateAREL", ya que será nuestro directorio de aplicación. Inmediatamente superior a ella, se encuentra "metaioSDK" con un proyecto que se usa a modo de librería para Metaio.

Otra vista diferente de la arquitectura del proyecto en Android Studio la podemos ver en la siguiente imagen de forma esquemática:

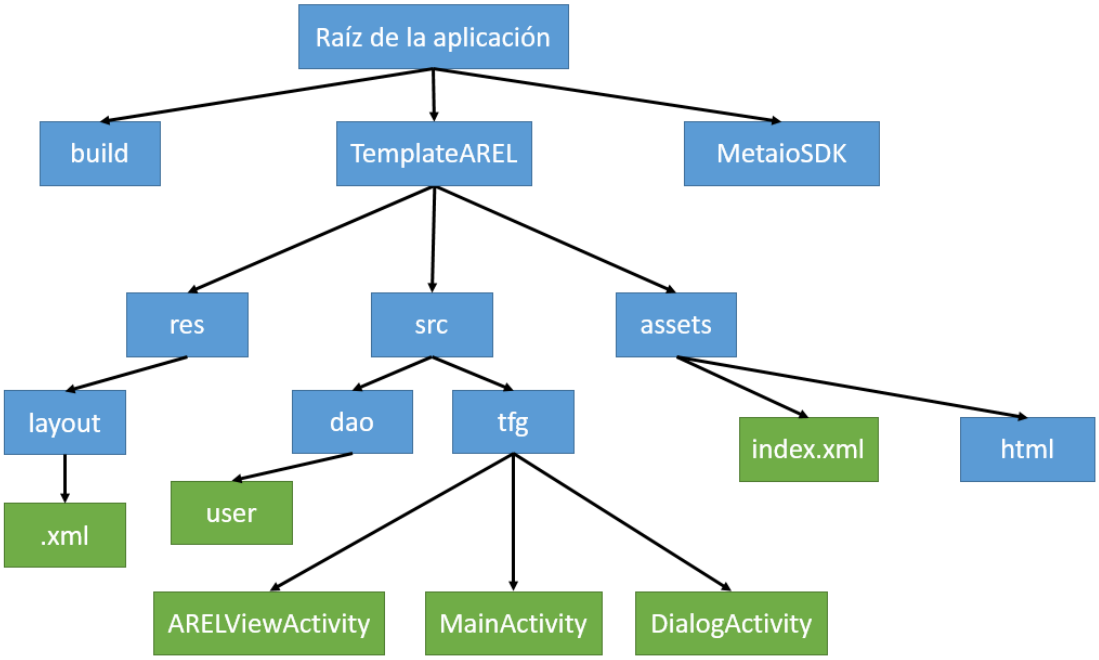

Ilustración 76 - Esquema de la estructura de un proyecto exportado por Metaio en Android Studio

Si comparamos esta ilustración con la Ilustración 75 - Estructura de un proyecto exportado por Metaio en Android Studio, podemos ver que nos muestran el mismo contenido, pero con un poco de ayuda de los colores podemos ver los archivos más relevantes y/o que hemos creado nosotros. De nuevo, en la carpeta "layout" podemos ver los ficheros "xml" con las diferentes vistas de cada opción del menú de opciones entre otras. Dentro de la carpeta "dao", vemos la clase "user" para un futuro control de usuario y en la carpeta "tfg" las dos clases principales "MainActivity" y "ARELViewActivity", generadas por Metaio y "DialogActivity", que es nuestra. También podemos ver la carpeta "assets" en el mismo nivel de la jerarquía que las carpetas de recursos y código.

Si continuamos por "TemplateAREL", nos encontramos con una carpeta seleccionada, "assets", que al igual que en el proyecto de Unity, contiene los ficheros relacionados con los "targets" de Metaio, el cual se compone de dos carpetas: "arel.js" en la que encontramos varios ficheros en formato "JavaScript" para reconocer los "targets" y añadirle los recursos AR que hemos añadido a cada uno de ellos, y "resources", en la que dentro de ella encontramos una carpeta por cada "target" que hemos hecho que Metaio reconozca y que contiene los recursos AR que le hayamos dicho, y un fichero ".zip" fuera con la relación del "target" y sus recursos AR y una parte de la imagen del "target" asociado.

Ya fuera de “assets”, podemos ver las carpetas “res” con los recursos que va a utilizar la aplicación y “src”, que contiene el código de nuestra aplicación. En esa carpeta vemos sólo 3 archivos o clases Java, “DialogActivity”, creada por nosotros, y “ARELViewActivity” y “MainActivity”, que fueron generados por Metaio.
En el fichero “MainActivity” se cargan los “assets” en segundo plano y una vez termina de cargarlos invoca a la clase “ARELViewActivity” para que se ejecute.
La carpeta “ARELViewActivity”, por defecto se encuentra vacía, pero en esta clase se puede añadir información a la pantalla mediante una vista o “layout” que se le pueda añadir; nosotros le hemos añadido el botón de opciones y la acción de que si lo pulsas se ejecute nuestra “DialogActivity” con toda la interacción. La única condición para que se ejecute correctamente es que en el “layout” haya un “RelativeLayout” o “vista relativa” que ocupe toda la pantalla, dentro de ese padre nosotros pondremos nuestro botón como su hijo y además en vez de ponerle un “listener” o escuchador de eventos como en el ejemplo de Unity, le diremos en la propia vista que ejecute el método que nosotros le indicamos como vemos a continuación:

```
<ImageButton
   […]
   android:clickable="true"
   android:onClick="onClickSettings"></ImageButton>
```

En el fragmento superior de código, vemos que le tenemos puesto a “true” o verdadero que se pueda pulsar y que, si se pulsa, con la opción “onClick”; se ejecute el método “onClickSettings”, un método público de la clase “ARELViewActivity” y creado por nosotros.
Para poder utilizar el SDK de Metaio, tendremos que indicarle la licencia dentro del código de Android Studio; para ello localizaremos el fichero “signature.xml” dentro de la carpeta “res/values” de nuestro proyecto. Una vez dentro veremos que sólo hay una cadena de texto con un valor que hay que modificar. Para obtener la licencia, nos iremos a la web del portal del desarrollador de Metaio[57]. Una vez hayamos iniciado sesión, nos iremos a la pestaña “My Apps” y en la parte inferior registraremos nuestra aplicación con su nombre y su identificador de paquete. Cuando esté registrada, pulsando sobre el nombre que aparecerá debajo, nos dará las licencias para Metaio 5 y 6. Nosotros

[57] https://my.metaio.com/index.php?state=portal - Última visita: 29/08/15

utilizaremos la versión 6 de Metaio y pegamos el valor de la licencia en el archivo "signatures.xml" que hemos comentado anteriormente.

Unas capturas de cómo queda la aplicación con el código que le hemos añadido son:

Ilustración 77 - Pantalla inicial de la aplicación exportada por Metaio con el botón de opciones

Ilustración 78 - Menú de opciones de la aplicación exportada por Metaio

Una de las cosas que podemos ver en las capturas comparándolas con las de la misma propuesta de aplicación pero exportada por Unity, es que el botón de opciones es diferente, ya que las diferentes librerías que hemos probado para poner el botón circular y con la funcionalidad de las últimas versiones de Android no eran compatibles con Metaio. Otra parte diferente del código es la relacionada con la forma de generar los diálogos, ya que la librería que estábamos usando en las propuestas con Vuforia y Unity

no es compatible tampoco con nuestro proyecto de Metaio. Hemos utilizado "AlertDialog"[58], que viene por defecto en el SDK de Android.
Con respecto al código, son muy similares de utilizar; aquí podremos ver un ejemplo de la misma llamada en las diferentes propuestas, empezaremos por la de esta propuesta, en la que usamos "AlertDialog":

```
about.setOnClickListener(new View.OnClickListener() {
  public void onClick(View v){
    AlertDialog.Builder builder = new AlertDialog.Builder(DialogActivity.this,
R.style.AlertDialog_AppCompat);
    builder.setTitle(R.string.app_name);
    builder.setMessage(R.string.content);
    builder.setPositiveButton("Aceptar", null);
    builder.show();
    }
});
```

Y a continuación el mismo ejemplo con la librería de "MaterialDialog":

```
about.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v){
        new MaterialDialog.Builder(DialogActivity.this)
            .title(R.string.app_name)
                .content(R.string.content)
                .positiveText("Aceptar")
                .show();
    }
});
```

Como podemos ver, "MaterialDialog" es mucho más limpio y hay que poner menos código para una llamada normal. Además, nos permite un abanico más grande de personalización del diálogo.

[58] http://developer.android.com/reference/android/app/AlertDialog.html - Última visita: 29/08/15

En el ejemplo, le indicamos al diálogo después de su creación con la llamada “new …” el título que debe tener, el contenido con “setMessage” en “AlertDialog” y “.title” en “MaterialDialog”, el texto que queremos que tenga en el botón con valor positivo y que se muestre.

Principales problemas encontrados

Con Metaio nos hemos encontrado varios problemas que, si bien algunos no tienen solución por las propias limitaciones impuestas a las licencias gratuitas y otros no los hemos podido solucionar de una forma lo suficientemente estética, comentaremos cuáles eran:

- El primer problema con el que nos encontramos fue el intento de poner un botón de opciones al igual que para su versión de Unity. Tras probar varias librerías diferentes para añadir el botón circular, todas daban error de compatibilidad con Metaio y tuvimos que desecharlas todas y optar por el botón cuadrado, que pese a cumplir su funcionalidad, no es tan estético como el circular con los nuevos estándares de diseño de Android.
- El siguiente problema está estrechamente relacionado directamente con el primero y aparece a la hora de añadir el botón a la interfaz, ya que no se podía añadir a la vista de la cámara de Metaio directamente porque la cámara dejaba de funcionar. Tras documentarnos más sobre el problema y sus posibles soluciones e ir probándolas, al final conseguimos añadir el botón y el evento de hacerle clic.
- Otro problema con las librerías que utilizamos con Unity y Vuforia fue la de “MaterialDialog”, que no es compatible con Metaio. Solventamos este problema utilizando la clase “AlertDialog” incluida en el SDK de Android, pero que nos permite menos posibilidades de personalización.
- Al igual que en Unity, tuvimos que cambiar el diseño de la “NavigationDrawer” por el hecho de que no se puede modificar el contenido sin crear una barra lateral nueva y reabrirla.
- En la búsqueda de añadir más información a la base de datos de la aplicación sin tener que actualizar la propia aplicación, observamos que debería ser más fácil en Metaio que en Unity, ya que los archivos no están en binario; salvo los de las animaciones asociadas a los recursos AR, todos los demás tienen formatos que pueden ser leídos sin complicación. Además, en la carpeta que comentamos al principio que contenía los archivos “JavaScript” que se encargaban de reconocer

los “targets” y mostrar los recursos AR, la carpeta “arel.js”, podemos ver y modificar los archivos que cargan los “targets” y ponerle los datos de otros para que los cargara, pero no funcionó ninguna de las ocasiones pese a haber cambiado todos los datos asociados con los “targets” ya existentes por nuevos, por lo que o bien no cargan los nuevos, por las limitaciones del SDK con licencia gratuita, ya sea porque faltarían más campos por cambiar o porque no es posible hacerlo como lo intentábamos nosotros.

Resumen del desarrollo en Android Studio

El desarrollo de las diferentes aplicaciones en Android Studio ha sido el mismo, en términos de explicación, no de código. La estructura de qué se ha hecho en Android Studio la podemos ver a continuación:

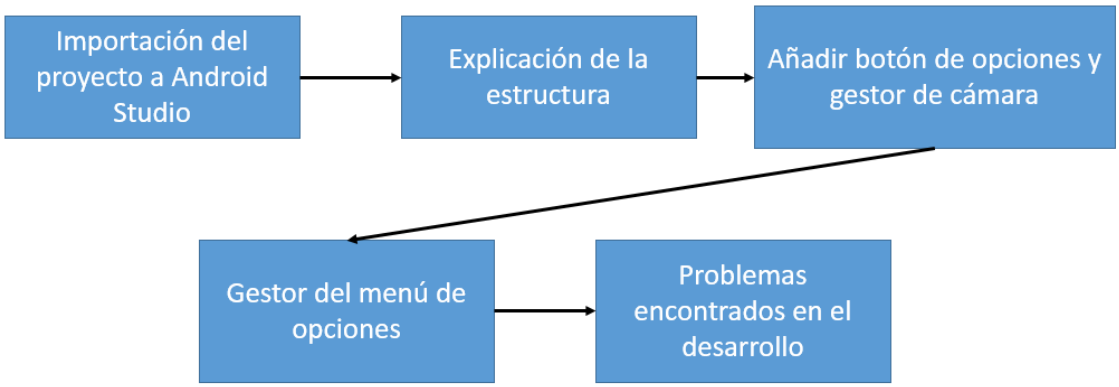

Ilustración 79 - Resumen de los desarrollos en Android Studio

En los diferentes desarrollos hemos comenzado por importar el proyecto para Android Studio, algunos incluso a convertirlos, ya que usaban versiones anteriores, pero este proceso se ha hecho automáticamente. Hemos continuado comentando la estructura de las carpetas y qué había en cada una de ellas que nos pudiera resultar relevante para nuestro desarrollo.

A continuación, hemos explicado cómo se ha añadido el botón de opciones que podemos ver en las diferentes aplicaciones, como por ejemplo podemos apreciarlo en la Ilustración 74 - Ejemplo de "MaterialDialog" en la aplicación Android en la parte superior derecha y cómo se ha realizado el proceso para que cargara correctamente las cámaras de los SDKs y que mostrara el botón a la vez.

Se ha procedido a explicar el código del menú de opciones y cómo funcionaba y por último, una breve lista en la que se explican los problemas que han tenido lugar durante el desarrollo con las explicaciones de cómo se han solucionado o de por qué no ha sido posible hacerlo.

Comparativa SDK durante el desarrollo

A continuación veremos una pequeña tabla comparativa de los diferentes SDKs que hemos probado y las características que hemos podido observar en el proceso. En la tabla, comprobaremos si la licencia era gratuita, las limitaciones que tenía la herramienta, si admite modelos 3D externos, si se ha podido añadir modelos aparte sin tener que recompilar la aplicación y si es compatible con el formato de imagen “gif”.

	Licencia Gratuita	Limitaciones	Admite modelos externos	Añadir modelos aparte sin recompilar	Compatible con GIF
Vuforia	**SI**	• 1000 peticiones de Targets a la nube. • 1000 Targets en la nube. • 100 Targets máximo en el dispositivo.	**SI*** **(**Bender mejor)	**NO**	**NO**
Metaio	**SI** [59]	• 2 animaciones a la vez. (con 1 movimiento cada una). • 6 recursos AR a la vez sobre el Target. • 2 Targets máximo por aplicación con la licencia gratuita. • Si tienes 2 targets, sólo puedes añadir 3 recursos AR a la vez en total.	**SI (**Bender Object mejor)	**NO**[60]	**NO**
ARMedia	**NO**				
Wikitude	**SI**	**NO (Sólo marca de agua)**	Tiene un Software propio para transformar los principales (.dae y .fbx) a un formato propio privado (.wt3)	**SI****	**NO**[61]

Tabla 6 - Comparativa SDK durante desarrollo de aplicaciones

[59] Desde la compra de la empresa por parte de Apple INC, ya no se pueden crear nuevas licencias que amplíen la gratuita.
[60] Se ha intentado modificar el código JavaScript que genera Metaio en Android Studio para intentar que cogiera todos los modelos que hay en la carpeta de recursos y es posible que se pueda, pero da bastantes problemas y al final no se ha conseguido.
[61] Tras haber hablado con algunos trabajadores de la empresa, nos han informado de que tienen puesto en su hoja de ruta como una de sus futuras principales funciones la de ser compatible con las imágenes en formato ".gif".

Vuforia tiene una licencia gratuita, como podemos ver en la Tabla 6 - Comparativa SDK durante desarrollo de aplicaciones, pero esa licencia tiene unas limitaciones u otras en función de dónde queramos almacenar los "Targets" que queremos que la aplicación reconozca. Si se almacenan en la nube, podemos tener 1000 y cada uno con 1000 peticiones; si se almacenan en el dispositivo, tenemos un máximo de 100 "Targets" por base de datos, aunque podemos tener varias bases de datos. *En lo referente a si admite modelos externos o no, podemos decir que sí los acepta, pero no con los formatos comunes. Con los formatos estándar como son Collada (.dae) o AutoDesk FBX (.fbx) podemos trabajar desde Unity 3D, porque él sí los acepta, pero para trabajar con Vuforia a la hora de generar la aplicación los reconvierte a modelos en OpenGL, por lo que realmente estamos trabajando con OpenGL y no con un lenguaje simple para alguien que no ha trabajado con él, aunque se pueden desarrollar scripts que portaran los modelos hechos con Blender o SketchUp a OpenGL para poder trabajar con ellos, pero sólo el modelo, sin animaciones de ningún tipo que habría que crear a mano con el inconveniente además de que estos ficheros los genera en binario, que tras intentar en Android Studio su modificación para mostrar diferentes "targets" o añadir más sin recompilar la aplicación, no fue posible sin cambiar por completo la carpeta "assets".

Hemos probado también con el formato ".gif" desde Unity 3D para mostrar imágenes animadas y no es compatible, se queda fija la8 imagen.

Metaio también tiene una licencia gratuita si te habías registrado antes de que fuera comprada por Apple Inc. y te habías registrado en la web) pero también tiene varias limitaciones a la hora de desarrollar una aplicación como es el hecho de que en el software propio que tienen para trabajar no se pueden poner más de 2 animaciones a la vez y cada una referente bien a cuando el "Target" sea reconocido o cuando pulses sobre el recurso de AR. Además, sólo podemos tener 6 recursos AR por cada "Target", lo cual es bastante, pero en función de lo que queramos hacer es un límite a tener en cuenta. Y el más restrictivo de todos es que no puedes tener más de 2 "Targets" por aplicación, por lo que no nos sirve para la aplicación que queríamos desarrollar, ya que no se puede comprar actualmente una licencia. También hemos tratado de modificar el contenido de su base de datos, que pensábamos en un principio que sería más fácil por no ser en binario y tras probar a modificar el contenido de varias formas diferentes no conseguimos que cargara la base de datos que le intentábamos poner, quizás porque reconocía que era diferente y saltaba la limitación de SDK de no usar más de 2 "targets".

ARMedia, por otro lado, te permite descargar su SDK y empezar a desarrollar lo que quieras con él, pero cuando quieres probar a ejecutarlo tienes que poner una clave de licencia de uso, como en todos los otros SDK. Para conseguir esta licencia, hay que estar suscrito a la página, que ronda de los 1000 a 3500€. Se ha descartado en ese momento.

Wikitude en un principio no tiene limitaciones, tiene las mismas funcionalidades que la versión PRO, pero con una marca de agua en la cámara que no se puede quitar. Para los modelos externos, tiene un software propio que te permite importar y cambiar los modelos de Collada (.dae) y AutoDesk FBX (.fbx) a un formato propio de Wikitude que es ".wt3" en binario, por lo que su contenido no puede ser visto. **En lo que respecta a la importación de modelos externos sin recompilar la aplicación, la carga de los "Targets" a reconocer y de los recursos a mostrar se realiza con JavaScript sobre un navegador web incorporado, lo que nos permite bastante flexibilidad a la hora de trabajar con él. Es posible que con JavaScript se puedan recuperar los ficheros de los diferentes "Targets" y sus recursos AR sin tener que recompilar la aplicación, hacer algo genérico y que la aplicación lo reconozca. Hemos probado también con el formato ".gif" para mostrar imágenes animadas y no es compatible, se queda fija la imagen. Con respecto a Wikitude, se ha estado en contacto por e-mail con trabajadores de la empresa, y su respuesta al correo ha sido que en las versiones actuales no dan soporte a imágenes "gif", pero que está en su hoja de ruta y en un futuro muy cercano lo van a añadir a su SDK. Además, nos comenta que para añadir contenido extra a la aplicación sin tener que recompilarla hay que subirla al servidor de Wikitude y así poder cambiar los recursos AR y dejar la aplicación sin tocar.

Como conclusiones de todo el proceso de desarrollo, nos quedaríamos con las herramientas de Vuforia y Wikitude.

Con Vuforia porque a pesar de no haber conseguido modificar la base de datos es con la que mejores resultados hemos obtenido y actualmente una de las que más se usa y se desarrolla con licencia gratuita. Desde que comenzamos el trabajo hasta que lo hemos finalizado han actualizado en varias ocasiones su SDK para añadir varias mejoras, aunque ninguna que nos haya servido en nuestras propuestas. Además, tienen un foro de usuarios muy activo y una gran documentación de todas las características que ofrecen.

Con Wikitude porque a pesar de que no utilice los formatos estándares de modelos para los recursos AR, hemos visto que se puede modificar el contenido de la aplicación también y es posible que pudiéramos modificar la base de datos de los "targets" a

nuestro antojo y añadirle más contenido y utilizar las características que teníamos en el diseño de "activar o desactivar" y lanzar un nuevo "target" en un momento concreto. Por último, también nos ha parecido bastante interesante que tengan en su hoja de ruta el soportar el formato ".gif" para las imágenes animadas, ya que para nuestra propuesta haría que cualquier persona pudiera generar una animación sin tanta complicación de crear los recursos, tener en cuenta su posición en los tres ejes de coordenadas ni configurar la animación de cada uno de los recursos AR de ese "target".

Planificación final

Tras la primera planificación en la que pensábamos que tendríamos todo preparado para la segunda quincena de mayo y listo para entregar y presentar en junio, por diferentes problemas con la documentación a la hora de encontrar algunas licencias de los SDK, cómo realizar las animaciones en Unity o Metaio o ya en Android Studio intentando que la aplicación funcionara bien y comprobar si se podían modificar los datos sin actualizar fue casi imposible tenerlo preparado para la entrega de julio. Pero ya teniéndolo todo casi terminado y preparado, la nueva planificación es tenerlo todo terminado, preparado y funcional para antes de la segunda quincena de agosto.

Si recordamos la tabla que había en la planificación inicial, Tabla 1 - Planificación inicial; ahora la compararemos con la final para ver la desviación temporal que ha tenido lugar:

Actividad	Inicio esperado	Duración esperada	Inicio final	Duración final	Desviación
Historia AR	18 de febrero de 2015	3 días	18 de febrero de 2015	2 días	**-1 día**
Comparativa SDK AR	20 de enero de 2015	30 días	20 de enero de 2015	35 días	**+5 días**
Diseño Centrado en el Usuario y propuestas de aplicación	11 de marzo de 2015	20 días	29 de marzo de 2015	27 días	**+7 días**
Desarrollo de los prototipos sin Android Studio	1 abril de 2015	50 días	29 de mayo de 2015	67 días	**+17 días**
Desarrollo de los prototipos en Android Studio	20 de mayo de 2015	7 días	14 de junio de 2015	15 días	**+8 días**

Tabla 7 - Planificación final y desviación de planificación

Para la realización de la tabla se han tenido en cuenta las fechas aproximadas de inicio y fin de cada actividad, ya que algunas se realizaban de forma simultánea porque estaban relacionadas y se han restado de la duración final todos los días en los que no se ha trabajado en las actividades.

Podemos sacar como conclusión de la tabla el hecho de que nos hemos quedado cortos a la hora de realizar la planificación inicial en todas las tareas menos en la primera, la "Historia de la AR" y que hemos tenido una gran desviación con respecto a lo previsto en la planificación inicial.

Como se ha podido ver en la tabla, los plazos previstos para la versión final varían en aproximadamente dos meses por falta de tiempo que dedicar al trabajo para llegar a la entrega de junio y por problemas finales y correcciones de la documentación llegábamos muy justos a la fecha de entrega de la convocatoria de julio. Optamos por asegurar que nos diera tiempo a la realización de las correcciones y a la estabilidad de los prototipos y presentarlo todo en la convocatoria de septiembre.

Proyectos relacionados en desarrollo

Durante el desarrollo del proyecto, se ha trabajado también en otros proyectos propios relacionados con la temática del mismo y con los que gracias a ellos hemos podido comprobar que la metodología de DCU agiliza mucho el avance del diseño de una aplicación y que la RA puede ayudar a expandir los conocimientos sobre una materia y el público al que puede llegar.

Proyecto con metodología DCU: Guía de Olivenza

La aplicación "Guía de Olivenza[62]", fue desarrollada a finales del mes de julio de 2015 y todo el diseño de la aplicación fue probado por los usuarios desde comienzos de su desarrollo hasta la última versión publicada. La aplicación muestra información sobre los servicios que tiene la ciudad con su información de contacto, las farmacias disponibles en la ciudad y cuál se encuentra actualmente de guardia, una galería de imágenes de la ciudad en constante actualización y empresas locales que deseen publicitarse.

El usuario principal de la aplicación, está en torno a los 40 años ya que sería el que más podría necesitar los servicios actualmente disponibles en la aplicación. Al igual que en las propuestas de este trabajo, se realizaron varias entrevistas con los diferentes usuarios para llegar a la versión final.

A continuación podremos echar un pequeño vistazo a parte de la evolución de la aplicación:

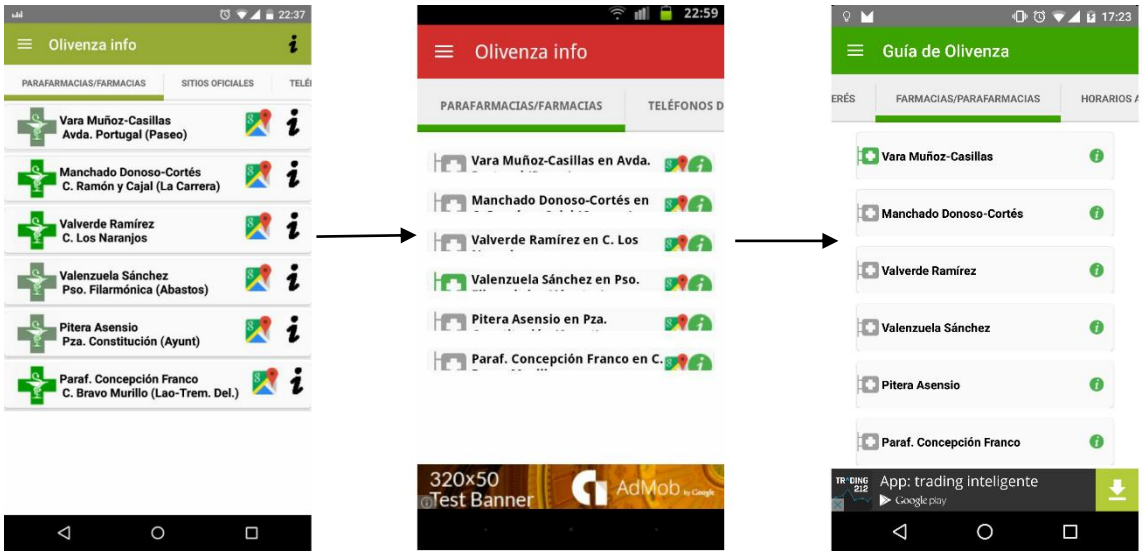

Ilustración 80 - Evolución de la aplicación entre 3 versiones diferentes de la misma información

[62] https://play.google.com/store/apps/details?id=afrasilv.es.olivenzainfo – Última visita: 29/08/15

Entre las tres versiones diferentes puestas arriba, se puede observar a grandes rasgos que en la parte superior, antes la vista de la información “Parafarmacias/Farmacias” ocupaba la primera posición en la lista del contenido de la aplicación y además que se ha cambiado el orden a “Farmacias/Parafarmacias” para dotar de más importancia a las farmacias. También podemos ver una clara evolución de los iconos a otros que muestran con un diseño más limpio y moderno, cuál sería la que está de guardia y nos invita a pulsar sobre el cuadro para mostrarnos más información.

Como podemos ver, son pequeños detalles que el usuario le puede dar más importancia que nosotros o que nosotros los pasamos por alto pero el usuario final considera importantes. Gracias a los usuarios hemos conseguido una aplicación con una interfaz propia de las últimas directrices de Google en términos de diseño.

Proyecto de Realidad Aumentada: Monumentos Olivenza

Este proyecto aún se encuentra en una fase temprana del desarrollo. La idea de la aplicación es que permita a turistas y ciudadanos ver los monumentos que hay en la ciudad y algunos que han sido derribados, en cualquier momento y en cualquier parte desde algunas imágenes vinculadas a recursos AR o utilizando los mosaicos de piedra que hay repartidos por el suelo de la ciudad en los diferentes puntos turísticos.

La aplicación ha sido desarrollada con el SDK de Vuforia, usando Unity 3D para asociar Vuforia a los diferentes recursos AR que queremos asociar y los modelos 3D han sido facilitados por una tercera persona.

En las siguientes imágenes, podremos ver los modelos 3D que hay en la aplicación en este momento:

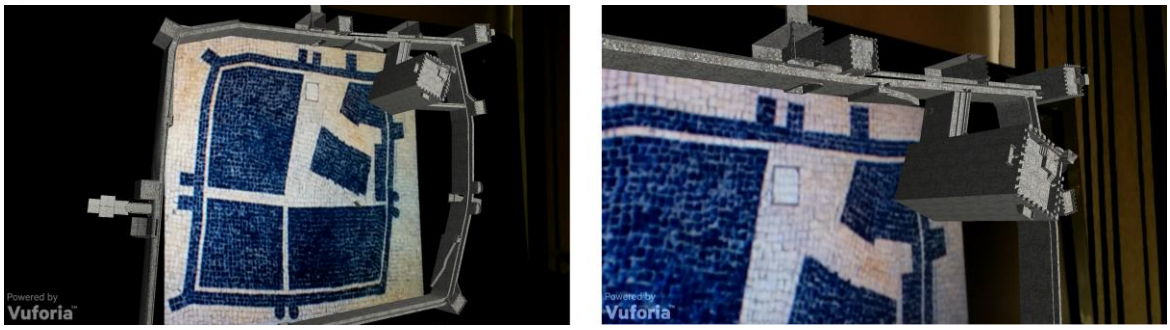

Ilustración 81 - Ciudadela y primera muralla de Olivenza con un mosaico de piedra asociado

Ilustración 82 - Puerta de San Francisco en Olivenza derribada a principios del S. XX

Como podemos ver en las imágenes, actualmente la aplicación muestra sólo el recurso AR asociado a una imagen, se está trabajando para añadir más, asociarlos a coordenadas GPS y añadirle alguna animación.

Esta aplicación, mostraría parte de la historia perdida de Olivenza, ya que parte de los antiguos monumentos ya no se pueden encontrar, además de que es fácilmente exportable a otras ciudades y municipios con gran interés histórico y turístico en cuanto se encuentre finalizada.

Conclusiones

Una de las conclusiones que podemos sacar del trabajo es que la realidad aumentada está ahora en un gran auge gracias a la potencia de los dispositivos móviles, que no para de crecer, y la facilidad que hay en la actualidad para distribuir una aplicación a una gran comunidad de usuarios, por lo que podemos asegurar que le esperan grandes avances a corto plazo. No hay más que ver el último dispositivo presentado por Microsoft, las HoloLens; que aunque hayan llegado tarde han dado un golpe sobre la mesa que ha hecho a Google retirar sus gafas para sacar una versión mejorada.

También la variedad de empresas que hay desarrollando sus SDKs para poder hacer el trabajo de una manera más fácil, cómoda y rápida está ayudando en gran medida a este crecimiento, como Vuforia, que es con la que más hemos tenido la oportunidad de trabajar y la que mejores resultados nos ha dado. Durante la realización del trabajo, Vuforia y Wikitude han actualizado sus SDKs en varias ocasiones, ofreciendo mejoras en el rendimiento y nuevas funciones. Esta última por ejemplo, ha publicado su nueva versión 5.0 con grandes novedades el pasado 27 de agosto de 2015[63].

Además, podemos encontrar empresas que trabajan más con software pero que han querido desarrollar hardware con dispositivos como las Google Glass de Google, Microsoft con las HoloLens y Oculus (Facebook) con las Oculus Rift, aunque estas estén más pensadas para realidad virtual son algunos de los ejemplos de estos nuevos dispositivos.

Por otro lado, vemos como el diseño centrado en el usuario o DCU hace que los desarrollos de los proyectos, sea cual sea su naturaleza; sean más largos al principio pero te permite crear un producto más acorde al usuario al que intentas conseguir, ya que al relacionar al usuario desde el comienzo del desarrollo se familiariza con la aplicación y sabrá usarla por intuición y nosotros conseguiremos así un producto mejor para la tarea y no tendremos que cambiar tantas veces el diseño como si lo hubiéramos terminado sin contar con el usuario y su opinión como retroalimentación o “feedback”. Ejemplos de estos desarrollos los podemos encontrar en las grandes empresas tecnológicas como Apple, Google o Microsoft que, antes de publicar la versión final de sus sistemas operativos, ofrecieron un sistema de versiones previas en la que pedían siempre la opinión del usuario en términos de diseño y estabilidad. Y, gracias a ellos, han podido mejorar mucho sus productos antes de publicarlos.

[63] http://www.wikitude.com/download - Última visita: 29/08/15

Por último, como se ha podido ver en el apartado “Proyectos relacionados en desarrollo”, nosotros tenemos publicada una aplicación y otra en desarrollo en la que utilizamos los conocimientos aprendidos en el presente trabajo. Gracias a ellos hemos podido hacer una aplicación que se ha adaptado a los deseos de un grupo de usuarios que consideramos como principales y nos ha facilitado el trabajo a la hora de modificar la estructura y el diseño de la aplicación, así como la curva de aprendizaje de los usuarios. Con respecto a la aplicación de AR, gracias a estos conocimientos podemos ver una aplicación propia que se puede llegar a usar en entornos reales y que puede ser de gran utilidad para los usuarios.

Bibliografía

- Cooper, A. (1999). The inmates are running the asylum. Indianapolis, Ind., Sams; [Hemel Hempstead: Prentice Hall] [distributor].
- Preece, J. (1994). Human-computer interaction. Wokingham, Addison-Wesley.
- Rubin, J. J. (1994). Handbook of usability testing: how to plan, design, and conduct effective tests. New York; Chichester, Wiley.
- Historia de la Realidad Aumentada Móvil por el Instituto de Gráficos y Visión por ordenador. Fecha de última consulta: 25 de agosto de 2015. Actualizada: abril de 2015. Disponible en: http://www.icg.tugraz.at/publications/the-history-of-mobile-augmented-reality .

1. Video de ejemplo de aplicación de McDonald's realizado con Vuforia. Fecha de última consulta: 18 de febrero de 2015. Actualizada: 24 de julio de 2014. Disponible en: https://www.youtube.com/watch?v=0l14awnnyqg .
2. Realidad Aumentada por el Secretariado de Recursos Audiovisuales y Nuevas Tecnologías en la Universidad de Sevilla. Fecha de última consulta: 24 de agosto de 2015. Actualizada: 30 de julio de 2015. Disponible en: http://realidadaumentada.us.es/ .
3. "A Survey of Augmented Reality" por Ronald T. Azuma como definición de Realidad Aumentada. Consultada página 2. Fecha de última consulta: 24 de agosto de 2015. Documento de: agosto de 1997. Disponible en: http://www.cs.unc.edu/~azuma/ARpresence.pdf .
4. Artículo sobre el primer sistema de Realidad Aumentada por Ivan Sutherland con título "A head-mounted three dimensional display". Fecha de última consulta: 18 de febrero de 2015. Fecha de publicación: 9 de diciembre de 1968. Disponible en: http://dl.acm.org/citation.cfm?id=1476686 .
5. Creación del término "Realidad Aumentada" por Tom Caudell. Fecha de última consulta: 18 de febrero de 2015. Fecha de publicación: enero de 1992. Disponible en: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=183317&isnumber=4717&punumber=378&k2dockey=183317@ieeecnfs .

6. Presentación de los marcadores de matriz 2D por Jun Rekimoto. Fecha de última consulta: 18 de febrero de 2015. Fecha de publicación: julio de 1998. Disponible en: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=704151&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D704151 .

7. Documentación de ARQuake, el primer juego al aire libre con dispositivos de AR por Bruce H. Thomas. Fecha de última consulta: 18 de febrero de 2015. Fecha de publicación: octubre de 2000. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.19.6863 .
8. Página oficial de Google Glass por Google. Fecha de última consulta: 18 de febrero de 2015. Actualizada: 26 de enero de 2015. Disponible en: https://www.google.com/glass/start/ .
9. Página oficial de HoloLens por Microsoft. Fecha de última consulta: 18 de febrero de 2015. Actualizada: enero de 2015. Disponible en: http://www.microsoft.com/microsoft-hololens/en-us .
10. Modelo de cráneo humano en aplicación AR por Mahei en colaboración con la Universidad de Navarra. Fecha de última consulta: 20 de enero de 2015. Actualizada: 2 de noviembre de 2012. Disponible en: http://mahei.es/iskull.php?os=ios&lang=es .
11. Vídeo de aplicación de simulación de soldaduras por Soldamatic. Fecha de última consulta: 20 de enero de 2015. Actualizado: 17 de septiembre de 2012. Disponible en: https://www.youtube.com/watch?v=TvgrG___wxY .
12. Artículo en blog de tecnología sobre la AR en la cirugía publicado por Jesús Maturana en Xataka. Fecha de última consulta: 20 de enero de 2015. Actualizado: 3 de octubre de 2013. Disponible en: http://www.xataka.com/medicina-y-salud/el-futuro-de-la-cirugia-segun-philips-y-accenture-pasa-por-google-glass .
13. Artículo en periódico nacional sobre la cirugía y Google Glass publicado La Vanguardia. Fecha de última consulta: 20 de enero de 2015. Actualizado: 21 de junio de 2013. Disponible en: http://www.lavanguardia.com/salud/20130621/54376177193/operacion-google-glass.html .

14. Guía de desarrollo para Google Glass sin tener las gafas por Ostap Andrusiv. Fecha de última consulta: 22 de enero de 2015. Actualizada: 1 de noviembre de 2013. Disponible en: http://elekslabs.com/2013/11/google-glass-development-without-glass.html .
15. Repositorio de código con las aplicaciones de Google Glass para cualquier dispositivo Android por "Zhouwei" y alojado en GitHub. Fecha de última consulta: 22 de enero de 2015. Actualizado: 6 de julio de 2013. Disponible en: https://github.com/zhuowei/Xenologer .
16. Guía de instalación y uso de las aplicaciones de Google Glass en cualquier dispositivo Android por "Zhouwei" y alojado en Imgur. Fecha de última consulta: 22 de enero de 2015. Actualizado: 3 de julio de 2013. Disponible en: http://imgur.com/a/IBqFf .
17. Guía de uso de la API no oficial de Google Glass por "Scarygami". Fecha de última consulta: 22 de enero de 2015. Actualizada: 1 de mayo de 2013. Disponible en: http://glass-apps.org/google-glass-emulator .
18. Página oficial de Google Glass Developers por Google. Fecha de última consulta: 29 de agosto de 2015. Actualizada: 28 de mayo de 2015. Disponible en: https://developers.google.com/glass/ .
19. Página web de simulación de tarjetas de Google Glass. Fecha de última consulta: 22 de enero de 2015. Disponible en: http://glasssim.com/ .
20. Código fuente de la librería ARToolKit publicado en SourceForge. Fecha de última consulta: 23 de enero de 2015. Disponible en: http://sourceforge.net/projects/artoolkit/ .
21. Página oficial de Human Interface Technology Laboratory (HIT Lab) de la Universidad de Washington que publicó ARToolKit. Fecha de última consulta: 20 de enero de 2015. Actualizada: 11 de marzo de 2008. Disponible en: http://www.hitl.washington.edu/home/ .
22. Página oficial de la comunidad de desarrolladores de Total Immersion (D'Fusion). Fecha de última consulta: 29 de agosto de 2015. Actualizada: 6 de agosto de 2015. Disponible en: http://www.t-immersion.com/partners/partners-programs/developer .
23. Listado de proyectos de Total Immersion. Fecha de última consulta: 29 de agosto de 2015. Actualizada: 5 de agosto de 2015. Disponible en: http://www.t-immersion.com/projects .

24. Página oficial de Metaio. Fecha de última consulta: 29 de agosto de 2015. Actualizada: 31 de julio de 2015. Disponible en: http://www.metaio.com/ .
25. Página oficial de Arpa Solutions. Fecha de última consulta: 23/01/2015. Disponible en: http://www.arpa-solutions.net/es .
26. Página oficial del proyecto NyARToolKit en japonés e inglés. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 27 de mayo de 2015. Disponible en: http://nyatla.jp/nyartoolkit/wp/ .
27. Página oficial de Wikitude. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 10 de agosto de 2015. Disponible en: http://www.wikitude.com/ .
28. Página oficial de ARmedia por Inglobe Technologies. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 24 de agosto de 2015. Disponible en: http://www.inglobetechnologies.com/our-products/ .
29. Página oficial de Layar. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 15 de julio de 2015. Disponible en: https://www.layar.com/features/ .
30. Página oficial del proyecto AndAR. Fecha de última consulta: 24 de enero de 2015. Disponible en: https://code.google.com/p/andar/ .
31. Página oficial de desarrolladores en Vuforia. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 29 de julio de 2015. Disponible en: https://developer.vuforia.com/ .
32. Página oficial de AprilTags. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 8 de julio de 2015. Disponible en: https://april.eecs.umich.edu/wiki/index.php/AprilTags .
33. Página oficial de ARLab. Fecha de última consulta: 25 de enero de 2015. Disponible en: http://www.arlab.com/ .
34. Repositorio del proyecto open source DroidAR alojado en GitHub por Simon Heinen. Fecha de última consulta: 5 de febrero de 2015. Actualizado: 25 de agosto de 2014. Disponible en: https://github.com/simon-heinen/droidar .
35. Página oficial de Aurasma. Fecha de última consulta: 23 de enero de 2015. Disponible en: http://www.aurasma.com/ .
36. Página oficial de Xloudia. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 28 de agosto de 2015. Disponible en: http://www.xloudia.com/start-today/ .

37. Página oficial de ARcrowd. Fecha de última consulta: 23 de enero de 2015. Disponible en: http://arcrowd.com/ .
38. Página oficial del plugin ADT (Android Development Tools) para el IDE Eclipse por Android Developers. Fecha de última consulta: 30 de agosto de 2015. Actualizada: agosto de 2015. Disponible en: http://developer.android.com/tools/sdk/eclipse-adt.html .
39. Página oficial del IDE Android Studio por Android Developers. Fecha última consulta: 30 de agosto de 2015. Actualizada: agosto de 2015. Disponible en: http://developer.android.com/sdk/index.html .
42. Prueba de 30 días para Xloudia SDK por LM3LABs. Fecha de última consulta: 13 de marzo de 2015. Disponible en: http://2.lm3labs.com/product/1806/ .
43. Comparación realizada por usuarios de los SDKs de Realidad Aumentada por SocialCompare. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 23 de agosto de 2015. Disponible en: http://socialcompare.com/en/comparison/augmented-reality-sdks .
45. Herramienta de desarrollo de prototipos, Pencil Project. Fecha de última consulta: 11 de abril de 2015. Actualizada: 21 de marzo de 2014. Disponible en: http://pencil.evolus.vn/ .
46. Herramienta que nos permite hacer esquemas con las tareas, Task Architect. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 1 de agosto de 2015. Disponible en: http://www.taskarchitect.com/ .
47. Herramienta de diseño de modelos en 3D por Blender. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 8 de julio de 2015. Disponible en: https://www.blender.org/ .
48. Herramienta de diseño de modelos en 3D, SketchUp de Google. Fecha de última consulta: 29 de mayo de 2015. Actualizada: 5 de febrero de 2015. Disponible en: http://www.sketchup.com/es .
49. Motor de desarrollo de videojuegos Unity 3D creado por Unity Technologies. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 24 de agosto de 2015. Disponible en: http://unity3d.com/es .
50. Archivo con todas las versiones de Unity 3D para descargar por Unity Technologies. Fecha de última consulta: 30 de agosto de 2015. Actualizado: 24 de agosto de 2015. Disponible en: https://unity3d.com/es/get-unity/download/archive .

51. Sección de descarga del SDK de Vuforia desarrollado por Qualcomm. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 31 de julio de 2015. Disponible en: https://developer.vuforia.com/downloads/sdk .
52. Página oficial de desarrolladores en Vuforia. Fecha de última consulta: 30 de agosto de 2015. Actualizada: 29 de julio de 2015. Disponible en: https://developer.vuforia.com/ .
54. Artículo de la compra de Metaio por la empresa Apple Inc en revista tecnológica Applesfera de Weblogs SL por Raúl Álvarez. Fecha de última consulta: 2 de junio de 2015. Actualizado: 28 de mayo de 2015. Disponible en: http://www.applesfera.com/apple-1/apple-quiere-entrar-de-lleno-en-la-realidad-virtual-y-adquiere-metaio .
55. Extensión de Unity 3D, iTween. Fecha de última consulta: 24 de junio de 2015. Disponible en: http://itween.pixelplacement.com/index.php .
56. Repositorio de código de la librería "FloatingActionButton" de "getbase" alojada en GitHub. Fecha de última consulta: 30 de julio de 2015. Actualizado: 17 de julio de 2015. Disponible en: https://github.com/futuresimple/android-floating-action-button .
57. Portal del desarrollador de Metaio. Fecha de última consulta: 29 de agosto de 2015. Disponible en: https://my.metaio.com/index.php?state=portal .
58. Referencia de la clase "AlertDialog" en la librería de Android por Android Developers. Fecha de última consulta: 29 de septiembre de 2015. Actualizada: agosto de 2015. Disponible en: http://developer.android.com/reference/android/app/AlertDialog.html .
62. Aplicación "Guía de Olivenza" para Android por Alejandro Franco Silva y alojada en Google Play. Fecha de última consulta: 29 de agosto de 2015. Actualizada: 6 de agosto de 2015. Disponible en: https://play.google.com/store/apps/details?id=afrasilv.es.olivenzainfo .
63. Página de descarga del SDK de Wikitude. Fecha de última consulta: 29 de agosto de 2015. Actualizada: 27 de agosto de 2015. Disponible en: http://www.wikitude.com/download .

Índice de ilustraciones

Ilustración 1 - Resumen del Diseño Centrado en el Usuario 23
Ilustración 2 - Prototipo 1 de aplicación 1 operación apilar (8) 31
Ilustración 3 - Prototipo 1 de aplicación 1 al pulsar en iniciar sesión 32
Ilustración 4 - Prototipo 1 de aplicación 1 al seleccionar activar animación oculta 32
Ilustración 5 - Evolución Botón Opciones. Antiguo (izq.) y Nuevo (drcha.)..... 32
Ilustración 6 - Prototipo 2 de aplicación 1 con botón atrás en la parte superior derecha 33
Ilustración 7 - Versión 3 de la propuesta 1. Menú sin iniciar sesión..... 34
Ilustración 8 - Versión 3 de la propuesta 1. Menú con sesión iniciada de profesor 34
Ilustración 9 - Versión 3 de la propuesta 1. Menú configuración de la cuenta 35
Ilustración 10 - Versión 4 de la propuesta 1. Menú configuración de la cuenta 35
Ilustración 11 - Versión 5 de la propuesta 1. Vista de la lista de asignaturas y el nuevo botón de configuración 36
Ilustración 12 - Versión 5 de la propuesta 1. Menú de configuración de un tema 37
Ilustración 13 - Versión 5 de la propuesta 1. Vista de la lista de selección de la asignatura a borrar 37
Ilustración 14 - Versión 5 de la propuesta 1. Vista del diálogo con la segunda confirmación al borrar. 38
Ilustración 15 - Tarea de la aplicación 1 referente a "Cambiar de asignatura" para el usuario "alumno" 38
Ilustración 16 - Tarea de la aplicación 1 referente a "Activar animación oculta de Lista Doblemente Enlazada" para el usuario secundario 39
Ilustración 17 - Ejemplo pantalla principal prototipo 2..... 51
Ilustración 18 - Captura del prototipo: "Crear nueva tarjeta"..... 51
Ilustración 19 - Evolución Botón Opciones. Antiguo (izq.) y Nuevo (drcha.)..... 52
Ilustración 20 - Prototipo 2 con botón atrás 52
Ilustración 21 - Versión 3 de la propuesta 2. Menú sin iniciar sesión..... 53
Ilustración 22 - Versión 3 de la propuesta 2. Menú con sesión iniciada 54
Ilustración 23 - Versión 3 de la propuesta 2. Menú de acciones de una tarjeta 54
Ilustración 24 - Tarea de la aplicación 2 referente a "Crear Nueva Tarjeta"..... 55
Ilustración 25 - Tarea de la aplicación 2 referente a "Eliminar Cuenta" 56
Ilustración 26 - Nuevo Proyecto Unity..... 59
Ilustración 27 - Inicio Unity y Vuforia..... 60
Ilustración 28 - Nueva Clave de Licencia Vuforia 60

Ilustración 29 - Key License en Unity 61
Ilustración 30 - Usar Target Manager de Vuforia 61
Ilustración 31 - Tipos de Target en Vuforia 62
Ilustración 32 - Cuboid en Vuforia 63
Ilustración 33 - Cylinder en Vuforia 63
Ilustración 34 - Single Image Target en Vuforia 64
Ilustración 35 - Importar Target Manager a Unity 64
Ilustración 36 - Configuración de Image Target y Data Set 65
Ilustración 37 - Configuración de ARCamera y Data Set 65
Ilustración 38 - Modelos añadidos en Unity con Vuforia 66
Ilustración 39 - Resultado primera aplicación con Vuforia en Unity 67
Ilustración 40 - Exportar proyecto a Android 68
Ilustración 41 - Modificando el Bundle Identifier* 69
Ilustración 42 - Exportar Android Project 69
Ilustración 43 - Vuforia en dispositivo Android real 1 70
Ilustración 44 - Vuforia en dispositivo Android real 2 70
Ilustración 45 - Vuforia en dispositivo Android real 3 71
Ilustración 46 - Configurando un segundo "target" en Vuforia 72
Ilustración 47 - Escena de la propuesta dos en Unity 3D y Vuforia con dos "targets" .. 72
Ilustración 48 - Rastreo de múltiples imágenes a la vez en Vuforia y Unity 3D 73
Ilustración 49 - Ejemplo Vuforia y Unity para añadir animaciones con "Animation"... 73
Ilustración 50 - Ejemplo animación en Unity propuesta de aplicación 2 74
Ilustración 51 - Prueba de animación de la propuesta 2 desde Unity en PC 75
Ilustración 52 - Prueba de animación de la propuesta 2 desde Unity en PC 75
Ilustración 53 - Ejemplo Metaio Creator 77
Ilustración 54 - Añadir animación 1 78
Ilustración 55 – Añadir animación 2 – Creación animación 78
Ilustración 56 – Añadir animación 3 – Configuración reproducción 79
Ilustración 57 - Demo de aplicación al reconocer el patrón 79
Ilustración 58 - Demo de aplicación al pulsar sobre el recurso AR 80
Ilustración 59 - Modelos Metaio Blender Vs SketchUp 1 80
Ilustración 60 - Modelos Metaio Blender Vs SketchUp 2 81
Ilustración 61 - Resumen desarrollo propuesta 1 en Unity con Vuforia 83
Ilustración 62 - Resumen desarrollo propuesta 1 en Metaio 84

Ilustración 63 - Modelos añadidos en Unity con Vuforia en propuesta 2 85
Ilustración 64 - Añadir nuevo Script a un objeto en Unity para la propuesta 2 86
Ilustración 65 - Script en la vista de "Inspector" en Unity 86
Ilustración 66 - Ventana "Animation" en Unity 88
Ilustración 67 - Ventana "Animation" en Unity con varios tiempos añadidos..... 88
Ilustración 68 - Propuesta 2 en Metaio Creator 91
Ilustración 69 - Acciones al pulsar un recurso AR en Metaio Creator..... 92
Ilustración 70 - Resumen desarrollo propuesta 2 en Unity 3D y Vuforia 93
Ilustración 71 - Resumen desarrollo propuesta 2 en Metaio 93
Ilustración 72 - Estructura de un proyecto exportado por Unity en Android Studio 94
Ilustración 73 - Esquema de la estructura de un proyecto exportado por Unity en Android Studio 95
Ilustración 74 - Ejemplo de "MaterialDialog" en la aplicación Android 99
Ilustración 75 - Estructura de un proyecto exportado por Metaio en Android Studio . 101
Ilustración 76 - Esquema de la estructura de un proyecto exportado por Metaio en Android Studio 102
Ilustración 77 - Pantalla inicial de la aplicación exportada por Metaio con el botón de opciones..... 104
Ilustración 78 - Menú de opciones de la aplicación exportada por Metaio..... 104
Ilustración 79 - Resumen de los desarrollos en Android Studio 107
Ilustración 80 - Evolución de la aplicación entre 3 versiones diferentes de la misma información..... 114
Ilustración 81 - Ciudadela y primera muralla de Olivenza con un mosaico de piedra asociado 115
Ilustración 82 - Puerta de San Francisco en Olivenza derribada a principios del S. XX 116
Ilustración 83 - Pantalla inicial de Blender 2.7.4 129
Ilustración 84 - Menú de Texturas en Blender 129
Ilustración 85 - Añadiendo imágenes al modelo en Blender..... 130
Ilustración 86 - Textura en cara del cubo en Blender..... 130
Ilustración 87 - Modelo de cubo en Blender 131
Ilustración 88- Ventana de inicio de Unity 3D..... 132
Ilustración 89 - Objetos que incluye Unity por defecto..... 132
Ilustración 90 - Captura de Unity con un objeto cubo con textura..... 133

Ilustración 91 - Ventana de inicio de SketchUp 134
Ilustración 92 - Dibujando un rectángulo en SketchUp 134
Ilustración 93 - Creando el cubo desde el rectángulo 135
Ilustración 94 - Cubo en SketchUp 135
Ilustración 95 - Añadiendo texturas al cubo en SketchUp 136
Ilustración 96 - Modelo de cubo completo en SketchUp 136
Ilustración 97 - Nueva imagen en GIMP 137
Ilustración 98 - Creación de capa en GIMP 138
Ilustración 99 - Creación de capa transparente en GIMP 138
Ilustración 100 - Primera imagen de la animación 139
Ilustración 101 - Capas y tiempo de cada capa en GIMP 139
Ilustración 102 - Exportar imagen como GIF en GIMP 140

Anexo 1 – Modelos y Animaciones

Creación de modelos con Blender

Blender es un software de diseño de modelos muy potente, gratuito y multiplataforma, por todo ello lo vamos a utilizar para crear nuestros modelos. Utilizaremos la versión 2.7.4 de la herramienta.

Por defecto cuando abramos la herramienta nos encontraremos con un diálogo de ayuda y un cubo sobre el mapa 3D en el que dibujaremos.

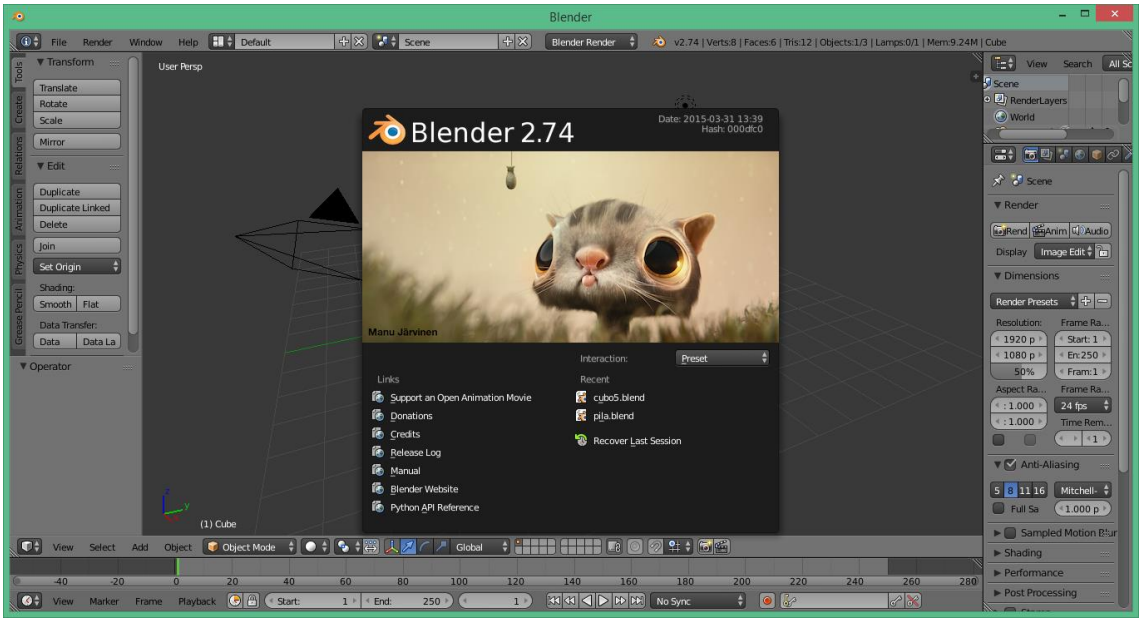

Ilustración 83 - Pantalla inicial de Blender 2.7.4

Como ya hay un cubo sobre el mapa, trabajaremos con él. Para añadirle las texturas o imágenes, lo seleccionaremos pulsando sobre él y en el menú de la derecha, nos iremos a la pestaña de "Texture".

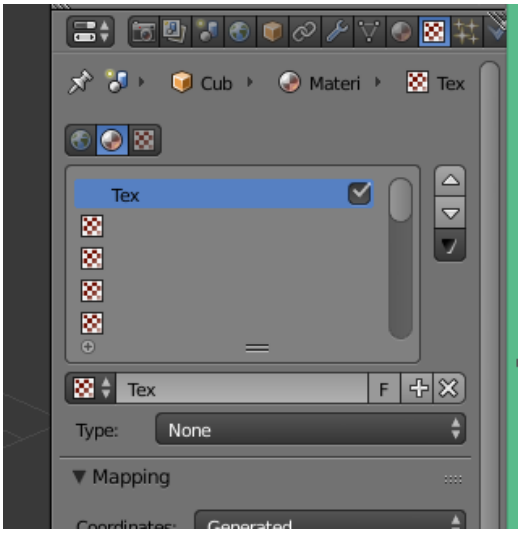

Ilustración 84 - Menú de Texturas en Blender

Una vez en esa vista, en el apartado de “Type” que podemos ver en la imagen superior, seleccionaremos “Image or Movie” y debajo en el apartado de “Image” seleccionamos la imagen que queremos ponerle a esa cara.

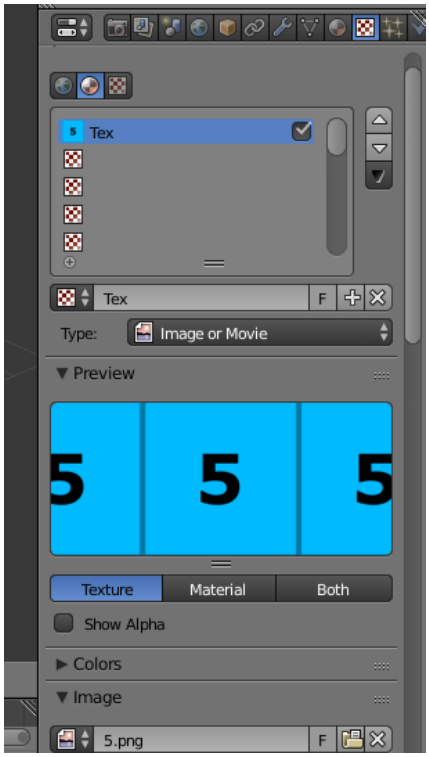

Ilustración 85 - Añadiendo imágenes al modelo en Blender

Para poder ver cómo queda la imagen que acabamos de añadir, nos vamos al menú “Render” en la parte superior y seleccionamos “Render Image” o directamente pulsamos “F12” y para salir pulsamos la tecla de escape o “ESC”.

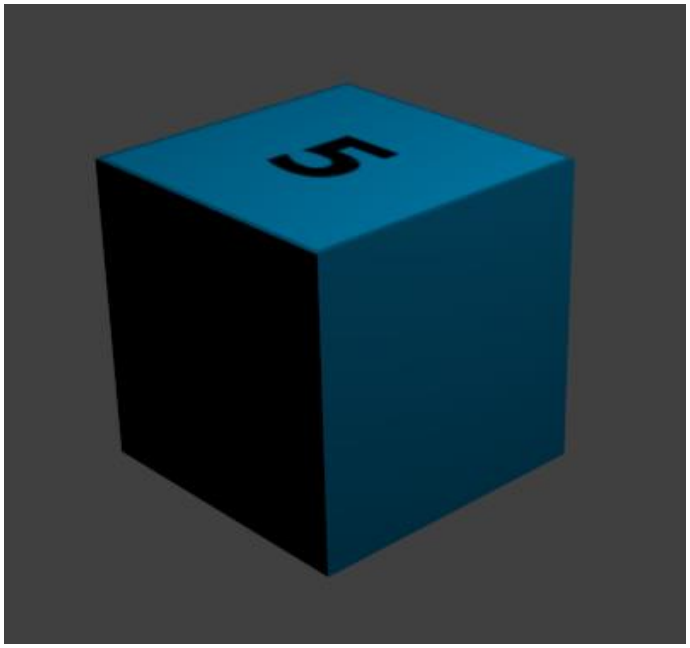

Ilustración 86 - Textura en cara del cubo en Blender

Deberemos hacer el mismo proceso para las caras restantes del cubo hasta que lo terminemos.

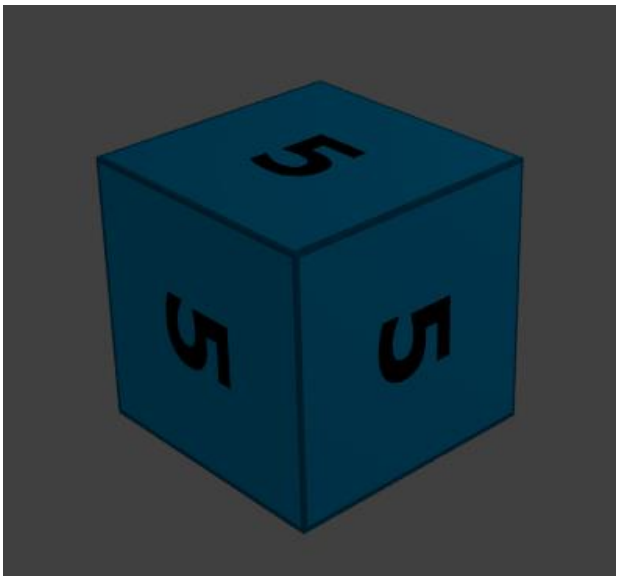

Ilustración 87 - Modelo de cubo en Blender

Para exportar ahora nuestro modelo, pulsaremos sobre el menú "File" y la opción "Export" y seleccionaremos la que queramos, en nuestro caso seleccionaremos las opciones de "Collada .dae", "FBX .fbx" y "Wavefront .obj".

Creación de modelos con Unity 3D

El proceso de creación de modelos simples con Unity 3D es muy sencillo. Nada más abrir la herramienta con un proyecto vacío nos encontraremos una interfaz similar a la siguiente[64]:

[64] La ventana puede variar en función de la versión de Unity 3D. Nosotros trabajaremos con la versión 4.6

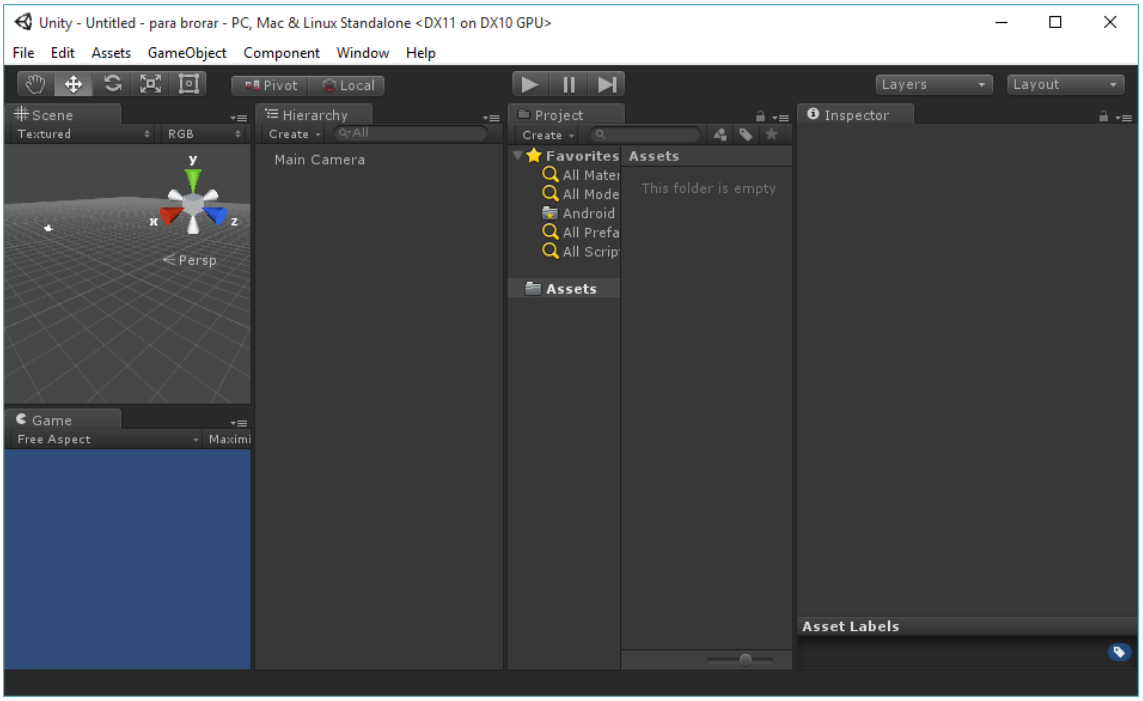

Ilustración 88- Ventana de inicio de Unity 3D

En la parte superior de la imagen y de la ventana de Unity, podemos ver una opción de la barra de tareas que dice "GameObject" en la que podemos encontrar una gran variedad de los objetos más simples que nos podemos encontrar y que para nosotros no nos harán falta más. Algunos de estos objetos de los que hablamos son los siguientes:

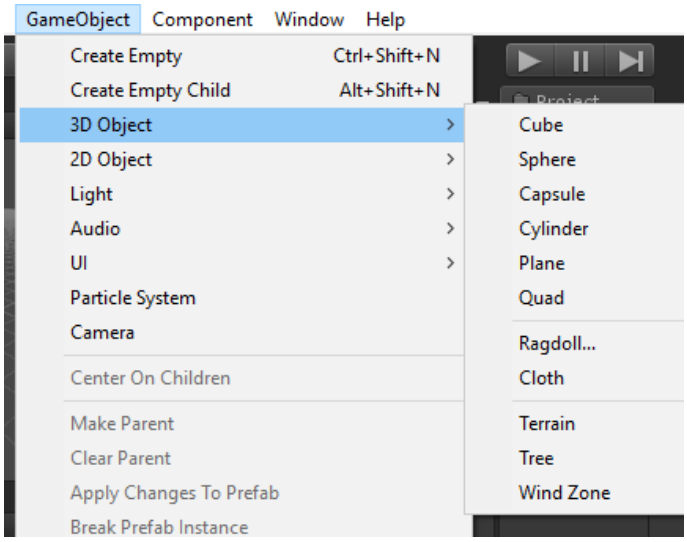

Ilustración 89 - Objetos que incluye Unity por defecto

Para nuestro ejemplo, nosotros añadiremos un cubo. En la parte derecha, en la vista de "Project", veremos una carpeta que pone "Assets" pues crearemos una carpeta que llamaremos "texturas" para almacenar las imágenes que queremos asociar con cada uno de los objetos. También buscaremos una imagen que queramos y la arrastramos hasta esa carpeta.

Una vez que tenemos la imagen dentro de la carpeta “Textures”, tan sólo tendremos que seleccionarla y arrastrarla sobre el cubo que hemos creado antes, que podremos ver ahora dentro de la vista de “Hierarchy” o jerarquía. Cuando lo hayamos completado veremos algo similar a la siguiente captura de Unity, en la que podemos ver el objeto del cubo que creamos antes en 3D al a izquierda y la textura dentro de “Asses/Texturas”:

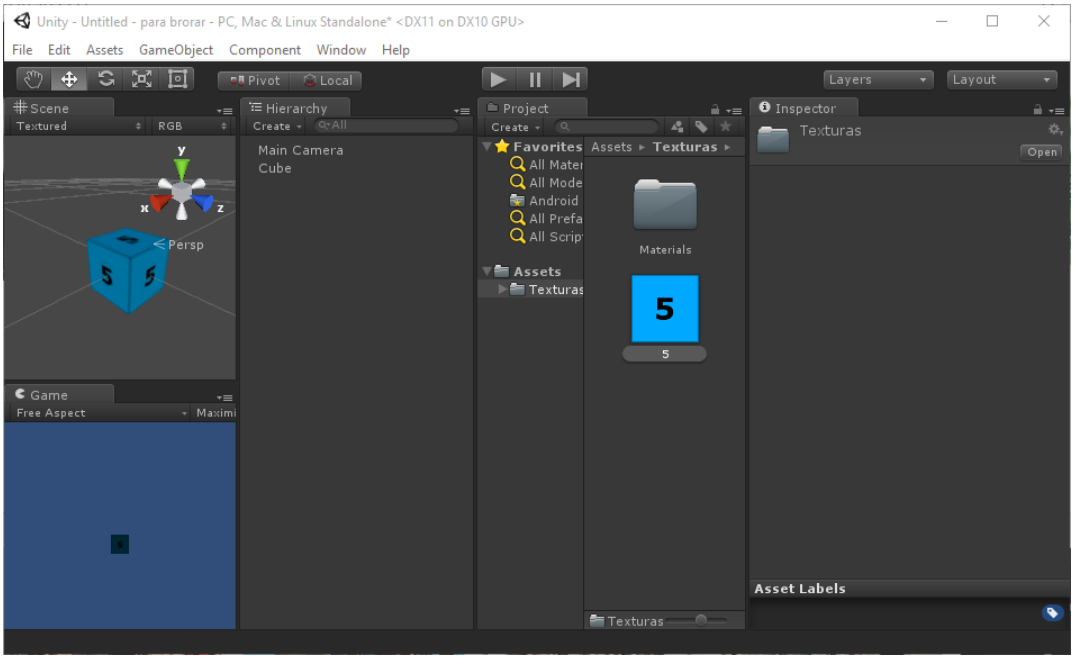

Ilustración 90 - Captura de Unity con un objeto cubo con textura

Una vez tenemos esto hecho, ya sabríamos como añadir un objeto y podríamos añadir todos los objetos que queramos.

Anteriormente vimos también como se añadía una animación a cada objeto, en este enlace podremos encontrarlo Animaciones en Unity; y así poder animar todos los objetos que queramos.

Creación de modelos con SketchUp

Crear los modelos con SketchUp es un proceso muy simple. Durante el proceso usaremos la versión 15.3.331 de 64-bits del programa. Cuando abramos la herramienta nos encontraremos con una ventana similar a la que podemos ver a continuación:

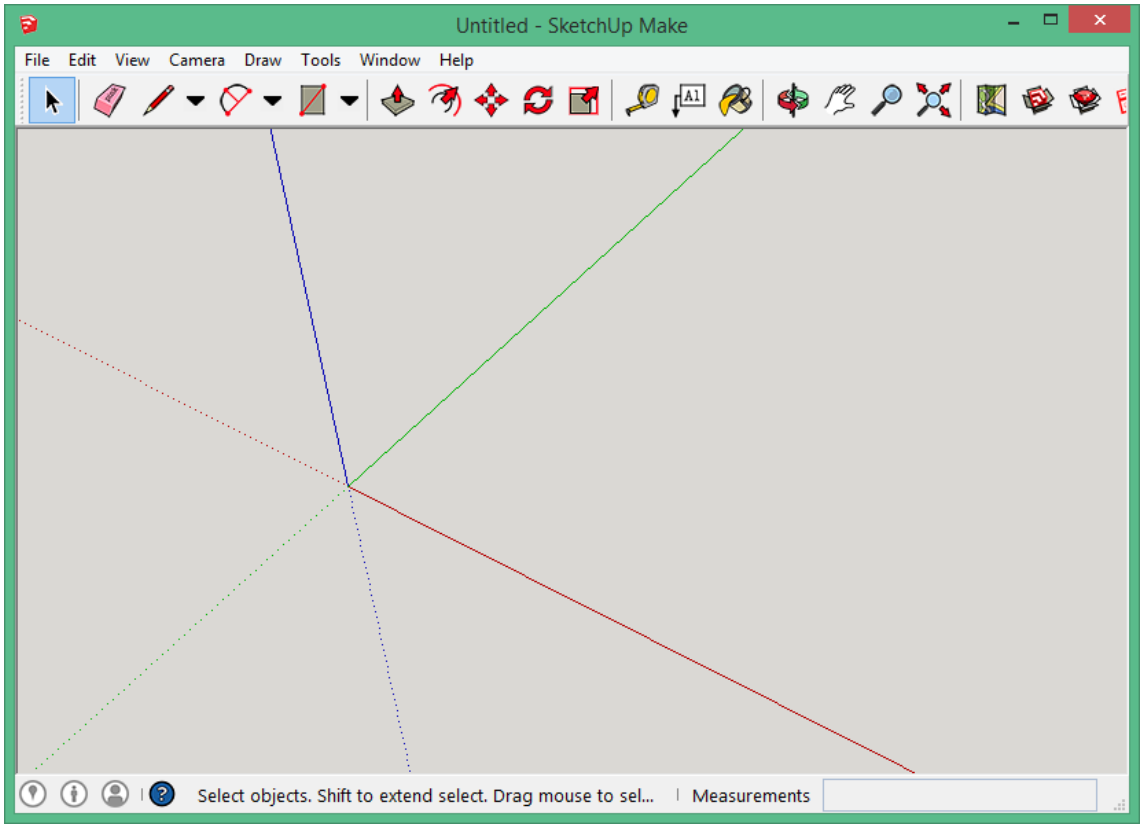

Ilustración 91 - Ventana de inicio de SketchUp

Para crear los modelos básicos, como cuadrados o círculos; nos iremos al menú "Draw" que podemos ver en la parte superior, una vez dentro ya podremos elegir lo que queremos dibujar sobre los ejes de coordenadas. Como en nuestro modelo queremos dibujar cubos ahora mismo, lo que haremos será dibujar un cuadrado seleccionándolo en el menú que acabamos de comentar

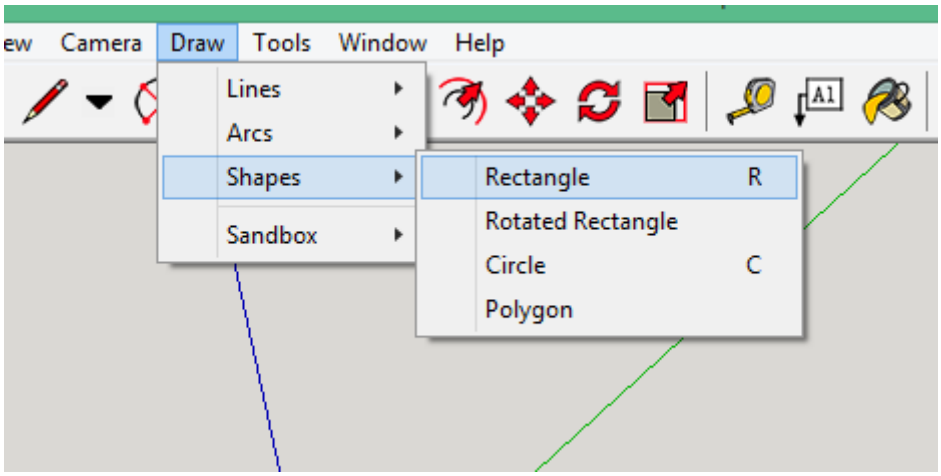

Ilustración 92 - Dibujando un rectángulo en SketchUp

Al seleccionarlo, dibujaremos un rectángulo con las medidas que nosotros queramos y en la posición que nosotros queramos. Una vez lo tengamos, pulsaremos en el botón de

“Push/Pull” y a continuación sobre el rectángulo y haremos que crezca en el eje Y para convertirlo en cubo.

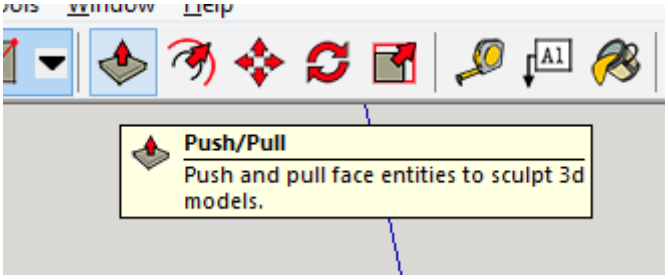

Ilustración 93 - Creando el cubo desde el rectángulo

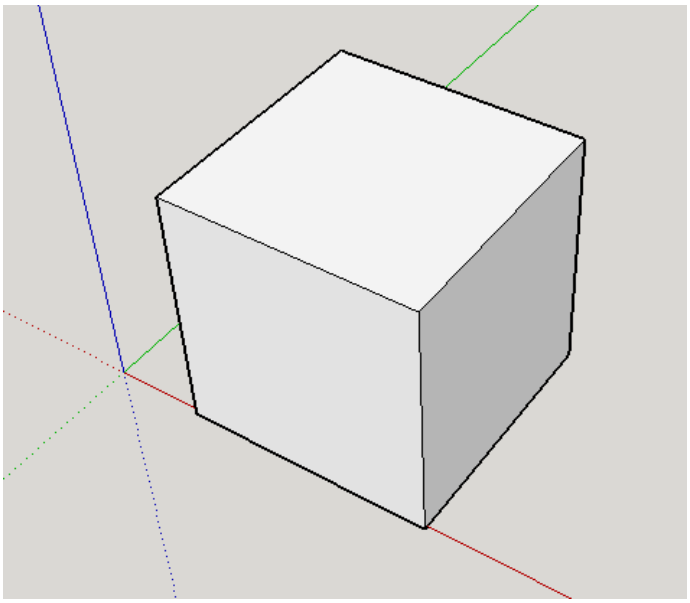

Ilustración 94 - Cubo en SketchUp

Una vez tengamos el cubo, le añadiremos las imágenes o texturas que queramos en cada cara, tendremos que ir una a una. Si una de las caras es mayor o menor que la imagen, en el caso de que queramos un cubo cuadrado; con la herramienta de “Push/Pull” podremos hacer crecer o decrecer el cubo en el eje que queramos para adaptar su tamaño.

Para añadir esas imágenes, nos vamos al menú “File” y a “Import”. En la ventana que se nos abra, buscaremos la imagen que queramos y la ponemos sobre la cara que queramos.

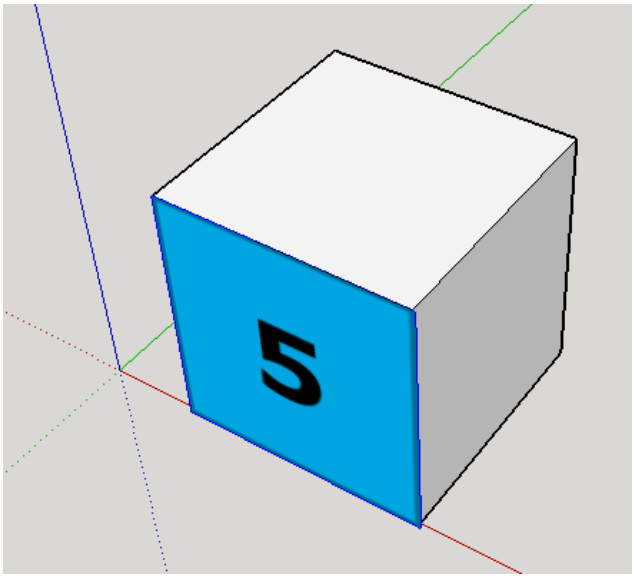

Ilustración 95 - Añadiendo texturas al cubo en SketchUp

Comprobamos si el tamaño de todos los ejes es el correcto porque nuestra imagen es cuadrada y volvemos a importar la imagen para las demás caras.

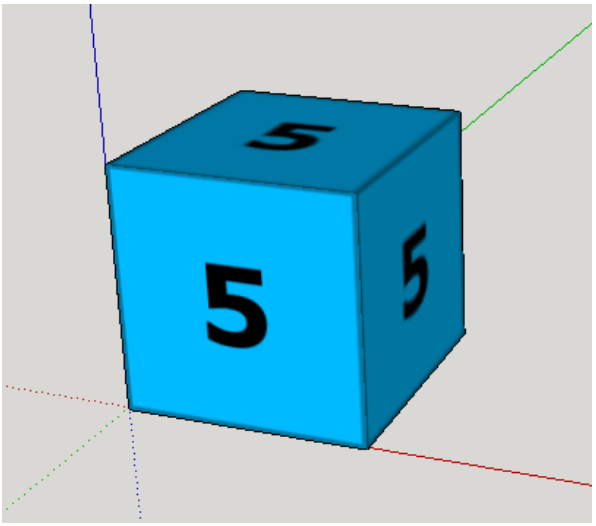

Ilustración 96 - Modelo de cubo completo en SketchUp

Cuando lo tengamos terminado, podemos exportar el modelo en los formatos más comunes para utilizarlos en los otros programas de creación de las aplicaciones AR como Unity3D o Metaio Creator, por ejemplo. Para ello, haremos el mismo proceso que para importar, pero ahora exportando; pulsamos en el menú “File” y a continuación en “Export” y seleccionamos un objeto 3D y el formato con el que queremos exportarlo en la siguiente ventana. Nosotros usaremos los formatos “OBJ .obj”, “Collada .dae” y “FBX .fbx”.

Creación de animaciones (2D) con GIMP

Con GIMP podremos crear animaciones en formato ".gif" de una forma bastante simple para ver si podemos utilizarla con nuestros prototipos o aplicaciones. Hemos seleccionado GIMP y no una herramienta similar, porque GIMP es gratuito, de software libre y está disponible en todas las plataformas sin dificultades para instalarlo o acceder a él.

La versión que utilizaremos de la herramienta es la versión 2.8. Una vez la tengamos abierta, crearemos una imagen nueva pulsando sobre "Archivo" y "Nuevo". Seleccionaremos una plantilla de las que hay para no complicarnos con el tamaño y aceptaremos. De ese modo ya tendremos la base de nuestra imagen creada.

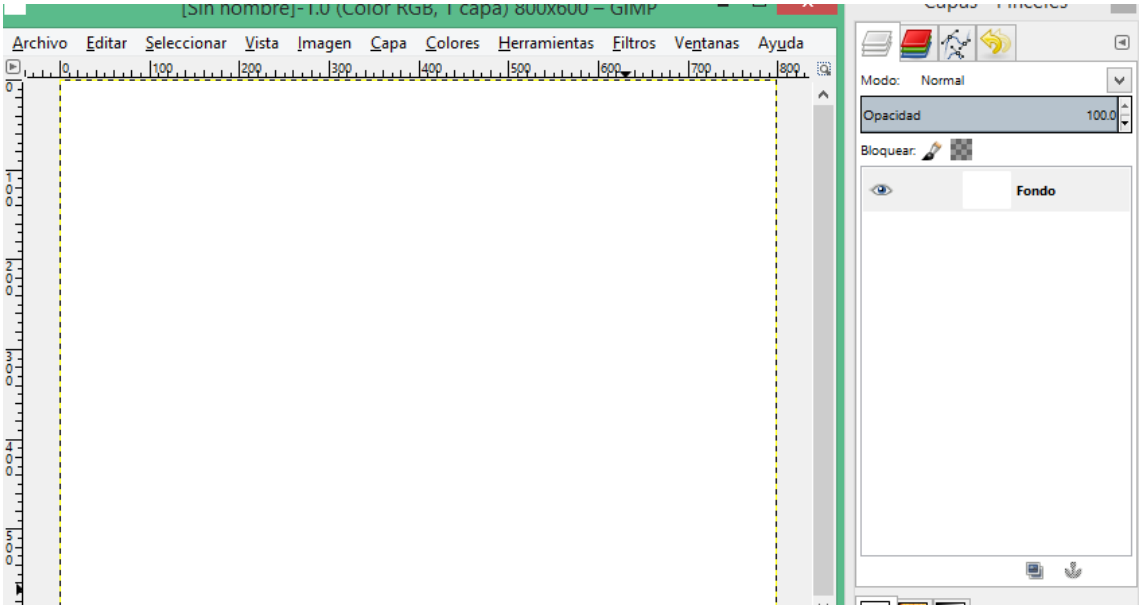

Ilustración 97 - Nueva imagen en GIMP

Como queremos que debajo de nuestra animación se vea la imagen que haya en la cámara, eliminaremos en la parte derecha la capa "Fondo" y crearemos una nueva seleccionando la opción de "Transparencia". Para crear la capa nueva, pulsaremos con el botón derecho sobre la ventana de "Capas", la misma en la que está ahora mismo el "Fondo".

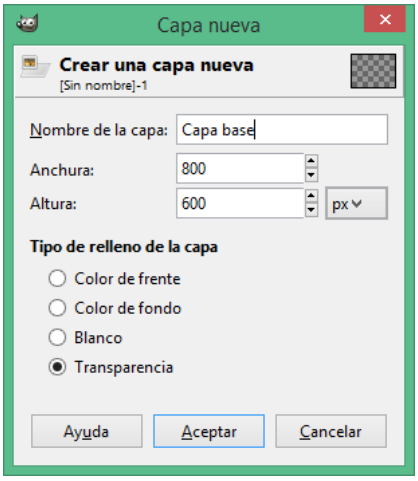

Ilustración 98 - Creación de capa en GIMP

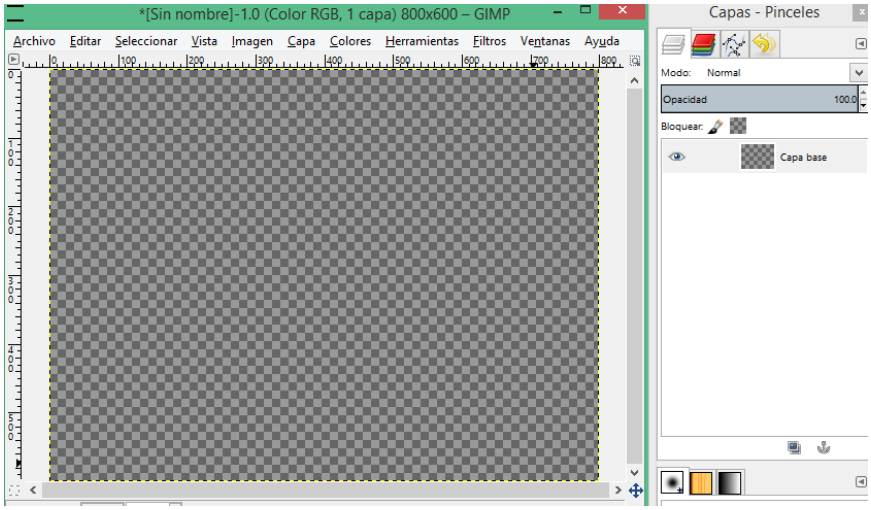

Ilustración 99 - Creación de capa transparente en GIMP

Una vez tenemos esta capa transparente, podemos pintar sobre ella lo que queramos que el fondo desaparecerá cuando queramos usarlo siempre que sea compatible con el formato.

Como una de nuestras propuestas de aplicaciones consiste en una animación sobre apuntes, la primera que crearemos será con los métodos de la pila, en la misma diapositiva que hemos usado para los prototipos de la segunda aplicación con la herramienta Pencil.

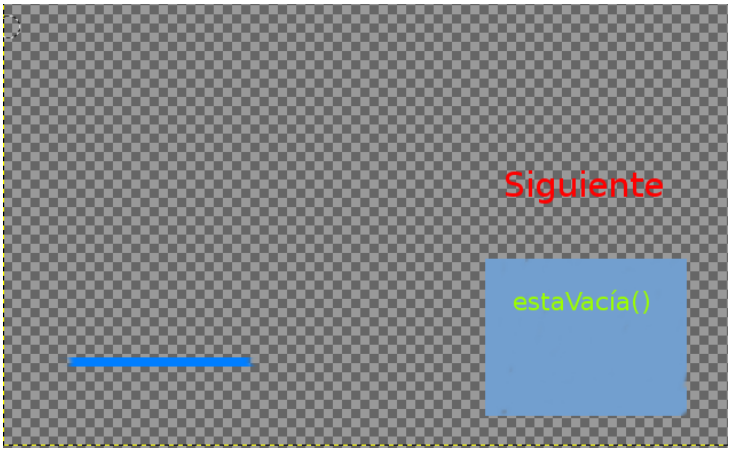

Ilustración 100 - Primera imagen de la animación

En la imagen superior, podemos ver un cuadro sobre el que saldrán los métodos y los objetos que van a irse "ejecutando" y en la parte izquierda una barra que correspondería a nuestra pila.

Una vez tenemos esa imagen creada, en la ventana de la derecha pulsaremos con el botón derecho y seleccionaremos "Duplicar la capa". Y en cada capa iremos nosotros creando la animación como queremos que se viera.

Para indicarle al programa los segundos o milisegundos que queremos que la imagen correspondiente a esa capa esté en la pantalla, se lo añadiremos al título de la capa con el siguiente formato:

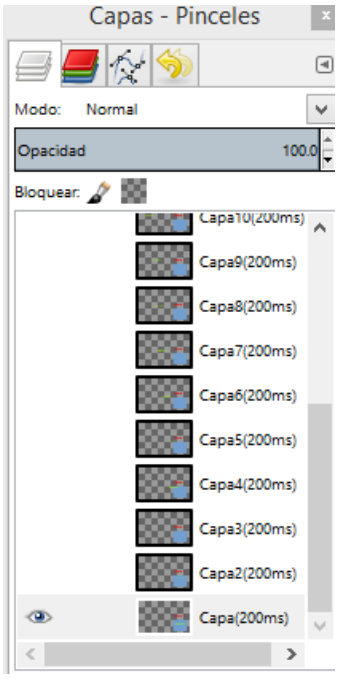

Ilustración 101 - Capas y tiempo de cada capa en GIMP

Como podemos ver, el formato sería poniendo entre paréntesis el tiempo que deseemos a continuación del título y sin espacios entre ellos.

Una vez tengamos creada nuestra animación con todas las capas, generaremos el archivo “.gif” en el que la almacenaremos. Para ellos iremos a “Archivo”, “Exportar como…”, le pondremos el nombre que queramos y el formato “gif”. Nos saldrá una nueva ventana con varias opciones para exportar la imagen como GIF.

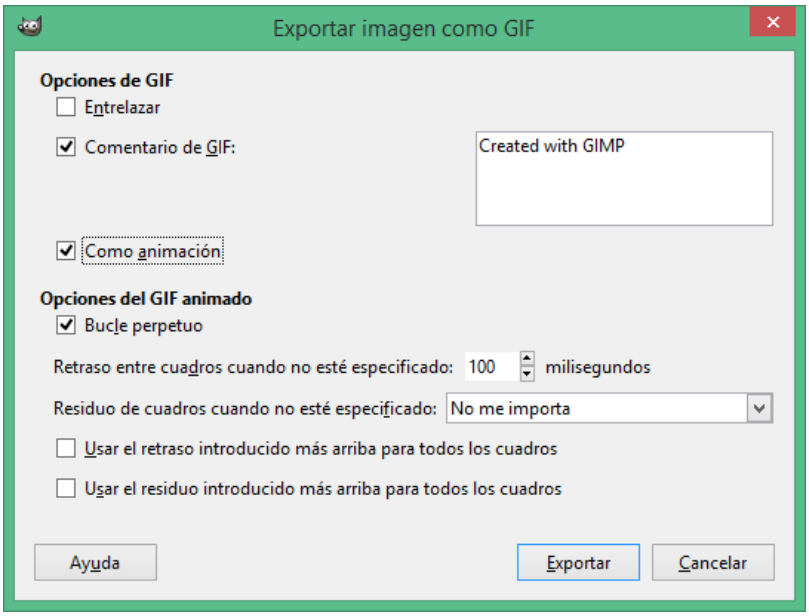

Ilustración 102 - Exportar imagen como GIF en GIMP

Le indicaremos que es una animación y que es también un “Bucle perpetuo” para que se repita una y otra vez. También nos da la opción de que le pongamos un tiempo estándar en milisegundos para una capa que no tenga puesto el tiempo, por si en alguna nos hemos equivocado u olvidado. Una vez tengamos todo configurado, pulsaremos sobre “Exportar” y ya tendremos nuestra animación con los modelos tal y como la hayamos creado lista para añadírsela al programa de Realidad Aumentada que sea compatible.