
# UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

*La Universidad Católica de Loja*

# ÁREA TÉCNICA

TÍTULO DE INGENIERO EN SISTEMAS INFORMATICOS Y COMPUTACIÓN

**Realidad Aumentada en la asignatura Desarrollo de la Inteligencia de la Modalidad Abierta y a Distancia**

TRABAJO DE TITULACIÓN

**AUTOR:** Freire Hidalgo, Michael Fernando

**DIRECTORA:** Jara Roa, Dunia Inés, Ing.

LOJA – ECUADOR

2015



*Septiembre, 2015*

## APROBACIÓN DE LA DIRECTORA DEL TRABAJO DE TITULACIÓN

Ingeniera.

Dunia Inés Jara Roa

**DOCENTE DE LA TITULACIÓN**

De mi consideración:

El presente trabajo de fin de titulación: Realidad Aumentada en la asignatura Desarrollo de la Inteligencia de la Modalidad Abierta y a Distancia realizado por Michael Fernando Freire Hidalgo, ha sido orientado y revisado durante su ejecución, por cuanto se aprueba la presentación del mismo

Loja, Octubre 2015

f). . . . . .

## DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

"Yo Michael Fernando Freire Hidalgo declaro ser autor del presente trabajo de titulación: Realidad Aumentada en la asignatura Desarrollo de la Inteligencia de la Modalidad Abierta y a Distancia, de la Titulación de Ingeniero en Sistemas Informáticos y Computación, siendo Dunia Inés Jara Roa directora del presente trabajo; y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales. Además certifico que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de mi exclusiva responsabilidad.

Adicionalmente declaro conocer y aceptar la disposición del Art. 88 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado o trabajos de titulación que se realicen con el apoyo financiero, académico o institucional (operativo) de la Universidad"

f.....
Autor: Michael Fernando Freire Hidalgo.
Cédula: 0704909449

## DEDICATORIA

Este presente trabajo se lo dedico principalmente a Dios, por permitir que este sueño sea una realidad.

A mis padres Victor Freire y Zoila Hidalgo, quienes siempre han estado presentes a lo largo de mi vida apoyándome y motivándome a cumplir mis sueños.

Y a todas aquellas personas que de una u otra manera siempre creyeron en mí, y en que podría lograr culminar mi carrera.

## AGRADECIMIENTO

Especialmente agradezco a mis padres, por todo el apoyo brindado durante cada uno de los obstáculos presentados a lo largo de todos estos años, por ser mi motivación y esa fuerza interior que me impulsa a ser mejor cada día.

A todas mis amistades que forme durante esta grata etapa de mi vida, con quienes compartimos el sueño de ser profesionales.

A la Ing. Dunia Inés Jara; Directora de la presente tesis, por haberme guiado exigido y motivado durante el desarrollo del presente trabajo de tesis.

## ÍNDICE DE CONTENIDOS

## ÍNDICE DE FIGURAS

### FIGURAS CAPITULO 1



### FIGURAS CAPITULO 2



### FIGURAS CAPITULO 3

## ÍNDICE DE TABLAS

**TABLAS CAPITULO 1**



**TABLAS CAPITULO 3**

## RESUMEN

En la actualidad, donde las técnicas convencionales para impartir los conocimientos a los estudiantes de educación a distancia, a través de libros y guías de estudio con contenidos estáticos no están logrando el objetivo de motivar el aprendizaje de los estudiantes. Es en este punto, donde las nuevas tecnologías enfocadas a la educación, ayudan a fomentar el interés y principalmente motivar el aprendizaje a través de herramientas tecnológicas mediante contenidos interactivos, donde el estudiante pasa de ser un mero espectador, a ser el actor principal de su propio aprendizaje.

En el presente trabajo se ha desarrollado una aplicación móvil de realidad aumentada para la asignatura de Desarrollo de la Inteligencia, a través de la cual los estudiantes pueden interactuar con contenidos u objetos 3D interactivos afines a dicha asignatura. Mediante la aplicación móvil el estudiante simplemente debe enfocar la tarjeta RA con su cámara para generar el escenario de realidad aumentada, pudiendo interactuar con el mismo a través de gestos táctiles u botones en pantalla dispuestos para este fin.



**PALABRAS CLAVES:** Realidad Aumentada, educación a distancia, aprendizaje, Android, Metaio, objetos 3D, aplicación móvil.

## ABSTRACT

Currently, where conventional techniques to impart knowledge to students of distance education through books and study guides with static content they are not achieving the desired motivate student learning goal. It is at this point, where new technologies focused on education, help stimulate interest and mainly motivate learning through technological tools through interactive content, where the student goes from being a mere spectator, to be the main actor of his own learning.

In this word we have developed a augmented reality mobile application for the subject of Development of Intelligence, through which students can interact with 3D interactive content or objects related to that subject. Using the mobile application the student must simply focus the RA card with your camera to generate the scenario augmented reality, can interact with them through touch gestures or onscreen buttons arranged for this purpose.



**KEYWORDS:** Augmented Reality, distance education, learning, Android, Metaio, 3D objects, mobile application.

# INTRODUCCIÓN

Entre las diversas dificultades que deben afrontar los estudiantes de educación a distancia, se puede resaltar a la falta de contenidos dinámicos o interactivos como un factor recurrente entre la mayoría de componentes académicos, los cuales únicamente se limitan a los contenidos estáticos impresos en los libros o guías de estudio. Ante estas dificultades surge la necesidad de buscar en las nuevas tecnologías herramientas de apoyo que permitan brindar a los estudiantes alternativas interactivas de estudio, como es el caso de la utilización de la Realidad Aumentada como herramienta de apoyo, en la que el estudiante pasa de ser un mero espectador de contenidos estáticos, a ser el actor principal de su propio aprendizaje mediante la inclusión de modelos 3D interactivos de Realidad Aumentada, con los cuales el estudiante puede interactuar libremente y de esta forma fomentar e incentivar su aprendizaje.

Por estas razones se cree conveniente el desarrollo de una aplicación móvil de realidad aumentada para el componente académico de Desarrollo de la Inteligencia, en la cual el estudiante puede interactuar con los contenidos más representativos de dicho componente académico, ya que al no existir comunicación directa con el docente, y solo contar con la guía de estudio impresa, resulta difícil para el estudiante asimilar todos los contenidos adecuadamente.

Este trabajo de fin de titulación está compuesto por tres capítulos, detallados a continuación:

En el Capítulo 1. Estado del Arte: Se abordan todos los conceptos básicos, componentes, herramientas y tecnologías empleadas para la generación de un escenario de realidad aumentada, así también las principales aplicaciones de realidad aumentada enfocadas a la enseñanza, además de una comparativa de los principales SDK de desarrollo de aplicaciones de realidad aumentada.

En el Capítulo 2: Metaio SDK: Se describe a fondo las principales características del SDK de Metaio, su arquitectura, los sistemas de seguimiento y los lenguajes de desarrollo soportados, así también se estudia el lenguaje AREL, su estructura y arquitectura para el desarrollo de aplicaciones de realidad aumentada multiplataforma.

En el Capítulo 3: Desarrollo: Se detalla el desarrollo de la aplicación de realidad aumentada en la cual se abordan temas como los dispositivos soportados, la recolección de contenidos y los lineamientos generales para la generación de objetos 3D, así mismo, se describe la metodología de desarrollo utilizada para la implementación de la aplicación de realidad aumentada. Y finalmente las conclusiones y recomendaciones que hemos llegado luego de realizar el trabajo de fin de titulación.

# CAPITULO 1
# ESTADO DEL ARTE

## 1.1. Introducción.

Al ver o escuchar “Realidad Aumentada” (RA) es fácil imaginar cómo describir su significado, quizá ligándolo a imágenes que adquieren movimiento, videos o imágenes 3D, o tal vez imaginarse un futuro totalmente virtual liderado por maquinas, robots y formas digitales. Para tener claro que es la (RA), se presenta el siguiente concepto: La RA es una tecnología que permite la superposición, en tiempo real, de imágenes generadas por un ordenador sobre imágenes del mundo real. En un entorno de este tipo el usuario puede, además, interactuar con objetos virtuales usando objetos reales de una forma transparente. (Fundación Telefonica, 2011). La RA tiene como objetivo incorporar información virtual al mundo real, de tal manera que el usuario pueda llegar a pensar que forma parte de su realidad cotidiana. Es preciso mencionar que la RA y la Realidad Virtual comparten una serie de características comunes como por ejemplo: la inclusión de información, modelos 3D y programación para crear simulaciones visuales dentro del campo de visión del usuario. (Basogain, Olabe, Espinoza, Roueche, & Olabe, 2007) Mencionan que la principal diferencia entre estas tecnologías según es que la RA no busca reemplazar el mundo real, sino enriquecerlo con información virtual adicional proyectada dentro del mundo real; mientras que la Realidad Virtual, busca sustituir el mundo real por un mundo completamente virtual. La RA permite al usuario interactuar con la información virtual sin perder el contacto con el mundo real.

## 1.2. Realidad Aumentada en la educación.

La RA en los últimos años toma fuerza en diferentes campo como la medicina, arquitectura, videojuegos, etc.; sin embargo, en el contexto de la educación hay mucho por hacer, estableciendo ideas innovadoras, ya que ofrece grandes posibilidades por su atractivo y capacidad de insertar objetos virtuales en un espacio real, para esto se agrupan tecnologías como Smartphone, Tablet, cámaras entre otras, las que permiten la superposición, en tiempo real, de imágenes, marcadores o información generadas virtualmente, sobre imágenes del mundo real. La RA también ha demostrado su función pedagógica como una potente herramienta para proporcionar a los estudiantes nuevos contenidos, y a profesores nuevas metodologías de enseñanza, ofreciendo la interacción con contenidos de aprendizaje.

## 1.3. Características.

Las principales características que se deben abordar al desarrollar aplicaciones de RA, según Azuma (consultado por (Olwal, 2010)) son:

- ✓ **Combinar lo virtual con lo real:** La Realidad Aumentada requiere de un dispositivo que permita al usuario ver información real y virtual en una vista combinada.
- ✓ **Registro en 3D:** La relación entre lo real y lo virtual es basada en su relación geométrica. Esto hace que sea posible mostrar el contenido virtual en la localización y perspectiva 3D correcta con respecto al mundo real
- ✓ **Interacción en tiempo real:** Los sistemas de Realidad Aumentada deben ser capaces de superponer la información virtual y permitir su interacción en tiempo real.

Como se observa, el principal objetivo de la RA es crear una perfecta inmersión por parte del usuario, donde objetos virtuales puedan confundirse con objetos reales, creando la ilusión que los dos mundos (virtual y real) coexisten perfectamente.

## 1.4. Componentes.

Silva, Oliveira, & Giraldi, (2003) Detallan de forma general que los componentes presentes en una aplicación de RA, son:

### 1.4.1. Generador de escenario.

Es software o dispositivo necesario para realizar la representación de la escena virtual en el mundo real (rendering). Este no es un problema mayor en las aplicaciones de realidad aumentada puesto que generalmente no se trabaja con demasiados objetos en escena y sus representaciones no necesitan ser extremadamente realistas. Además, se debe tomar en cuenta que la capacidad de procesamiento de los dispositivos de RA crece constantemente.

### 1.4.2. Sistema de seguimiento (Tracking).

Es uno de los principales problemas en los sistemas de RA, los objetos en el mundo real y virtual requieren que se encuentren correctamente alineados unos de otros, para poder conseguir una completa inmersión de RA, dando la impresión de que los 2 mundos coexisten

(virtual y real). En campos como la medicina, el sistema de seguimiento cobra vital importancia, ya que sus aplicaciones por lo general requieren una precisión milimétrica.

### 1.4.3. Visualización.

Las aplicaciones de realidad aumentada se desarrollan de acuerdo a los dispositivos de visualización que se pretenda utilizar, ya se traten dispositivos ópticos (gafas de realidad aumentada, Google Glass), o dispositivos basados en video como Smartphone o computadoras dotados de cámara web, etc.

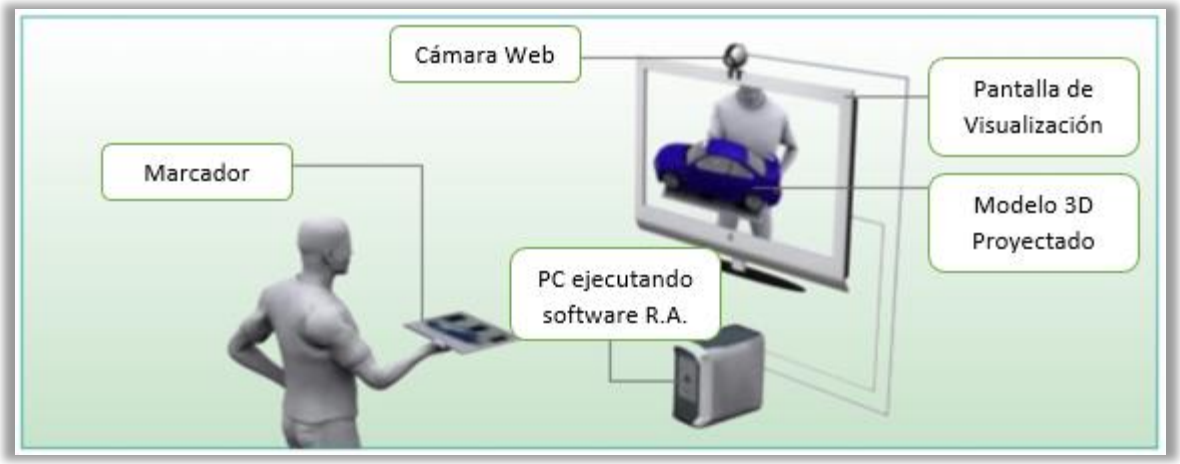



**Figura 1**. Escenario de Realidad Aumentada
Fuente: (Eventos Realidad Aumentada , 2012)

Como se puede observar en la Figura 1, un escenario de RA está compuesto por un dispositivo ejecutando el software de RA (generador de escenario), un marcador y cámara (sistema de seguimiento) y una pantalla de visualización de modelos 3D (visualización).

## 1.5. Objetos 3D.

Para la creación de objetos 3D con los cuales trabajar en aplicaciones de RA, se dispone de un gran número de herramientas de modelado en 3D, donde se puede destacar las siguientes:

- ✓ **Autodesk 3ds Max:** Proporciona una solución completa de modelado, animación, simulación. Con multitud de herramientas para la creación de contenido 3D y permite exportación a múltiples formatos utilizados por los SDK de desarrollo de aplicaciones de RA.

- ✓ **Blender:** Es un software especialmente orientado al modelado, iluminación, renderizado, animación y creación de gráficos tridimensionales. Blender, además permite el desarrollo de videojuegos, ya que cuenta con un motor de juegos integrado y es completamente gratuito.
- ✓ **SketchUp:** Software de modelado y diseño 3D basado en caras, propiedad de Google, su principal característica es la de poder realizar diseños complejos en 3D de forma extremadamente sencilla.

## 1.6. Tipos de tecnologías de visualización (Displays).

Entre las principales tecnologías utilizadas para la visualización de escenarios de RA se cita lo propuesto por Renukdas, Ghundiyal, Gadgil, & Pathare, (2013):

### 1.6.1. Lentes de Realidad Aumentada.

Las nuevas tecnologías en pantallas de RA permiten que estas puedan ser acopladas a unos lentes, este tipo de lentes emplean unas cámaras que registran la visión del mundo real y vuelven a mostrar al usuario dicha visión en combinación con elementos virtuales. Un ejemplo notable de este tipo de tecnología es el proyecto realizado por Google, llamado Google Glass.

### 1.6.2. Virtual Retina Display.

Es un dispositivo personal de visualización de RA desarrollado por el Laboratorio de Tecnologías de Interfaz Humana de la Universidad de Washinton. Con esta tecnología, una pantalla se escanea directamente en la retina del ojo de un espectador. El usuario ve lo que parece ser una pantalla convencional flotando en el espacio delante de ellos.

### 1.6.3. Dispositivos portátiles.

Los dispositivos portátiles como Smartphone, Tablet, etc. utilizan el sistema de video para mostrar escenarios de RA en su pantalla, inicialmente se empleaba marcadores fiduciarios para mostrar los objetos de RA. Los enfoques más actuales y futuros incluyen GPS, giroscopio, acelerómetro, etc., y sin marcadores a través de reconocimiento de escena. Las dos principales ventajas de los dispositivos portátiles de RA son el carácter portátil de éstos y la naturaleza generalizada de los mismos de contar con una cámara.

**Figura 2**. Dispositivos Portátiles
Fuente: (Lozano, 2013)

Como se puede observar se realiza una categorización de dispositivos basada principalmente en la tecnología usada para presentar los contenidos de RA al usuario, dentro de esta categorización los dispositivos portátiles son los más utilizados en el desarrollo de aplicaciones de RA por su gran cuota de mercado, debido principalmente a que dentro de esta categoría se engloban los Smartphone, que se encuentran ampliamente extendidos dentro de la población.

## 1.7. Aplicaciones de realidad aumentada en la educación.

Dentro del campo educativo se encuentran algunas aplicaciones interesantes como las detalladas a continuación:

### 1.7.1. LearnAR.

Es una nueva herramienta de aprendizaje que aporta características para la investigación e interacción en RA. Se trata de un paquete de diez planes de estudio para que maestros y estudiantes exploren mediante la combinación del mundo real con contenido virtual utilizando una cámara web. El paquete de recursos consiste en actividades de matemáticas, ciencias, anatomía, física, geometría, educación física e idiomas (LeanAR, 2014).

LearnAR fue el software pionero en implementar la RA en el campo de la educación, se expandió rápidamente a través del internet a múltiples escuelas y colegios de todo el mundo donde generó un gran interés por parte de profesores y alumnos, debido a que permitía afianzar y mejorar el aprendizaje realizado en las aulas mediante la inclusión de modelos en 3D de RA con los cuales los estudiantes podían interactuar. Actualmente es utilizada por centenares de establecimientos educativos como complemento a la enseñanza.

### 1.7.2. Google Sky Map.

Ofrece al usuario un mapa completo del cielo alrededor de la tierra en el que vienen indicadas constelaciones, estrellas, planetas e incluso meteoritos. Es una aplicación ideal para apoyar el estudio de astronomía, principalmente para quienes suelen interesarse en observar el espacio por las noches. Solamente basta con enfocar la cámara del móvil al cielo, y la aplicación mediante el uso del GPS identifica lo que se observa a través del mismo (Google, Google Sky Map web site, 2014).

Google Sky Map es una de las aplicaciones preferidas por los amantes de la astronomía, la que permitió a usuarios y estudiantes conocer nuestro espacio de una forma rápida y sencilla a través de su aplicación móvil de RA. Actualmente el sistema de Google Sky Map, se encuentra incorporado en Google Earth y la empresa Google libero el código de la aplicación de RA a la comunidad de desarrolladores.

### 1.7.3. Word Lens.

Es un traductor de idiomas que funciona simplemente enfocando con la cámara el texto que se desea traducir y la aplicación lo traduce en tiempo real al idioma seleccionado. El proceso es muy sencillo: el software identifica las letras que aparecen en el objeto y busca la palabra en el diccionario. Una vez que encuentra la traducción, la dibuja en lugar de la palabra original. La aplicación es ideal para aquellas personas que están aprendiendo otros idiomas, o aquellas que viajan mucho (Quest Visual, 2014).

World Lens se ha convertido por méritos propios en una herramienta indispensable para turistas y estudiantes de idiomas en todo el mundo, su impacto ha sido tal que Google adquirió la empresa desarrolladora de la aplicación Quest Visual para en un futuro utilizar la tecnología implementada en World Lens en su propio traductor Google Traslator. Actualmente World

Lens se encuentra disponible en Google Play Store de forma totalmente gratuita y cuenta con los siguientes paquetes de idiomas:

- ✓ Inglés ⇆ ruso
- ✓ Inglés ⇆ italiano
- ✓ Inglés ⇆ francés
- ✓ Inglés ⇆ español
- ✓ Inglés ⇄ alemán
- ✓ Inglés ⇄ portugués

### 1.7.4. Hallux Angels.

Es una aplicación que ayuda en la medición y el proceso de planificación radiográfica preoperatoria, su objetivo es complementar, no reemplazar, las técnicas convencionales utilizadas por el médico en este tipo de tratamiento. Mediante la utilización de la cámara y la radiografía, permite visualizar mediante RA el posible tratamiento a seguir (Ockendon, 2014).

La aplicación de RA Hallux Angels tiene gran acogida por parte de los médicos en Norteamérica, principalmente como herramienta de apoyo a las tareas de planificación preoperatoria de los pacientes. Actualmente Hallux Angels, se encuentra disponible gratuitamente para los dispositivos móviles con Sistema Operativo IOS.

## 1.8. Marcadores.

Uno de los requerimientos para las aplicaciones de RA es la utilización de marcadores o tarjetas RA, en las cuales se encuentra impresa una figura u imagen reconocible por la aplicación de RA, la cual sirve de guía o punto de referencia para indicarle a la aplicación donde generar los contenidos virtuales de RA.

**Figura 3**. Aplicación de Marcador (Tracking)
Fuente: (Loup, 2012)

### 1.8.1. Funcionamiento.

Para el reconocimiento del marcador la aplicación de RA utiliza un sistema de reconocimiento de imagen el cual mediante cálculos matemáticos es capaz de reconocer si la geometría de la imagen impresa en el marcador es la misma que ha sido cargada previamente en la aplicación, para esto la aplicación escanea constantemente el entorno comparando la imagen del marcador previamente precargada en la aplicación con las imágenes visualizadas del entorno, y cuando encuentra imagen con un porcentaje de aceptación dentro de los umbrales establecidos, es decir encuentra un marcador que corresponda con el de la aplicación genera los contenidos virtuales de RA sobre dicho marcador.

### 1.8.2. Tecnologías de reconocimiento de marcadores.

Antiguamente y debido a las bajas prestaciones de los dispositivos utilizados para la generación de entornos de RA se utilizaban patrones geométricos fácilmente reconocibles como marcadores, en la actualidad y con el desarrollo tecnológico de dichos dispositivos se puede utilizar casi cualquier imagen como un marcador, ya se trate de patrones geométricos, códigos QR, códigos de barras, o una fotografía. Adicionalmente los enfoques más modernos son capaces de utilizar el reconocimiento 3D para reconocer como marcadores objetos tridimensionales del mundo real, la geolocalización el cual mediante el GPS es capaz de servir de guía para la generación de contenidos de RA o una combinación de estos.

Entre las principales tecnologías utilizadas para el reconocimiento de marcadores de RA se puede destacar las siguientes:

- ✓ **Reconocimiento de Imagen:** La tecnología de reconocimiento de imagen utiliza algoritmos capaces de reconocer los patrones geométricos y contraste de las imágenes visualizadas permitiendo de esta manera diferenciar con una gran taza de acierto una imagen de otra.
- ✓ **Reconocimiento 3D:** Se basa en el reconocimiento de objetos 3D del mundo real a través de su geometría tridimensional, extrayendo de dichos objetos 3D datos tales como el tamaño, volumen, vértices, y profundidad, mediante los cuales a través de algoritmos especializados es capaz de reconocer los objetos 3D visualizados.
- ✓ **Reconocimiento mediante Geolocalización:** Es el reconocimiento de lugares a través de las coordenadas extraídas del GPS, permitiendo reconocer sitios u objetos a través de su ubicación (longitud y latitud), estos enfoques a su vez también se ayudan del giroscopio y acelerómetro de los dispositivos móviles para mejorar su exactitud.

### 1.8.3. Tipos de Marcadores.

Entre los principales tipos de marcadores se puede destacar los siguientes:

#### 1.8.3.1. Marcadores Geométricos.

La RA basada en marcas fiduciarias o tarjetas AR es sin duda la más extendida y reconocida. Su principal ventaja es que el patrón de reconocimiento no requiere de excesivo procesado en el equipo y por tanto funciona en la mayoría de los dispositivos.

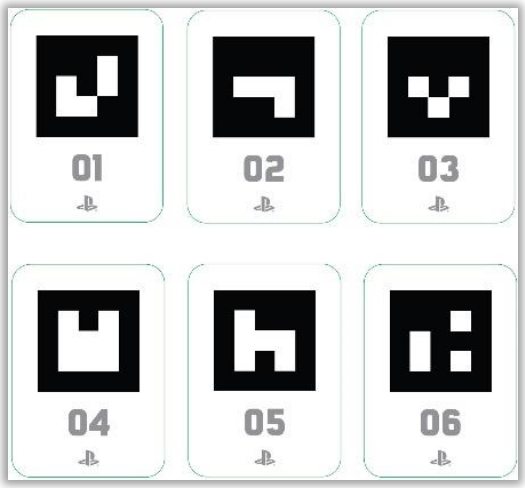



**Figura 4.** Marcadores Geométricos
Fuente: (Sony, 2012)

### 1.8.3.2. Marcadores de imagen.

Se caracterizan por la posibilidad de utilizar casi cualquier imagen como marcador y su principal ventaja es que no pierde el reconocimiento de la misma cuando la imagen es tapada parcialmente o en condiciones poco favorables


**Figura 5.** Marcadores de imagen
Fuente: (Nintendo, 2014)

#### 1.8.3.3. Marcadores 3D.

Permite el reconocimiento de objetos dentro del mundo real; es decir, se puede utilizar un objeto real con una geometría específica y utilizarla como si de un marcador de RA se tratase. De esta forma, es posible interactuar con objetos reales mediante el reconocimiento de la geometría 3D de los mismos.

**Figura 6.** Marcadores 3D
Fuente: (Metaio, Metaio Developer Portal, 2014)

#### 1.8.3.4. Reconocimiento Facial.

Es una tecnología que permite el reconocimiento de personas a través de su rostro mediante técnicas de reconocimiento 3D, permitiendo de esta manera utilizar el rostro de una persona para la generación de contenidos de RA.

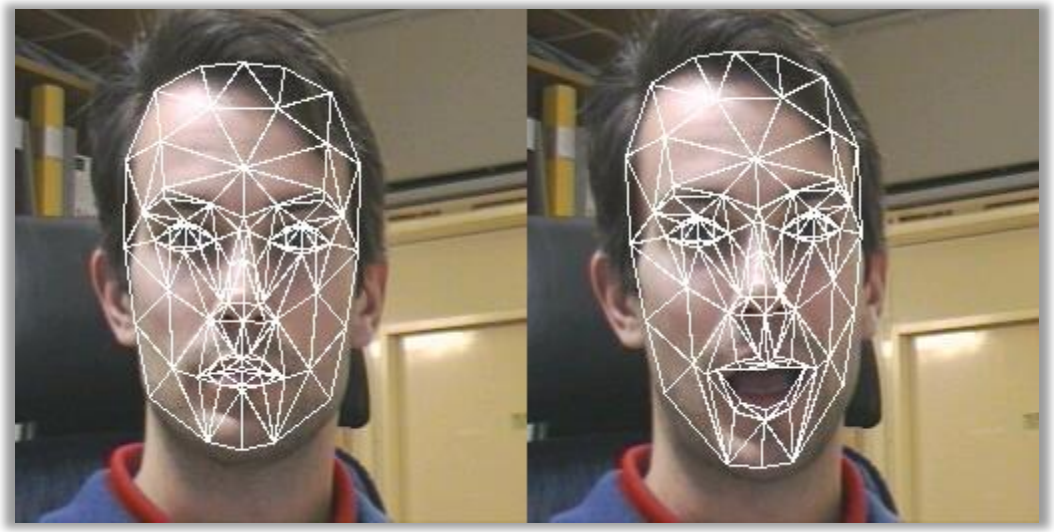


**Figura 7.** Reconocimiento Facial
Fuente: (Edwards, s/a)

### 1.8.3.5. Geolocalización.

Mediante la utilización del GPS, giroscopio y acelerómetro permite a los dispositivos móviles el reconocimiento de objetos o lugares mediante sus coordenadas.


**Figura 8.** Reconocimiento basado en Geolocalización.
Fuente: (Wikitude, 2014)

### 1.8.3.6. Sistema de Reconocimiento SLAM.

La tecnología SLAM (Simultaneous Localization And Mapping) es un sistema basado en el reconocimiento de objetos y ambientes en tiempo real, generalmente utilizado en campos como la robótica y la milicia.

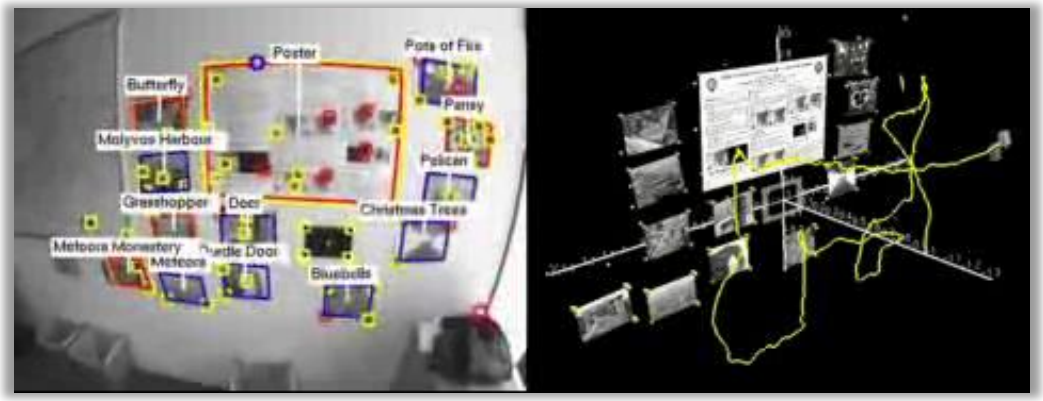



**Figura 9.** Sistema de Reconocimiento SLAM
Fuente: (Castle, 2009)

#### 1.8.3.7. Otros.

Entre estos podemos destacar los marcadores basados en códigos de barras y códigos QR


**Figura 10.** Marcadores basados en códigos de barra y códigos QR
Autor: Michael Freire

## 1.9. Herramientas de desarrollo.

En el campo de la RA se puede encontrar un gran número de herramientas o SDK para el desarrollo de aplicaciones, de las cuales se destaca las siguientes:

### 1.9.1. Metaio.

Metaio es una empresa de origen Alemán fundada en el 2003, enfocada principalmente al desarrollo de aplicaciones y soluciones de RA. Permite desarrollar aplicaciones nativas en IOS, Android y PC mediante un solo SDK (Metaio, Metaio web site, 2014).

#### 1.9.1.1. Características.

Entre sus principales características se puede citar las siguientes:

- ✓ Metaio ofrece la habilidad de reconocer cualquier imagen del mundo real, objeto o ambiente.
- ✓ La estructura modular del SDK de Metaio permite desarrollar una sola aplicación e implementarla en múltiples plataformas Android, IOS, PC, etc.
- ✓ Soporta HTML5 y JavaScript.
- ✓ Metaio posee su propio motor de renderizado.

- ✓ Soporta contenido basado en la nube.
- ✓ Soporta la mayoría de formatos de objetos 3D.
- ✓ Brinda un gran soporte a los usuarios con un helpdesk y una comunidad de desarrolladores muy activa.

"Metaio es ganador de múltiples premios, incluyendo el Concurso Tracking ISMAR y el 2013 Volkswagen Tracking" (Ortiz, 2014). Metaio además cuenta con un gran apoyo de diversas marcas reconocidas en todo el mundo como:

- ✓ Addidas
- ✓ Audi
- ✓ Epson
- ✓ Honda
- ✓ Intel
- ✓ Mitsubishi Electrics
- ✓ Redbull

A inicios del año 2014 "Metaio anuncio en el Consumer Electronics Show (CES) la integración prevista del sistema patentado de seguimiento de realidad aumentada 3D con la REALSENSE ™ Kit Intel ® Software Development (SDK)" (Ortiz, 2014). Más tarde Metaio anuncia una asociación con Epson para trabajar en unas gafas de realidad aumentada llamadas **SpaceGlasses Moverio BT– 200** (Zeis, 2014)**.**

#### 1.9.1.2. Metaio Creator.

Metaio Creator, es una aplicación que permite desarrollar aplicaciones de RA de manera rápida y simple sin requerir ningún tipo de conocimiento de programación previo, todo esto mediante una interfaz amigable e intuitiva. Permite importar fácilmente modelos de objetos en 3D, así como marcas o imágenes de seguimiento para aplicaciones de RA (Metaio, Metaio web site, 2014). Entre las principales características se pueden destacar las siguientes:

- ✓ No requiere ningún conocimiento de programación.
- ✓ Interfaz amigable e intuitiva drag and drop (arrastrar y soltar).
- ✓ Trabajar con los principales formatos de objetos 3D, donde se puede destacar los siguientes: fbx, da2, md2, obj.
- ✓ Permite la exportación de aplicaciones a Metaio SDK, así como exportar la configuración de tracking.

Su principal inconveniente es que, en su versión gratuita limita el número de objetos y marcadores para las aplicaciones.

#### 1.9.1.3. Metaio Cloud & Visual Search.

Mediante el uso de la computación en la nube (Cloud Computing) Metaio ofrece dos herramientas para las aplicaciones de RA.

**Metaio Cloud** es una herramienta que permite gestionar los distintos elementos de las aplicaciones de RA como: objetos 3D, marcadores y demás información relevante en un servidor web, mediante el cual los usuarios pueden recibir nuevos contenidos sin necesidad de actualizar su aplicación, siempre y cuando se disponga de una conexión a internet (Metaio, Metaio web site, 2014).

**Visual Search** permite mediante una conexión a internet reconocer gran cantidad de objetos de diversa índole que se encuentren almacenados en la base de datos Metaio, dicho repositorio cuenta con más de 1 millón objetos reconocibles del mundo real. (Metaio, Metaio web site, 2014)

Entre sus principales características se tiene:

- ✓ No requiere actualización constante de la aplicación para obtener nuevos contenidos.
- ✓ Aplicaciones mucho más pequeñas y.
- ✓ Menor consumo de recursos.
- ✓ Mayor flexibilidad y portabilidad.
- ✓ Requiere conexión constante de internet
- ✓ Puede tener cierto retardo en la obtención de algunos contenidos

### 1.9.2. Vuforia.

Detrás del SDK de Vuforia se encuentra la empresa Qualcomm, conocida especialmente por el desarrollo de microprocesadores para dispositivos móviles. Vuforia es una plataforma de desarrollo para aplicaciones de RA basada principalmente en la tecnología de reconocimiento de imágenes con soporte para desarrollar aplicaciones para IOS, Android and Unity 3D (Vuforia, 2014).

Entre las principales características de Vuforia se pueden destacar las siguientes:

- ✓ SDK completamente gratuito sin ninguna limitación.
- ✓ Sistema de reconocimiento o tracking basado tanto en marcadores como en reconocimiento de imágenes.
- ✓ Permite el desarrollo de aplicaciones de RA en Android e IOS.
- ✓ Mediante su plugin para la plataforma de Unity 3D permite el desarrollo tanto de aplicaciones como juegos basados de RA.
- ✓ Permite aplicaciones en la nube mediante Vuforia Cloud.
- ✓ Existe una gran comunidad de desarrolladores.

La principal característica de Vuforia, es que su SDK es completamente gratuito siendo el principal SDK utilizado por los desarrolladores que desean incursionar en el mundo de la RA, debido a lo cual cuenta con una comunidad de más de 95.000 desarrolladores en más de 130 países y cerca de 10000 aplicaciones (Vuforia, 2014). Entre otros, las principales empresas que utilizan Vuforia se encuentran las siguientes:

- ✓ Guinnes World Records
- ✓ Hot Wheels
- ✓ LEGO
- ✓ Toyota
- ✓ AUDI
- ✓ Johnson & Johnson

### 1.9.3. Aumentaty.

Aumentaty es una empresa de origen español, cuenta con dos SDK para el desarrollo de aplicaciones de RA Aumentaty Marker basada en el reconocimiento de marcadores y Aumentaty Markerless basada en el reconocimiento de imágenes. A través de sus respectivos SDK y su integración con Unity 3D permite a los desarrolladores total libertad para la creación de aplicaciones de RA, (Aumentaty, 2014).

#### 1.9.3.1. Características.

Entre sus principales características se pueden destacar las siguientes:

- ✓ Permite el desarrollo de aplicaciones para distintas plataformas, una misma escena de RA se puede visualizar tanto en sistemas de escritorio (Windows, MAC) como en Smartphone y tabletas Android e IOS.
- ✓ Permite interacciones con las marcas, poder situar botones virtuales, diseñar sus propios juegos, integrar la tecnología con sistemas de información tradicionales, etc.
- ✓ Sistema de reconocimiento basado en marcadores y reconocimiento de imágenes
- ✓ Integración con Unity 3D
- ✓ Empresa Española que brinda soporte en español

#### 1.9.3.1. Aumentaty Author.

Aumentaty Author forma parte de las herramientas para generar contenidos en RA. Utiliza tecnología de marcas fiduciales para reconocer el espacio tridimensional mostrado por la webcam y posicionar el contenido. “Ha sido diseñado teniendo en cuenta por encima de toda la facilidad de uso, que permite a personas sin ningún conocimiento de programación realizar contenidos en muy poco tiempo” (Aumentaty, 2014).

Una vez exportada la escena o contenido realizado, se puede visualizar y compartir instalando el visualizador de contenido 3D Aumentaty Viewer.

#### 1.9.3.2. Aumentaty Viewer.

Es el visualizador de contenidos de Aumentaty, donde se puede ver los proyectos de RA realizados con Aumentaty Author. Además permite compartir y visualizar proyectos de RA sin necesidad de tener instalado Aumentaty Author (Aumentaty, 2014).

### 1.9.4. Wikitude.

Wikitude es una empresa dedicada al mundo de la RA, son los creadores de Wikitude World Browser. Wikitude, pone a disposición su SDK para el desarrollar aplicaciones de RA cuyo principal objetivo es que sea lo más simple y fácil de usar. El SDK trabaja con los estándares

web HTML5, CSS y JavaScript. Con este enfoque, las programaciones no necesitan aprender nuevos lenguajes de programación. Este método reduce drásticamente los tiempos de desarrollo, permitiendo que las aplicaciones se desarrollen en un menor tiempo, y en última instancia, a un menor costo para el cliente final. Wikitude permite desarrollo para múltiples plataformas (IOS, Android, BlackBerry 10) con un mismo SDK, además de contar con plugin y extensiones para Unity 3D, PhoneGap y Titaniun Studio (Wikitude, 2014).

Entre sus principales características se pueden destacar las siguientes:

- ✓ Versión gratuita completamente funcional con marca de agua.
- ✓ Desarrollo de aplicación de RA en plataformas como: Android, IOS y BlackBerry.
- ✓ Sistema de reconocimiento basado en marcadores, reconocimiento de imagen y geolocalización.
- ✓ Soporte para las principales herramientas de desarrollo de aplicaciones móviles PhoneGap, Appcelerator y Unity 3D mediante la inclusión de plugins.
- ✓ Soporte para HTML5 y JavaScript.
- ✓ Permite el desarrollo de aplicaciones de RA sin requerir conocimientos de programación mediante la herramienta Wikitude Studio.
- ✓ Empresa altamente reconocida en el mundo de la RA

Wikitude World Browser aplicación galardona 2 años consecutivos (2010, 2011), como The Best AR Browser por la Augmented Planet Readers Choice. Dicha aplicación es una divertida, innovadora e informativa plataforma de RA que permite descubrir qué hay a tu alrededor de un modo completamente nuevo. Simplemente, se utiliza la cámara de un Smartphone para explorar el entorno y Wikitude añadirá información y contenido interactivo adicional sobre la imagen que muestra la cámara (Wikitude, 2014).

### 1.10. Evaluación de las herramientas de desarrollo.

- ✓ **Patrones reconocimiento (Tracking):** Aquí se tratara sobre cada una de las tecnologías de reconocimiento que utilizan los SDK, si utilizan marcadores, reconocimiento de imagen, GPS, etc.
- ✓ **Plataformas:** Se hace énfasis en las plataformas a las cuales permite desarrollar los SDK ya sean Android, IOS, PC, etc.
- ✓ **Características especiales:** Se considera a cada una de las virtudes que tienen cada herramienta; así también, las características que hacen único a cada SDK.

- ✓ **Soporte:** Se evalúa el soporte por parte de cada una de las empresas de los SDK y a su vez de su comunidad de desarrolladores.
- ✓ **Precio:** Y por último pero no menos importante, el costos que tiene cada uno de los SDK que se han seleccionado.

**Tabla 1.** Comparativa de las herramientas de desarrollo de Realidad Aumentada

| | Plataformas | Tracking (reconocimiento) | Características Esp. | Soporte | Precio (dólares) |
|---|---|---|---|---|---|
| **Metaio** | • Android<br>• IOS<br>• Web<br>• Windows<br>• MAC | • Marcadores (Marker)<br>• Reconocimiento de Imagen (NFT)<br>• Reconocimiento Facial<br>• Geolocalización (GPS)<br>• Localización y mapeado simultaneo (SLAM) | • Integración con Unity 3D<br>• Soporte OpenGL<br>• Plataforma Independiente<br>• Html5 & JavaScript<br>• Metaio Creator<br>• Metaio Cloud | • Comunidad con más de 80000 desarrolladores y más de 1000 app<br>• Portal para desarrolladores<br>• Help Desk | • Full: $5490<br>• Básica: $3490<br>• Versión Gratuita con marca de agua |
| **Vuforia** | • Android<br>• IOS | • Marcadores (Marker)<br>• Reconocimiento de Imagen (NFT) | • Integración con Unity 3D<br>• Vuforia Cloud | • Comunidad con más de 95000 desarrolladores y más de 8500 app<br>• Portal para desarrolladores | • Gratuito |
| **Wikitude** | • Android<br>• IOS<br>• BlackBerry 10 | • Marcadores (Marker)<br>• Reconocimiento de Imagen (NFT)<br>• Geolocalización (GPS) | • Integración con Unity 3D, PhoneGap y Appcelerator<br>• Wikitude Studio<br>• Html5 & JavaScript | • Más de 5 años en el mercado y una gran comunidad | • Professional: $2300<br>• Versión Gratuita con marca de agua |
| **Aumentaty** | • Android<br>• IOS<br>• Windows<br>• MAC | • Marcadores (Marker)<br>• Reconocimiento de Imagen (NFT) | • Integración con Unity 3D | • Empresa española, brinda soporte en español | • 2 SDK ($1600 + $2700)<br>• No hay versión de prueba |

Autor: Michael Freire

Después de haber evaluado y analizado cada uno de los principales SDK de desarrollo de aplicaciones de RA, se ha seleccionado Metaio SDK como la plataforma más robusto en cuanto a funcionalidad y soporte. Metaio permite trabajar con algunas plataformas móviles y de escritorio, además de contar con las principales tecnologías de seguimiento (tracking). Adicionalmente, cuenta con soporte de parte de la empresa de Metaio, lo cual se traduce en multitud de tutoriales y documentación disponible para el desarrollador, aparte de contar con una comunidad de desarrolladores muy activa.

# CAPITULO 2
# METAIO SDK

## 2.1. Introducción.

Después de haber analizado las diversas opciones en cuanto a SDK de desarrollo de aplicaciones de RA, Metaio ha sido seleccionado como la plataforma idónea para la realización del actual proyecto de tesis. Metaio es una empresa plenamente reconocida en el mundo de la RA con más de 10 años de experiencia en el sector, actualmente cuenta con cientos de proyectos de RA desarrollándose y decenas de empresas que respaldan su trabajo.

A lo largo de este capítulo repasaremos cada una de las principales virtudes y características que hacen de Metaio uno de los SDK más robustos para el desarrollo de aplicaciones de RA.

## 2.2. Metaio SDK.

Metaio se presenta como la opción más robusta y fiable a la hora de abordar el desarrollo de cualquier aplicación de RA, cuenta con su propio leguaje AREL (Augmented Reality Experience Language), además de permitir el desarrollo nativo en las principales plataformas. Metaio SDK, gracias a la integración de un motor propio de renderizado logra alcanzar un alto rendimiento en lo que a visualización y representación de objetos 3D se refiere.

El SDK de Metaio tiene una versión gratuita con algunas limitaciones, en lo que respecta a las tecnologías de tracking y la inclusión de una marca de agua en las aplicaciones, pero conserva su potencial en el procesamiento de objetos y escenarios de RA.

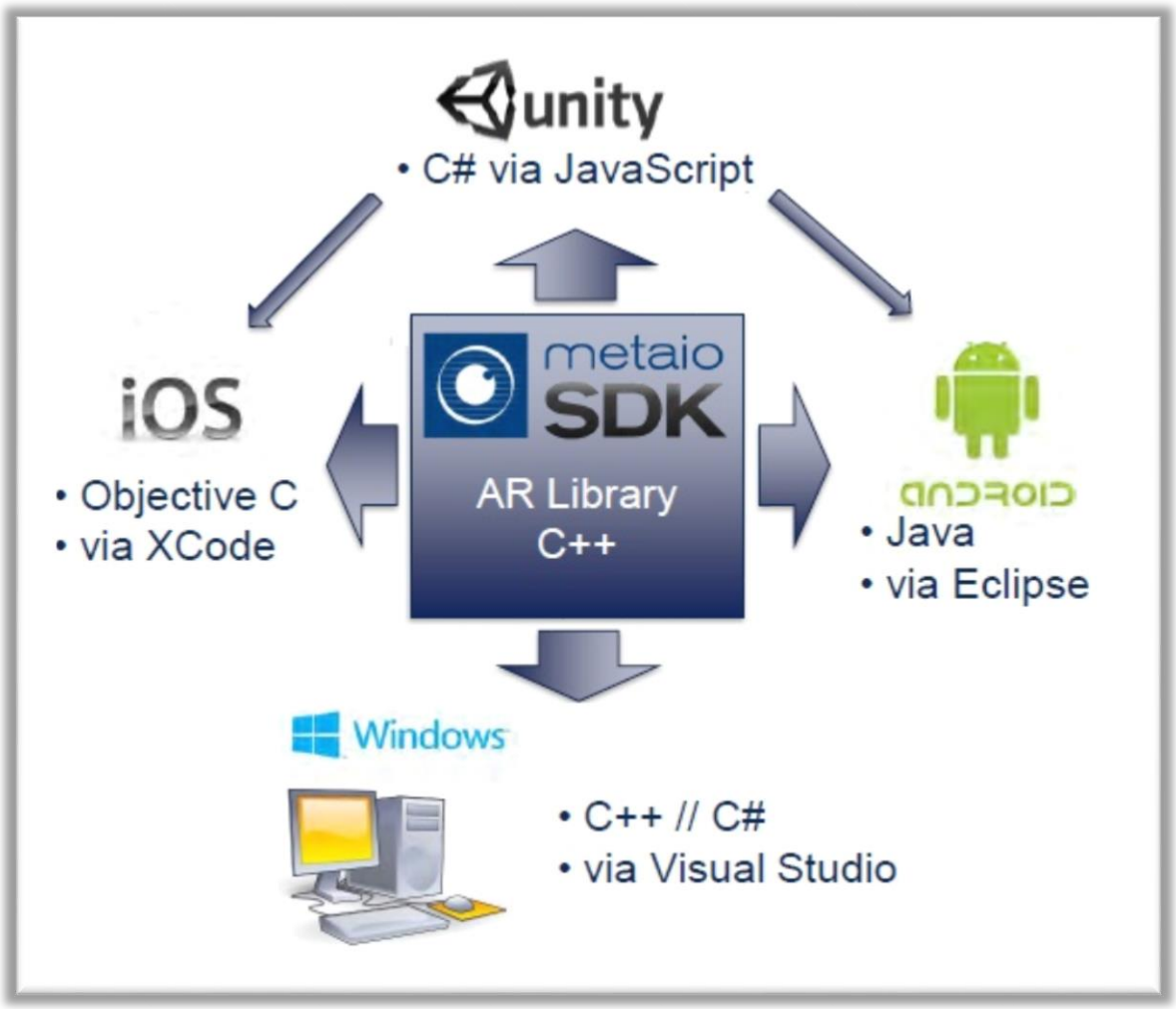



**Figura 11.** Plataformas Metaio SDK
Fuente: (Greiner, 2013)

Como se puede observar en el gráfico anterior Metaio SDK permite desarrollar aplicaciones de RA para las principales plataformas del mercado, brindando soporte para el desarrollo nativo en los leguajes de programación propios de dichas plataforma como Objective-C para IOS, Java para Android, C++ y C# para Windows, etc., también cuenta con un plugin para Unity (plataforma para el desarrollo de videojuegos).

Adicionalmente a partir de su versión 5.0 Metaio SDK lanza su propio Lenguaje de Programación AREL, con el que permite desarrollar una sola aplicación e implementarla en múltiples plataformas como: Android IOS, PC.

## 2.3. Plataformas de desarrollo.

Metaio SDK provee soporte para el desarrollo nativo de aplicaciones de RA para las principales plataformas del mercado, adicionalmente mediante su propio lenguaje de programación de RA, llamado AREL brinda soporte para el desarrollo de aplicaciones de RA independientes de la plataforma.

### 2.3.1. Unity (Multi-plataforma / C#).

Unity es una plataforma para el desarrollo de videojuegos con un gran prestigio dentro de esta industria; cuenta con un motor de renderizado totalmente integrado con un conjunto completo de herramientas intuitivas y flujos de trabajo rápido para crear contenido 3D interactivo. El ecosistema Unity, permite desarrollo multi-plataforma facilitando enormemente el desarrollo de videojuegos, entre las principales plataformas de desarrollo de Unity se destacan Android, IOS, PC, PlayStation, Xbox. (Unity, 2014).

Entre otras, las principales características expuestas en (Unity, 2014), son:

- ✓ **Flujo de trabajo:** Permite agrupar rápidamente sus escenas en un espacio de trabajo con un editor intuitivo y extensible.
- ✓ **Calidad:** Permite crear un juego con gran fidelidad visual, auditiva y acción al máximo con un nivel AAA que se ejecuta suave y limpiamente en cualquier pantalla
- ✓ **2D y 3D:** Unity tiene herramientas dedicadas para la creación de contenido 2D y 3D con flujos de trabajo eficientes que utilizan convenciones compartidas
- ✓ **Mecanim:** El sistema de animación poderoso y flexible, con características únicas, de Unity da vida a cualquier personaje u objeto con movimiento natural y fluido.
- ✓ **Rendimiento:** Rendimiento confiable, framerate (tasa de imágenes por segundo) uniforme y soberbias experiencias de juego en las plataformas de destino.
- ✓ **Multiplataforma:** Ningún otro motor de juegos permite elegir entre tantas plataformas de publicación con prácticamente ningún esfuerzo de implementación.
- ✓ **Colaboración:** Control de versión completa para todos los activos del juego; obtiene instantáneamente los cambios de otros miembros del equipo.

Como se puede observar Unity es una plataforma especializada principalmente en el desarrollo de videojuego para diversos dispositivos entre los que podemos destacar los dispositivos móviles basados en Android e IOS. Unity, gracias a la integración con el plugin

de Metaio y sus herramientas de modelado y renderizado 3D es capaz de crear videojuegos y aplicaciones basados en RA muy robustos.

### 2.3.2. Xcode (IOS / Objetive-C).

Xcode es el Entorno de Desarrollo Integrado (IDE) de Apple para el desarrollo de aplicaciones de sus respectivos sistemas operativos. La suite de software de desarrollo y herramientas de desarrollo de Xcode integra los frameworks de Cocoa y Cocoa Touch para la creación de aplicaciones en OSX e IOS. La nueva tecnología de compilador de Apple LLVM analiza el código constantemente, manteniendo cada símbolo que aparece en el depurador LLDB consistente con el editor y el compilador (Apple, 2014).

Para poder trabajar con aplicaciones móviles es necesario instalar el SDK de IOS, mediante este SDK Xcode se puede construir, instalar y correr Cocoa Touch apps; es decir, aplicaciones IOS a través de su simulador de IOS. El lenguaje de programación nativo de desarrollo para IOS es el Objetive-C, el cual es un lenguaje de programación orientado a objetos.

### 2.3.3. Eclipse (Android / Java).

La naturaleza fundamentalmente modular de la plataforma Eclipse hace que sea fácil de instalar características adicionales dentro de Eclipse, lo cual permite le extenderse usando otros lenguajes de programación como son C/C++ y Phyton, o incluso trabajar con lenguajes para procesado de texto como LaTex, o aplicaciones en red como Telnet, entre otros. El IDE de Eclipse está estructurado como una colección de plugins, donde cada plugin contiene el código que proporciona algunas de las funcionalidades del mismo. (Eclipse, 2014).

Eclipse es una plataforma modular lo que permite que sea el usuario quien decida qué es lo que necesita dentro de Eclipse, lo cual es de vital importancia a la hora de maximizar los recursos dentro del entorno de desarrollo.

Eclipse para su funcionamiento requiere la instalación del JDK de Java (Java Development Kit) y para el desarrollo de aplicaciones móviles basadas en Android es necesario la instalación del SDK de Android (Software Development Kit). El SDK provee las API necesarias para construir, instalar y correr aplicaciones Android, mediante el lenguaje de programación

Java que al igual que Objetive-C es orientado a objetos (Google, Android Developer web site, 2014).

### 2.3.4. Visual Studio (PC Windows / C++, C#).

> Visual Studio es una colección completa de herramientas y servicios para desarrollar aplicaciones de escritorio, la Web, dispositivos y la nube. Tanto si va a crear su primera aplicación para la Tienda Windows, como si va a compilar un sitio web compatible con los últimos exploradores, puede aprovechar los conocimientos que ya tiene con el entorno de desarrollo vanguardista que ofrece Visual Studio para lenguajes .NET, HTML/JavaScript y C++. Para aquellos equipos que trabajen en varias plataformas, Visual Studio proporciona un entorno de colaboración flexible que permite conectar con otras herramientas de desarrollo, como Eclipse y Xcode. (Microsoft, 2014).

Gracias a la integración de Metaio SDK dentro Visual Studio se puede realizar aplicaciones de RA para equipos de escritorio (Windows). El lenguaje utilizado para el desarrollo de aplicaciones de RA bajo Visual Studio es el C++ o C#.

## 2.4. Arel (Augmented Reality Experience Language).

Metaio a partir de la versión 5.0 de su SDK pone a disposición de desarrolladores el lenguaje AREL (Augmented Reality Experience Language) para el desarrollo de aplicaciones de RA cuya principal característica es la independencia de plataforma; es decir, solo se desarrolla una única aplicación, la cual mediante el leguaje AREL es capaz de ejecutarse en cualquier plataforma.

> AREL es la unión de JavaScript con la API del SDK Metaio en combinación con una definición de contenidos XML estático. AREL permite trabajar conjuntamente con tecnologías web comunes como HTML5, XML y JavaScript. También se puede utilizar para la creación de canales junaio, así como para trabajar con el SDK Metaio. Para la creación de la interfaz gráfica de usuario AREL utiliza una capa HTML 5 lo cual permite un desarrollo independiente de la plataforma. (Metaio, Metaio Developer Portal, 2014).

> Similar a una página web, una aplicación de realidad aumentada desarrollada en AREL cuenta con una parte estática, AREL XML, que define todo el contenido y los enlaces y una parte dinámica, AREL JavaScript, que define las interacciones y comportamientos de los objetos individuales o de toda la escena. Por supuesto que no es necesario definir todos los contenidos de forma estática en un archivo XML, también se puede cargar dichos contenidos mediante código JavaScript de ser necesario, al igual que lo haría con el SDK Metaio desde código nativo (Metaio, Metaio Developer Portal, 2014)

A pesar de las diversas virtudes que presenta AREL para el desarrollo de aplicaciones de RA multiplataforma, aun es un lenguaje muy joven con diversas limitaciones, lo cual, dicho sea de paso no permite que AREL se afiance como el lenguaje de referencia para el desarrollo de aplicaciones de RA. Desde el punto de vista del desarrollador, al abordar proyectos de gran envergadura, es preferible trabajar nativamente en cada plataforma con Metaio SDK, ya que cuenta con un mayor soporte en cuanto a funciones y características que facilitan su desarrollo, de esta forma AREL es rezagado a aplicaciones sencillas o poco ambiciosas debido a sus limitaciones.

### 2.4.1. AREL Y Metaio SDK.

Con Metaio SDK y el lenguaje de programación AREL, es posible construir una experiencia de RA independientemente de la plataforma de destino; en lugar de utilizar los lenguajes de programación específicos de cada plataforma: Java para Android, Objective C para iOS y C + + para Windows, se puede utilizar el leguaje AREL, con lo cual únicamente se construye una aplicación que se puede desplegar o implementar en las diversas plataformas soportadas. De esta manera se puede desarrollar una aplicación que sea compatible con las plataformas de preferencias, lo cual permite mejorar ampliamente la cuota de mercado con poco esfuerzo adicional (Metaio, Metaio Developer Portal, 2014)

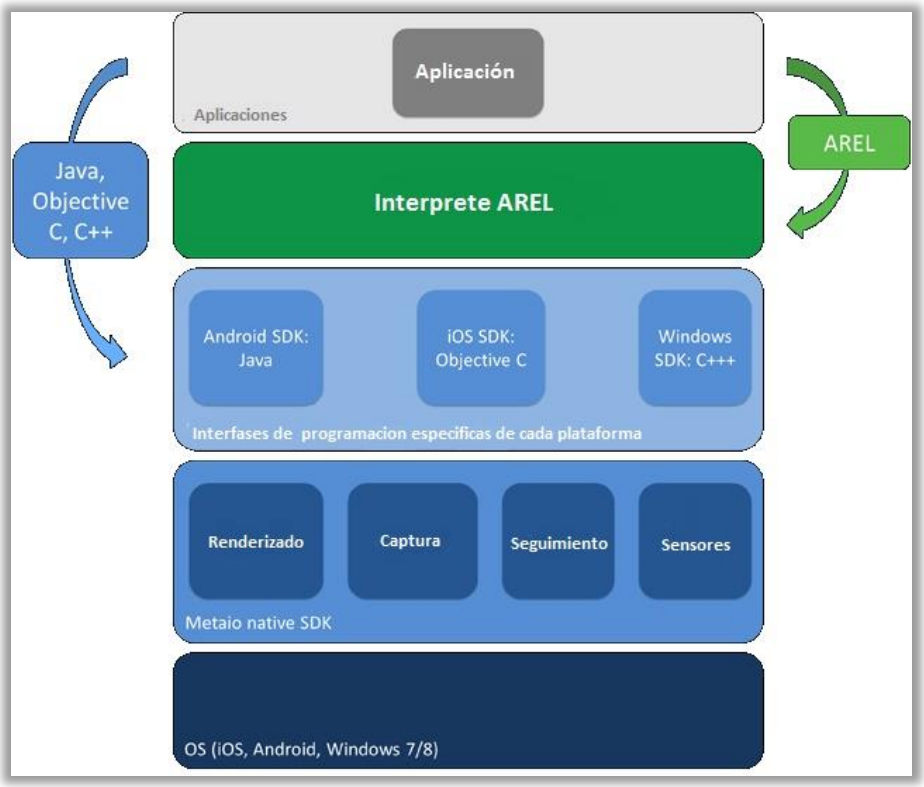



**Figura 12.** Arquitectura de AREL
Fuente: (Metaio, Metaio Developer Portal, 2014)

En la figura anterior se aprecia la arquitectura de una aplicación construida en AREL, en la que AREL sirve de enlace entre la capa de Aplicación y la de los SDK específicos de cada plataforma, lo cual dicho sea de paso, permite a AREL proporcionar independencia de plataforma. Adicionalmente de ser necesario, se puede implementar funcionalidades específicas de cada plataforma nativa como se observa en la figura, aunque, con esto último, se perdería la principal característica de AREL, que es la independencia de plataforma.

### 2.4.2. Funcionamiento de AREL.

Para su funcionamiento AREL hace uso de los componentes detallados en (Metaio, Metaio Developer Portal, 2014):

La parte XML define el contenido que se debe cargar en la aplicación; es decir, los objetos 3D, adicionalmente también permite definir las propiedades iniciales de dichos contenidos como el tamaño, ubicación, animación, etc.

La capa de HTML5 define la interfaz gráfica de usuario y la interacción con Metaio SDK a través de  JavaScript. Dicha interfaz se puede diseñar como cualquier sitio web para móviles, con la excepción de que el fondo de dicho sitio web es transparente, para de esta manera, permitir al usuario observar la vista de la cámara y cualquier contenido de RA proyectado en ella

Todas las devoluciones de llamadas del SDK Metaio se remiten a la lógica de aplicación JavaScript, por lo que pueden reaccionar a éstos en tiempo de ejecución; por ejemplo, el evento en que se empieza a reconocer el patrón de seguimiento (marcador).


**Figura 13.** Funcionamiento de AREL
Fuente: (Metaio, Metaio Developer Portal, 2014)

En la gráfica anterior, se puede observar la estructura de una aplicación desarrollada bajo AREL, en donde la **Capa HTML 5** contiene la interfaz gráfica de usuario y define el comportamiento de la misma, la **Capa AREL JS** es la que contiene la lógica de la aplicación y las herramientas necesarias para generar los contenidos de RA, mientras que la **Capa de Renderizado OpenGL** permite procesar y cargar en pantalla los contenidos de RA en tiempo real.

A continuación mediante un ejemplo se explicara más a detalle la estructura de una aplicación de RA desarrollada bajo AREL

### 2.4.3. Ejemplo Básico AREL.

A continuación se muestra mediante un ejemplo el flujo básico de una aplicación de RA realizada en AREL, en el cual se observa como los distintos componentes o capas de AREL interactúan entre sí. (Metaio, Metaio Developer Portal, 2014):

#### 2.4.3.1. Definición de contenido estático.

El siguiente es un ejemplo básico de como el código AREL XML luce.

```
<?xml version="1.0" encoding="UTF-8"?>
<results>
    <arel>simpleAREL.html</arel>  <!-- HTML GUI that should be loaded -->
    <object id="Tiger">
        <assets3d>
            <model>Assets/tiger.md2</model>
            <texture>Assets/tiger.png</texture>
            <transform>
             <translation>
                 <x>10.0</x><y>0.0</y><z>0.0</z>
             </translation>
             <rotation type="eulerdeg">
                 <x>90.0</x><y>0.0</y><z>0.0</z>
             </rotation>
            </transform>
            <properties>
             <coordinatesystemid>1</coordinatesystemid>
            </properties>
        </assets3d>
    </object>
</results>
```

**Figura 14.** Código AREL XML
Fuente: (Metaio, Metaio Developer Portal, 2014)

La etiqueta <arel> define la GUI (Interfaz Gráfica de Usuario) HTML que debe ser cargada. La etiqueta object contiene el contenido 3D (objeto 3D) que debe ser cargado por Metaio SDK. En este ejemplo, el objeto es un modelo M2D el cual es rotado 90° en dirección al eje x y trasladado 10 mm en dirección al eje x.

### 2.4.3.2. Capa HTML 5.

El siguiente código detalla una simple interfaz HTML 5 en el cual e muestra el enlace con AREL JavaScript.

simpleAREL.html

```
<html>
<head>
   <!-- Integrates the arel javascript bridge -->
   <script type="text/javascript" src="http://dev.junaio.com/arel/js/arel.js"></script>
   <!-- Includes application logic -->
   <script  type="text/javascript" src="logic.js"></script>
</head>

<body>
</body>
</html>
```

**Figura 15.** Fragmento de Código HTM 5
Fuente: (Metaio, Metaio Developer Portal, 2014)

### 2.4.3.3. Lógica de Aplicación JavaScript.

En el siguiente código se muestra la función que es ejecutada luego que el contenido de AREL ha sido cargado (arel.sceneReady), en el cual el objeto con el ID “Tiger” se le da una escala de factor 2 y luego se define la llamada a la función que recibe el evento desde el sistema de seguimiento o tracking.

```
1.  function trackingHandler(type, param)
2.  {
3.     // called when a pattern is tracked / lost
4.  }
5.
6.  arel.sceneReady(function()
7.  {
8.     // scale up loaded 3D model by factor 2
9.     arel.Scene.getObject("Tiger").setScale(new arel.Vector3D(2, 2, 2));
10.
11.    // set a listener for tracking events
12.    arel.Events.setListener(arel.Scene, trackingHandler);
13. });
14.
```

**Figura 16.** Fragmento de Código JavaScript
Fuente: (Metaio, Metaio Developer Portal, 2014)

## 2.5. Tipos de marcadores soportados por Metaio.

Dentro de los SDK de desarrollo de aplicaciones de RA, Metaio es el que cuenta con el mayor soporte para los diversos tipos de marcadores existentes. Metaio es capaz de reconocer todos los tipos de marcadores descritos en la sección 1.8.3

De entre todos los tipos de marcadores soportados por Metaio se ha seleccionado a los “Marcadores de Imagen” por ser los más robustos y los que más se ajustan a las necesidades actuales de la aplicación de RA que se está desarrollando.

## 2.6. Formatos de objetos 3D soportados por Metaio.

El modelado en 3D es una ardua tarea que requiere mucho tiempo de trabajo. Metaio a través de su SDK brinda soporte nativo para los principales formatos de modelado en 3D como obj, fbx y md2, lo cual brinda una mayor flexibilidad en la elección de la herramienta de modelado 3D. A su vez, Metaio también cuenta con un formato propio como es el mfbx, el cual nos brinda un mayor rendimiento en contraste con los formatos anteriormente mencionados (Renukdas, Ghundiyal, Gadgil, & Pathare, 2013).

Adicionalmente Metaio proporciona una herramienta llamada FBXMeshConverter el cual permite transformar de manera fácil los formatos de modelado 3D obj, fbx y m2d al formato de mfbx propiedad de Metaio con el fin de maximizar el rendimiento de las aplicaciones.

# CAPITULO 3
# DESARROLLO

## 3.1. Introducción.

Hoy en día la educación a distancia tradicional a través de libros y guías de estudio con contenidos estáticos, en los cuales el estudiante no tiene interacción alguna, se terminan de una u otra forma volviendo en su contra con contenidos poco claros o trabajados. Es en este punto, donde las nuevas tecnologías ayudan a la educación a subsanar esos vacíos didácticos a través de herramientas tecnológicas, con contenidos audiovisuales y aplicaciones interactivas de aprendizaje, donde el estudiante pasa; de ser un mero espectador, a ser el actor principal. Logrando de esta manera estimular el aprendizaje, e incentivar al estudiante, mayormente con el uso de la RA en aplicaciones interactivas de aprendizaje.

## 3.2. Ámbito de la aplicación.

A fin de potenciar la calidad de la educación a distancia, la UTPL desarrolla una herramienta de apoyo al aprendizaje mediante la utilización de nuevas herramientas tecnológicas. Con esta premisa y con el fin de hacer algo diferente a la educación tradicional, se optó por crear una aplicación de RA, en la que los estudiantes puedan interactuar con diversos contenidos, incentivando de esta forma el aprendizaje activo.

El componente académico “Desarrollo de la Inteligencia” en la que se construyeron 3 objetos 3D para la aplicación de RA

Para entender mejor como ayudar al estudiante en el aprendizaje a distancia se propone el siguiente ejemplo:

“*Un estudiante se encuentra leyendo el contenido sobre cerebro humano y cada una de sus partes en la guía didáctica o libro base, esto puede resultar poco intuitivo si únicamente se cuenta con la ilustraciones presentes en dicha guía, las que en muchas ocasiones son poco detalladas y muy difíciles de entender si no se cuenta con una representación física que afiance dichos conocimientos teóricos*”.

Mediante este ejemplo se puede apreciar un grave problema con los contenidos estáticos, es aquí donde cobra vital importancia la RA como herramienta de apoyo al aprendizaje del estudiante. Volviendo al ejemplo anterior, una de las formas de facilitar aprendizaje, es con una representación 3D del cerebro humano a través de una aplicación de RA en la que el estudiante pueda interactuar.

### 3.3. Selección de dispositivos.

Como cualquier desarrollo de software primeramente se debe determinar hacia qué tipo de dispositivos va dirigida la aplicación de RA, ya sea un Smartphone, una Tablet, un PC, etc. Así mismo, es importante delimitar la plataforma (Sistema Operativo) para la aplicación: Android, IOS, Windows, MAC, etc.

Como los estudiantes de la modalidad abierta y a distancia de la UTPL ya cuentan con Tablets Android PAPYRE Pad 1015, se ha escogido a esta como el dispositivo base de la aplicación de RA, cuenta con las siguientes características:

**Tabla 2.** Características de la Tablet Android PAPYRE Pad 1015

| Características | Datos |
|---|---|
| **Modelo** | PAPYRE Pad 1015 |
| **Sistema Operativo** | Android 4.2 |
| **Pantalla** | 10.1” 1280 x 800 |
| **Procesador** | Intel® Clover Trail+ Z2520 (Dual Core) – 1.2GHz |
| **RAM** | 1Gb LPDDR2 |
| **Almacenamiento** | 16 GB |
| **Cámara** | 2.0 Mpx |

Autor: Michael Freire

Como se puede observar en la tabla anterior la plataforma elegida para el desarrollo de la aplicación de RA es una Tablet con Sistema Operativo Android.

En un principio se planteó la idea de que la aplicación de realidad aumentada también fuera implementada en Smartphone Android, pero fue descartado debido a las reducidas

dimensiones de pantalla que estos presentan, las cuales en su gran mayoría no superan las 4 pulgadas. Y debido a que el tamaño de pantalla es un factor determinante para la interacción con los objetos 3D, se decidió brindar soporte a únicamente dispositivos con un tamaño de pantalla mínimo de 7 pulgadas. Quedando delimitado el soporte de la aplicación para dispositivos que cumplan los siguientes requisitos mínimos:

### 3.3.1. Requisitos Mínimos.

**Tabla 3.**Características Mínimas del dispositivo Android para la aplicación de RA

| Características | Datos |
|---|---|
| **Sistema Operativo** | Android 4.0 |
| **Pantalla** | Touhscreen de 7” |
| **Procesador** | 1 Ghz |
| **RAM** | 512 MB |
| **Cámara** | 2.0 Mpx |

Autor: Michael Freire

### 3.3.2. Requisitos Recomendados.

**Tabla 4.** Características Recomendadas del dispositivo Android para la aplicación de RA

| Características | Datos |
|---|---|
| **Sistema Operativo** | Android 4.4 |
| **Pantalla** | Touhscreen de 10” |
| **Procesador** | Doble Núcleo 1Ghz |
| **RAM** | 1 GB |
| **Cámara** | 5 Mpx |

Autor: Michael Freire

## 3.4. Selección de contenidos.

Una vez seleccionado el ámbito de la aplicación, la asignatura y el dispositivo al cual va dirigida la aplicación de RA, es el momento de trabajar con los contenidos de la aplicación, para lo cual conjuntamente con los tutores de la asignatura de Desarrollo de la Inteligencia se tuvo algunas reuniones con el fin de determinar los contenidos claves, y de mayor importancia para trabajarlos en la aplicación.

En las entrevistas se acordó trabajar con un máximo de 4 objetos 3D dentro de la aplicación acorde a los requerimientos de Tutor principal de la asignatura. Adicionalmente con el fin de mejorar la interactividad de la aplicación de RA se decidió que los objetos 3D fueran subdivididos en varias partes, por ejemplo, el cerebro humano está dividido en las siguientes partes: lóbulo frontal, lóbulo parietal, lóbulo occipital, cerebelo, etc.; cada una de estas partes es un objeto 3D individual e independiente que conjuntamente formarían el cerebro humano.

### 3.4.1. Plantilla: Recolección de Objetos.

Para la recolección de información de cada uno de los objetos se utilizó la siguiente plantilla acompañada de una imagen del objeto que se pretende desarrollar:

**Tabla 5.** Encabezado de la plantilla para la recolección de información de los objetos

| DIRECCIÓN DE TECNOLOGIAS PARA LA EDUCACION | | | |
|---|---|---|---|
| **DOCENTE:** | | **TITULAR:** | |
| **PERIODO:** | | **AUXILIAR:** | |
| **TITULACION:** | | | |
| **FECHA:** | | | |
| **COMPONENTE:** | | **TIPO:** | |

Autor: Michael Freire

**Tabla 6.** Cuerpo de la plantilla para la recolección de información de los objetos

| OBJETO | | | | | | |
|---|---|---|---|---|---|---|
| **Titulo Unidad** | | | | | | |
| | **N° de Unidad en Libro Base** | | | **N° de Unidad es Plan Docente** | | |
| **N°** | **Nombre Modelo** | **Descripción** | | | | |
| **Principal** | | | | | | |
| **1** | | | | | | |
| **2** | | | | | | |
| **3** | | | | | | |
| **4** | | | | | | |
| **5** | | | | | | |

Autor: Michael Freire

A continuación se muestra una imagen preliminar tomada como base para la creación posterior el objeto cerebro:

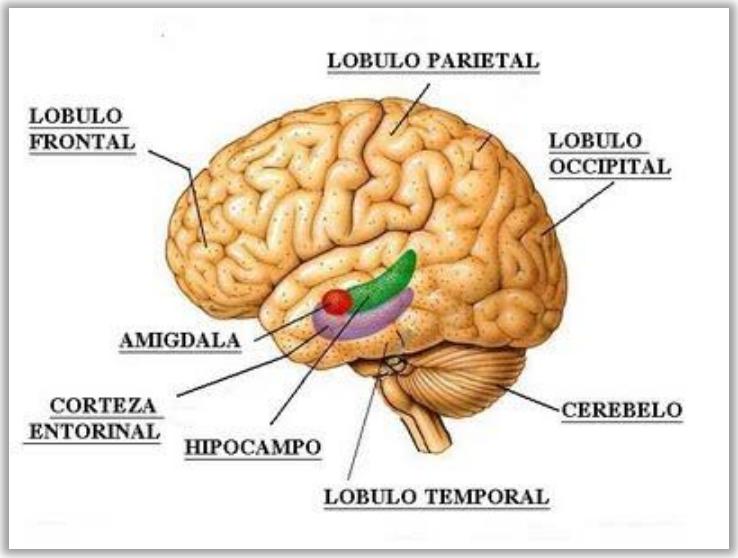



**Figura 17.** Imagen preliminar de la creación del objeto cerebro
Fuente: (Docente, 2015)

### 3.5. Estrategias para la creación de objetos 3D.

Para la creación de objetos 3D es preciso considerar:

- ✓ Utilizar la misma escala en la creación de todos los objetos 3D que se van a utilizar en la aplicación.
- ✓ Tratar de minimizar el número de vértices (polígonos) de los objetos 3D, con el objetivo de minimizar su tamaño y maximizar el rendimiento en dispositivos de poca capacidad computacional.
- ✓ Tomar en cuenta la interacción con los objetos que se utilizará en la aplicación. Por ejemplo, si se tiene un vehículo en 3D, y si se desea interactuar con cada una de sus partes (ruedas, puertas, chasis, etc.), es necesario que cada parte sea un objeto independiente para facilitar su tratamiento dentro de la aplicación.
- ✓ Las texturas de los objetos 3D deben ser trabajadas como imagen, ya que si se utiliza otro método, dificulta su reconocimiento y adecuada visualización.
- ✓ La ubicación del objeto 3D en escena (plano (x, y, z)) debe ser tomada en cuenta desde la creación del objeto, para evitar la superposición de objetos entre sí. Esto es de vital importancia cuando se trabaja con múltiples objetos en un mismo marcador.

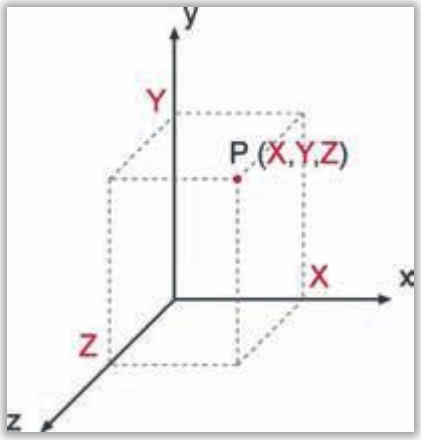



**Figura 18.** Plano 3D (x,y,z).
Autor: Michael Freire

En la figura 18, se observa cada uno de los planos x, y z, donde el punto P representa la ubicación del objeto 3D en el espacio.

### 3.6. Funciones básicas dentro de la aplicación.

Con el fin de fomentar el aprendizaje y maximizar la interactividad de los contenidos de RA se detallan las principales características o funciones que la aplicación de RA debe cumplir:

#### 3.6.1. Tamaño.

Al trabajar con objetos 3D se debe disponer de una función para redimensionar los contenidos en 3D (aumentar, o disminuir su tamaño), esto beneficia su visualización y facilita la interacción por parte de los usuarios. Es muy importante, manejar correctamente la escala del objeto (escala original) para cada uno de los ejes (x,y,z) por igual, con el fin de mantener las proporciones originales del objeto.


**Figura 19.** Metaioman, escala 1:1 (ejemplo de tamaño)
Autor: Michael Freire

En el caso de que el objeto principal este compuesto de varios objetos, se debe tomar en cuenta que todos los objetos sean redimensionados a la misma escala con el fin de evitar la superposición de los mismos, salvo que la interacción así lo requiera.

### 3.6.2. Rotación.

Otra función básica es la rotación de los objetos, generalmente no es muy tomada en cuenta, puesto que al rotar el marcador se puede rotar el objeto en la posición deseada; sin embargo, esto no resulta muy cómodo cuando dicho marcador se encuentra impreso dentro de un libro o texto, de ahí la importancia de disponer de un marcador transportable o de definir funciones de rotación de objetos en 3D, en la aplicación de RA.

A diferencia del tamaño en la que se debía conservar la misma escala en cada uno de los ejes, la rotación de objetos se la puede trabajar en cada eje por separado; es decir, se puede rotar el objeto 3D únicamente en un eje.

El movimiento de rotación de un objeto 3D viene denotado por el radio de la circunferencia pi ($\pi = 3.1416$ ...); es decir, si se desea rotar un objeto 180 grados, el valor a utilizar es $\frac{\pi}{2}$; si se desea rotar 90 grados, el nuevo valor es $\frac{\pi}{4}$; y así, sucesivamente.

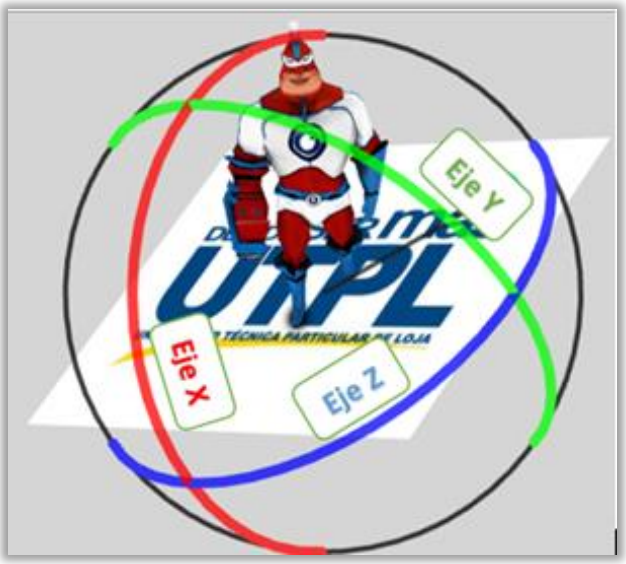



**Figura 20.** Metaioman (ejemplo de rotación)
Autor: Michael Freire

### 3.6.3. Interacción.

Otra función importante dentro de la RA es la interactividad; es decir, permitir que el estudiante pueda interactuar con los objetos 3D más allá de redimensionar o rotar los contenidos en 3D:

#### 3.6.3.1. Mensajes desplegables.

Permite proporcionar información adicional útil de los objetos representados, como por ejemplo, al seleccionar un objeto virtual este despliega mayor información sobre el mismo, así:


**Figura 21.** Ejemplo de despliegue de información adicional
Autor: Michael Freire

#### 3.6.3.2. Audio.

Otra función importante es la de contar con contenidos auditivos dentro de la aplicación que faciliten la inmersión, por ejemplo si se selecciona un objeto se reproduzca un sonido de “Clic” o también se puede reproducir un audio informativo acerca del objeto selecciona, etc.

## 3.7. Objetos 3D.

Una vez definidos los contenidos 3D de la aplicación se decidió emprender una búsqueda de dichos contenidos en varios repositorios de contenidos 3D, con el fin de reutilizar contenidos existentes de estos repositorios, pero el proceso de búsqueda fue infructuoso y no se pudo encontrar ningún contenido que pueda ser utilizado o adaptado a la aplicación de RA. Por este motivo se tomó la decisión de crear dichos contenidos, para lo cual se pidió a los tutores de la asignatura adjuntar un esquema o imagen del objeto requerido en la plantilla anterior, con el fin de poder transformar dicha imagen en un contenido u objeto 3D de aprendizaje.

A continuación, se detalla cada uno de los objetos 3D que se crearan para la asignatura Desarrollo de la Inteligencia, acorde a la plantilla anteriormente mencionada:

### 3.7.1. Objeto 1: Cerebro.

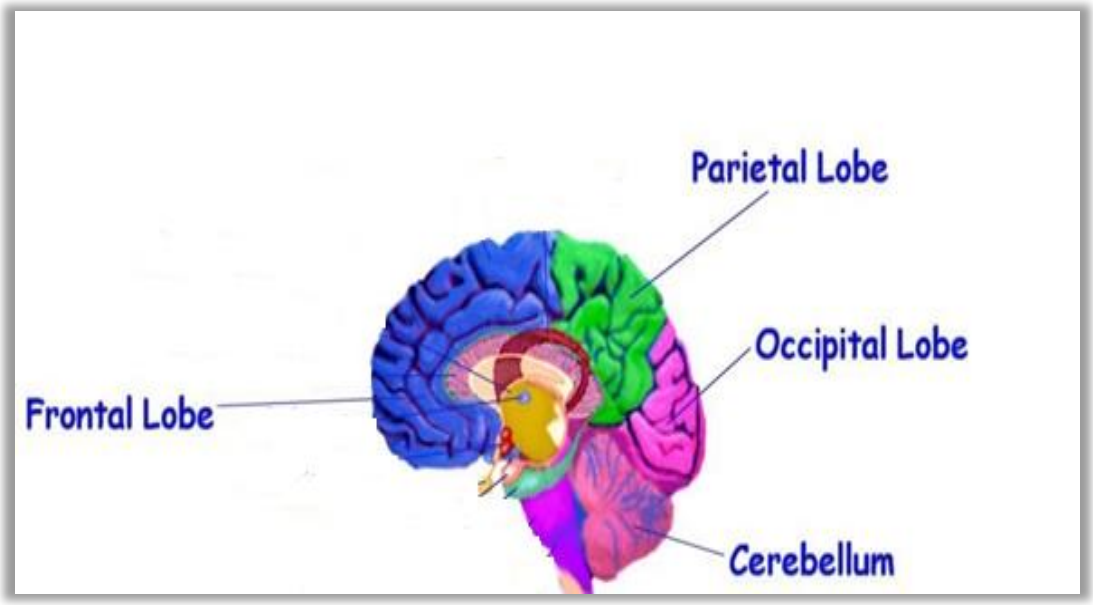



**Figura 22**. Imagen de referencia para la creación del objeto cerebro
Fuente: (Docente, 2015)

**Tabla 7.** Encabezado de la plantilla para la recolección de información de todos los objetos

| DIRECCIÓN DE TECNOLOGIAS PARA LA EDUCACION | | | |
|---|---|---|---|
| **DOCENTE:** | Dra. Ruth Maldonado | **TITULAR:** | |
| **PERIODO:** | Sept 14 – Feb 15 | **AUXILIAR:** | |
| **TITULACION:** | Psicología | | |
| **FECHA:** | 24 de Septiembre del 2014 | | |
| **COMPONENTE:** | Desarrollo de la Inteligencia | **TIPO:** | |

Autor: Michael Freire

**Tabla 8.** *Cuerpo de la plantilla para la creación de información del objeto cerebro*

| OBJETO | | | | | | |
|---|---|---|---|---|---|---|
| **Titulo Unidad** | Cerebro Humano | | | | | |
| | **N° de Unidad en Libro Base** | | | **N° de Unidad es Plan Docente** | | |
| **N°** | **Nombre Modelo** | **Descripción** | | | | |
| **Principal** | Cerebro | Se encuentra ubicado en la cabeza; por lo general, cerca de los principales órganos de los sentidos como la visión, audición, equilibrio, gusto y el olfato. Corresponde, por tanto, al encéfalo de humanos y otros vertebrados y se subdivide en cerebro anterior, medio y posterior | | | | |
| **1** | Lóbulo Parietal | Está ubicado en las partes medias y laterales de la cabeza, los mayores entre los que forman el cráneo. Se trata de la zona cerebral que está encargada especialmente de recibir las sensaciones de tacto, calor, frío, presión, dolor, y coordinar el equilibrio. | | | | |
| **2** | Lóbulo Occipital | Es el centro de nuestro sistema visual de la percepción. Recibe información visual de esta área, desde donde esta información va a otras zonas cerebrales que se especializan en temas como la identificación de palabras. | | | | |
| **3** | Lóbulo Frontal | Es un área de la corteza cerebral de los vertebrados. En los seres humanos está localizado en la parte anterior del cerebro. Los lóbulos temporales están localizados debajo y detrás de los lóbulos frontales. | | | | |
| **4** | Lóbulo Temporal | Se localizada frente al lóbulo occipital, aproximadamente detrás de cada sien, interviene en la retención de memoria visual, el procesamiento de la información sensorial, comprensión del lenguaje, el almacenamiento de nuevos recuerdos y emociones. | | | | |
| **5** | Cerebelo | Es una región del encéfalo cuya función principal es de integrar las vías sensitivas y las vías motoras. Existe una gran cantidad de haces nerviosos que conectan el cerebelo con otras estructuras encefálicas y con la médula espinal. | | | | |

Autor: Michael Freire

### 3.7.2. Objeto 2: Formula coeficiente intelectual.

A continuación se muestra la plantilla donde esta recabada la información del Objeto: Formula Coeficiente Intelectual

$$\text{Coeficiente Intelectual ( CI )} = \frac{\text{Edad mental}}{\text{Edad Cronológica}} \times 100$$

**Figura 23**. Imagen de referencia para la creación del objeto cerebro
Fuente: (Docente, 2015)

**Tabla 9.** Cuerpo de la plantilla para la creación del objeto formula del coeficiente intelectual

| OBJETO | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Titulo Unidad** | Formula Coeficiente Intelectual | | | | | | |
| | **N° de Unidad en Libro Base** | | | **N° de Unidad es Plan Docente** | | | |
| **N°** | **Nombre Modelo** | **Descripción** | | | | | |
| **Principal** | Formula Coeficiente Intelectual | El cociente intelectual (CI), o "coeficiente intelectual", es el número que resume el desempeño de un sujeto al realizar un test en el que se miden sus habilidades cognitivas, su nivel de "inteligencia". Se lo obtiene gracias a la división de la edad intelectual, entre la edad real del individuo y el resultado final multiplicado por 100 o medida numérica de estandarización. | | | | | |
| **1** | Coeficiente Intelectual | El cociente intelectual, o "CI" en forma abreviada, es una puntuación, resultado de alguna de las pruebas estandarizadas diseñadas para valorar la inteligencia. | | | | | |
| **2** | Edad Cronológica | La Edad Cronológica se encuentra definida por la edad natural de la persona | | | | | |
| **3** | Edad Natural | Se la mide a través de instrumentos como test de inteligencia, usualmente se utiliza los:<br>- Test de Raven.- mide la inteligencia general se da principalmente sobre imágenes.<br>- Test de Wesley.- evalúa la parte verbal (numérica, lógica); y la manipulable (rompecabezas, figuras) | | | | | |
| **4** | 100 | El valor constante 100 es la medida numérica estándar para obtener el Coeficiente Intelectual. | | | | | |

Autor: Michael Freire

### 3.7.3. Objeto 3: Helen Adams Keller.

A continuación se muestra la plantilla donde esta recabada la información del Objeto: Helen Adams Keller

**Figura 24.** Imagen de referencia para la creación del objeto Helen Adams Keller
Fuente: (Docente, 2015)

**Tabla 10.** Cuerpo de la plantilla para la creación del objeto Helen Adams Keller

| OBJETO | | | | | | |
|---|---|---|---|---|---|---|
| **Titulo Unidad** | Helen Adams Keller | | | | | |
| | **N° de Unidad en Libro Base** | | | **N° de Unidad es Plan Docente** | | |
| **N°** | **Nombre Modelo** | **Descripción** | | | | |
| **Principal** | Helen Adams Keller | (Alabama, 27 de junio de 1880 - Connecticut, 1 de junio de 1968) fue una escritora, oradora y activista política sordo-ciega estadounidense. A la edad de 19 meses, sufrió una grave enfermedad que le provocó la pérdida total de la visión y la audición.<br>A lo largo de toda su vida, redactó una multiplicidad de artículos y más de una docena de libros sobre sus experiencias y modos de entender la vida, entre ellos "La historia de mi vida (1903) y Luz en mi oscuridad (1927)".<br>Hellen es un ícono en el desarrollo atípico de la inteligencia debido a los logros alcanzados durante su vida pese a sus discapacidades. Dio conferencias en 25 países y viajo alrededor del mundo. Su cociente intelectual era equivalente al 92,85%; en una escala en la cual el 57,14% e inferior, representa inteligencias con evidentes problemas de subdesarrollo normal. El 71,42% representa a la inteligencia normal y el porcentaje igual a un superdotado es igual a 100%. | | | | |

| | | |
|---|---|---|
| **1** | Ojo | Cuando cumplió siete años, sus padres decidieron buscar una instructora y fue así como el Instituto Perkins para Ciegos les envió a una joven especialista en problemas de visión y con mucha experiencia en el trabajo con personas ciegas, Anne Sullivan, que se encargó de su formación y logró un avance en la educación especial centrándose específicamente en su discapacidad visual. Helen continuó viviendo al lado de Sullivan hasta la muerte de esta en 1936. |
| **2** | Oído | Después de graduarse de la escuela secundaria en Cambridge, Keller ingresó en el Radcliffe College, donde recibió una licenciatura, convirtiéndose en la primera persona sordo-ciega en obtener un título universitario. Durante su juventud, comenzó a apoyar al socialismo y en 1905, se unió formalmente al Partido Socialista, pesar de su discapacidad auditiva Helen era una de las activistas más representativas y tenía una de las mayores acogidas. |

Autor: Michael Freire

## 3.8. Creación de Objetos 3D.

Para la creación de los objetos 3D detallados en la presente investigación, se utilizó el software de modelaje, animación, simulación y renderización de 3DS Max Studio con su respectiva licencia académica destinada a profesionales en formación. Mediante dicho software de modelaje 3D se pudo crear y modelar todos y cada uno de los contenidos 3D requeridos por los docentes del componente académico Desarrollo de la Inteligencia.

### 3.8.1. Características de objeto 3D para RA en Metaio.

Para que un objeto 3D sea reconocido adecuadamente por Metaio es necesario que el mismo cumpla con las siguientes características:

- ✓ El modelo del objeto 3D debe estar en formato m2d, obj y fbx, aunque Metaio es capaz de trabajar nativamente con los formatos anteriormente mencionados, es sumamente recomendable exportarlos modelos u objetos 3D al formato mfbx (formato propietario de Metaio) para maximizar el rendimiento de la aplicación.
- ✓ Es recomendable que el tamaño de los objetos una vez exportados a formato mfbx no supere el tamaño de 5 mb, con el fin de maximizar el rendimiento de la aplicación cuando se muestren varios objetos en pantalla.
- ✓ La textura del objeto debe ser trabajada en un archivo por separado en jpg o png y tener uno de las siguientes resoluciones 128x128, 256x256, 512x512, 1024x1024 pixeles; aunque Metaio es capaz de reconocer texturas con mayores resoluciones no

se recomienda sobrepasar los 1024x1024 pixeles. Normalmente se recomienda la utilización de la resolución 512x512 pixeles.

- ✓ La ubicación, rotación y tamaño es recomendable trabajarla a nivel de diseño, aunque es posible en Metaio definir estos parámetros manualmente, es sumamente complicado de realizar, sobre todo los parámetros de ubicación y rotación.

Una vez creado y modelado los objetos 3D y predefinido a nivel de diseño su ubicación, rotación y tamaño se procede a exportarlos mediante la opción “Export” desde el mismo 3DS Max Studio al formato FBX (proprietary file format) ya que no es posible exportarlos al formato mfbx (propiedad de Metaio) directamente. Fbx es un formato de archivo 3D gratuito e independiente de la plataforma, que proporciona acceso al contenido creado en los principales paquete de software de modelado 3D.

Posteriormente se procedió exportar o transformar los objetos 3D de formato fbx al formato mfbx mediante la herramienta proporcionada para este fin, llamada FBXMeshConverter

A continuación se muestra cada objeto 3D creado acorde a las plantillas de recolección de información de los objetos 3D.

### 3.8.2. Objeto 1: Cerebro.


**Figura 25.** Objeto 3D: Cerebro
Autor: Michael Freire

### 3.8.3. Objeto 2: Formula Coeficiente Intelectual.


**Figura 26.** Objeto 3D: Formula del Coeficiente Intelectual
Autor: Michael Freire

### 3.8.4. Objeto 3: Helen Adams Keller.


**Figura 27.** Objeto 3D: Helen Adams Keller
Autor: Michael Freire

### 3.9. Marcador.

LA aplicación utilizara el sistema de seguimiento basado en reconocimiento de imagen, es decir, utilizaremos una tarjeta de RA con el logotipo de la UTPL para la generación de contenidos 3D en la aplicación de RA. La imagen a utilizar es la siguiente:


**Figura 28.** Marcador con el logo de la UTPL
Autor: UTPL

Para su correcto funcionamiento dentro de la aplicación este marcador debe tener un tamaño mínimo de 240x240 pixeles.

### 3.10. Metodología de Desarrollo.

Para el desarrollo de la aplicación de RA se ha tomado como base la metodología de desarrollo Ágil ICONIX, la cual es una metodología que se encuentra en medio camino entre RUP (Rational Unified Process) y XP (eXtreme Programming) haciendo hincapié en la simplicidad de XP y la utilización del modelado UML de RUP para las diversas fases del

proyecto, siendo sus principales características: Iterativo-incremental, permite trazabilidad, y maneja una UML dinámica. (De San Martin Oliva, 2009)

Adicionalmente ICONIX es fácilmente adaptable a pequeños proyectos con pequeños equipos de trabajo donde se prioriza el desarrollo y entrega del producto versus la documentación exhaustiva sobre el mismo.

Durante el desarrollo de la aplicación de RA, y por tratarse de un proyecto pequeño y realizado por una sola persona se simplificó al máximo la documentación, dejando solo aquella documentación realmente trascendental para el proyecto sin perder la esencia de la metodología ICONIX, y los esfuerzos se centraron mayormente en el desarrollo de la aplicación de RA.

A continuación se detalla las diferentes fases de desarrollo de la metodología ICONIX y los diversos entregables obtenidos en cada uno de ellas:

**Análisis de Requisitos**

- ✓ Especificación de requisitos
- ✓ Modelo de dominio
- ✓ Prototipación rápida
- ✓ Modelo de casos de uso

**Análisis y Diseño Preliminar**

- ✓ Descripción de casos de uso

**Diseño**

- ✓ Arquitectura del sistema
- ✓ Diagrama de clase

**Implementación**

- ✓ Escribir y generar código
- ✓ Prototipos
- ✓ Casos de pruebas
- ✓ Bitácora de errores

### 3.11. Análisis de requisitos.

El análisis de requisitos es la parte fundamental de cualquier metodología de desarrollo, siendo especialmente importante la opinión del usuario y los diversos artefactos utilizados para la recolección e interpretación de los mismos, determinando que es lo que le agradaría ver al usuario y así mismo que es lo que no le gustaría ver. Una correcta recolección de requisitos da como resultado sistemas robustos y bien implementados que cumplen a cabalidad las necesidades del usuario, una débil recolección de requisitos origina problemas en las fases superiores del desarrollo y desemboca en sistemas inestables que no cumplen con adecuadamente las necesidades del usuario.

#### 3.11.1. Especificación de Requisitos.

**Tabla 11.** Especificación de Requisito: ER 01 Reconocimiento de marcador

| **Código:** | **ER 01** |
|---|---|
| **Requisito:** | **Reconocimiento de marcador** |
| **Solicitado por:** | Usuario |
| **Prioridad:** | Alta |
| **Descripción:** | La aplicación debe ser capaz de utilizar marcadores o tarjetas de RA basados tanto en patrones geométricos como en imágenes ya sean a color o blanco y negro. |
| **Comentarios:** | Ninguno |

Autor: Michael Freire

**Tabla 12.** Especificación de Requisito: ER 02 Generar contenido de RA

| **Código:** | **ER 02** |
|---|---|
| **Requisito:** | **Generar contenido de RA** |
| **Solicitado por:** | Usuario |
| **Prioridad:** | Alta |
| **Descripción:** | La aplicación debe ser capaz de generar contenido 3D de Realidad Aumentada sobre el marcador o tarjeta RA cuando esta sea escaneado por la cámara del dispositivo. |
| **Comentarios:** | Un contenido 3D puede estar dividido a su vez en varios subobjetos que en un conjunto conforman un todo. |

Autor: Michael Freire

**Tabla 13.** Especificación de Requisito: ER 04 Intercambiar contenidos de RA

| Código: | **ER 03** |
|---|---|
| **Requisito:** | **Intercambiar contenidos de RA** |
| **Solicitado por:** | Usuario |
| **Prioridad:** | Alta |
| **Descripción:** | La aplicación debe ser capaz de poder intercambiar los objetos 3D disponibles en tiempo real mediante un menú de selección de objetos dispuesto para este fin. |
| **Comentarios:** | Ninguno |

Autor: Michael Freire

**Tabla 14.** Especificación de Requisito: ER 04 Interacción mediante clic

| **Código:** | **ER 04** |
|---|---|
| **Requisito:** | **Interacción mediante clic** |
| **Solicitado por:** | Usuario |
| **Prioridad:** | Alta |
| **Descripción:** | La aplicación debe permitir la interacción con los objetos 3D mediante clic o gestos táctiles, permitiendo seleccionar e interactuar con los subobjetos que conforman el objeto principal. |
| **Comentarios:** | Al seleccionar una parte del objeto principal esta debe resaltar sobre las demás partes o subobjetos. |

Autor: Michael Freire

**Tabla 15.** Especificación de Requisito: ER 05 Redimensionar contenidos de RA

| **Código:** | **ER 05** |
|---|---|
| **Requisito:** | **Redimensionar contenidos de RA** |
| **Solicitado por:** | Usuario |
| **Prioridad:** | Media |
| **Descripción:** | LA aplicación debe permitir redimensionar los contenidos 3D visualizados mediante gestos táctiles de zoom in y zoom out. |
| **Comentarios:** | Ninguno |

Autor: Michael Freire

**Tabla 16.** *Especificación de Requisito: ER 06 Rotar contenidos de RA*

| **Código:** | **ER 06** |
|---|---|
| **Requisito:** | **Rotar contenidos de RA** |
| **Solicitado por:** | Usuario |
| **Prioridad:** | Media |
| **Descripción:** | La aplicación debe permitir rotar los contenidos 3D visualizados mediante gestos táctiles o mediante botones dispuestos en la pantalla del dispositivo. |
| **Comentarios:** | Ninguno |

Autor: Michael Freire

**Tabla 17.** Especificación de Requisito: ER 07 Desplegar información relevante

| **Código:** | **ER 07** |
|---|---|
| **Requisito:** | **Desplegar información relevante** |
| **Solicitado por:** | Usuario |
| **Prioridad:** | Media |
| **Descripción:** | La aplicación debe permitir mostrar información relevante sobre el objeto o subobjeto cuando este sea seleccionado por el usuario mediante un cuadro de texto emergente desplegado a una esquina de la pantalla, además debe permitir que se pueda redimensionar el tamaño del texto visualizado. |
| **Comentarios:** | Ninguno |

Autor: Michael Freire

**Tabla 18.** Especificación de Requisito: ER 08 Reproducir Audio

| **Código:** | **ER 08** |
|---|---|
| **Requisito:** | **Reproducir Audio** |
| **Solicitado por:** | Usuario |
| **Prioridad:** | Baja |
| **Descripción:** | La aplicación debe ser capaz de reproducir en audio la información presentada en el cuadro de texto emergente mediante la utilización de un botón adjunto en dicho cuadro de texto. |
| **Comentarios:** | Ninguno |

Autor: Michael Freire

### 3.11.2. Modelo de Dominio.

El modelo de dominio describe de forma rápida los diferentes objetos y las relaciones que tienen entre sí de una forma muy general acorde a los requisitos recolectados.

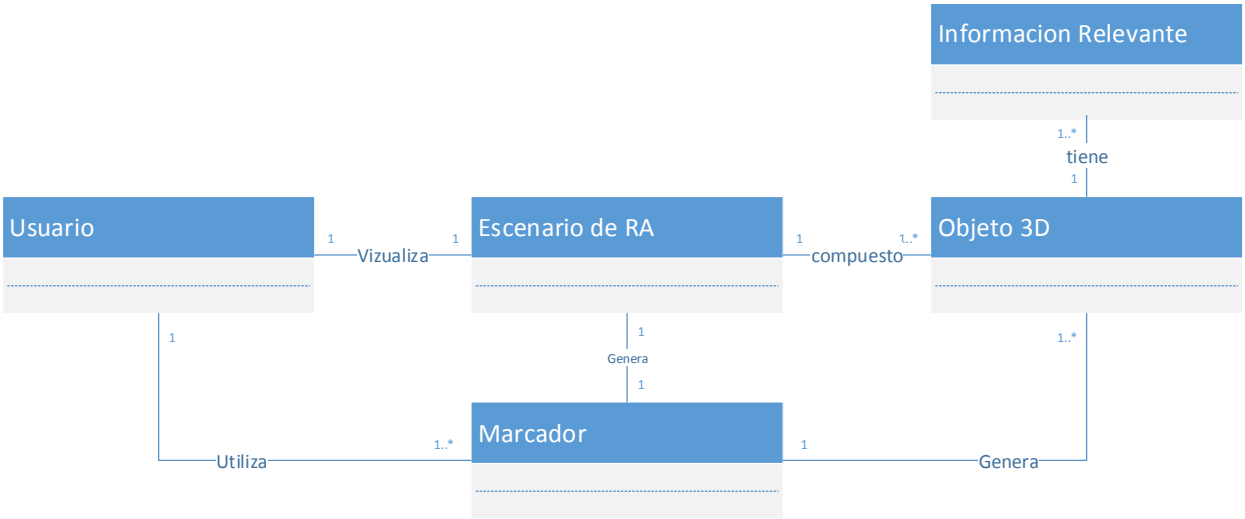



**Figura 29.** Modelo de Dominio
Autor: Michael Freire

### 3.11.3. Prototipación Rápida.

La prototipación rápida del sistema permite brindar al usuario una aproximación de la interfaz que se desarrollaría para el sistema y a su vez permite una mayor comprensión de la aplicación por parte del usuario.

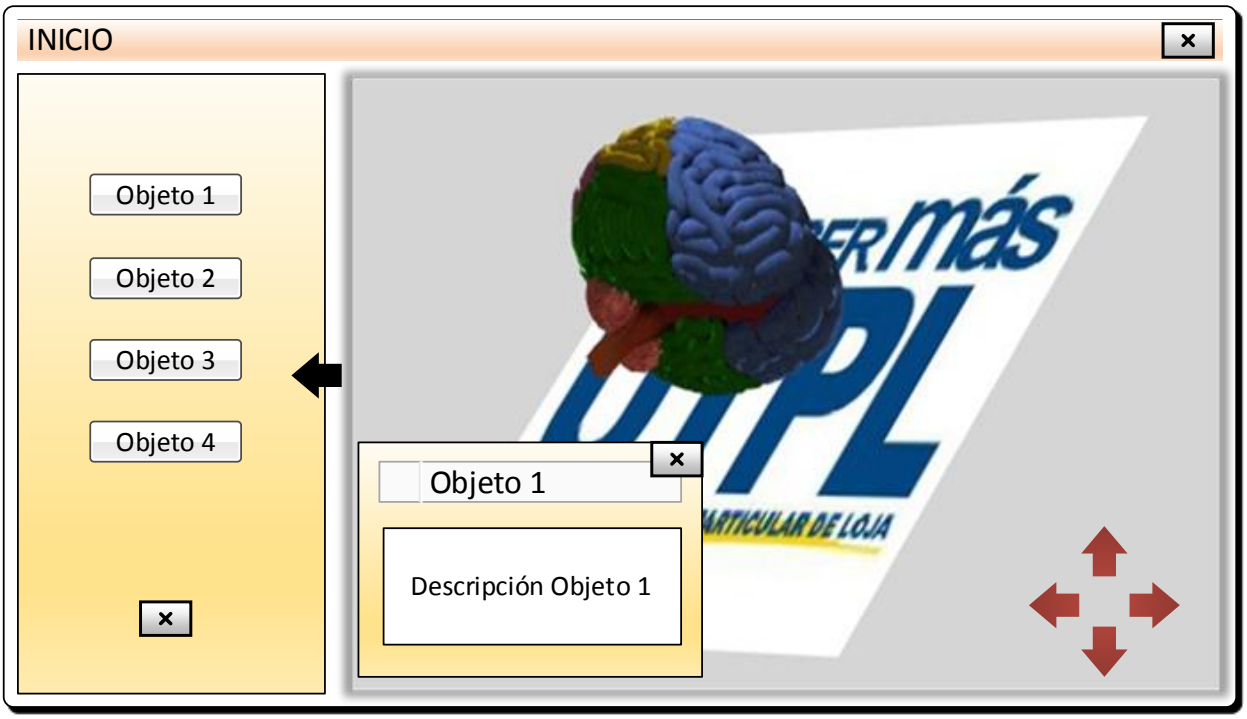



**Figura 30.** Prototipo de interfaz grafica
Autor: Michael Freire

### 3.11.4. Modelo de Casos de Uso.

El modelado de casos de uso permite mostrar una aproximación de alto nivel de las funcionalidades que tendrá el sistema desde el punto de vista del usuario y los actores que intervienen en el mismo.

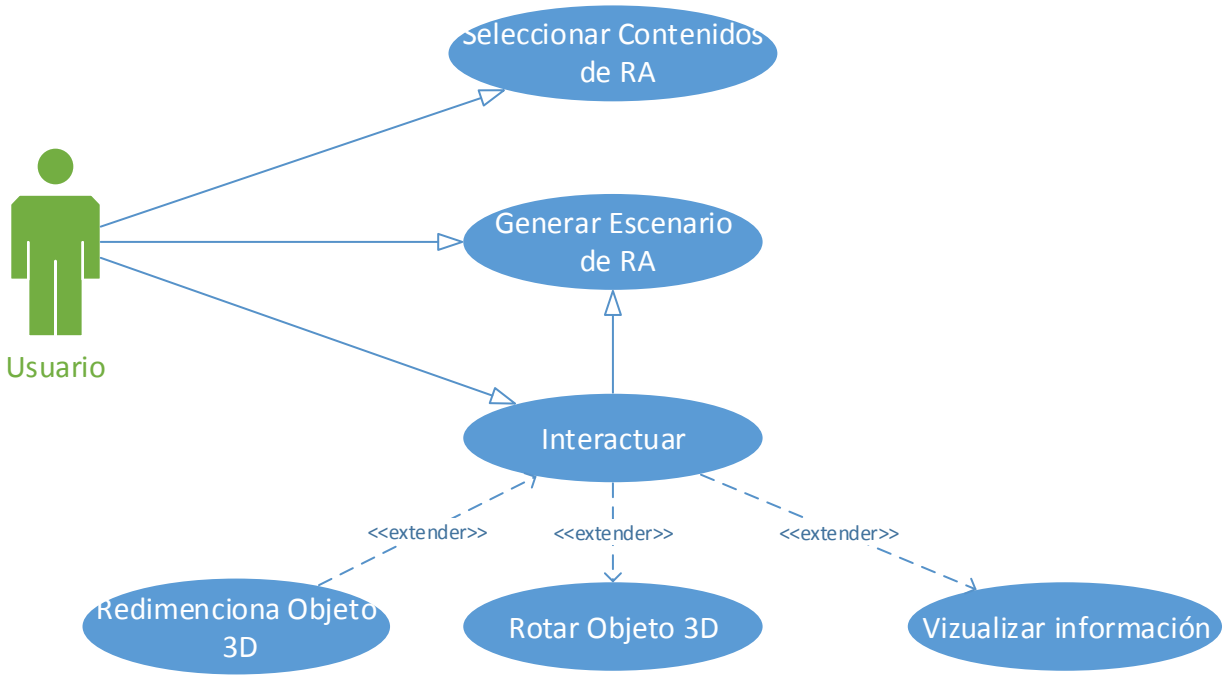



**Figura 31.** Modelo de Casos de Uso
Autor: Michael Freire

### 3.12. Análisis y diseño preliminar.

En esta fase se realizó la descripción de casos de uso acorde al análisis de requerimientos realizados en la fase de anterior de Análisis de Requisitos.

#### 3.12.1. Descripción de Casos de Uso.

La especificación de casos de uso describe en forma de acciones y reacciones el comportamiento de la aplicación desde el punto de vista del usuario.

**Tabla 19.** Caso de Uso: CU 01 Generar Escenario de RA

| **Código:** | **CU 01** |
|---|---|
| **Caso de Uso:** | **Generar Escenario de RA** |
| **Requisitos que satisface:** | Generar contenido de RA, Reconocimiento de marcador |
| **Actor:** | Usuario |
| **Descripción:** | Muestra los objetos 3D de RA dentro de la vista de cámara del dispositivo al enfocar o escanear el marcador |
| **Pre condiciones:** | El usuario debe haber iniciado la aplicación<br>El dispositivo debe ser soportado por la aplicación |
| **Post condiciones:** | Se ha mostrado los objetos 3D de RA |
| **Flujo:** | 1. Iniciar la aplicación<br>2. La aplicación carga los contenidos 3D en memoria<br>a. La aplicación verifica que todos los contenidos se han cargado correctamente (Ex 2.1).<br>3. Se muestra la vista de cámara en espera de escanear un marcador<br>4. El usuario escanea un marcador con la cámara<br>a. Si el marcador corresponde al de la aplicación continua<br>b. Caso contrario la aplicación sigue en espera de que se escanee el marcador correcto.<br>5. Se muestra el escenario de RA dentro de la vista de cámara del dispositivo |
| **Excepciones:** | 2.1 La aplicación verifica los contenidos cargados en memoria.<br>2.2 Si existe un problema muestra el respectivo mensaje de error.<br>2.3 Cierra la aplicación. |

Autor: Michael Freire

**Tabla 20.** *Caso de Uso: CU 02 Seleccionar contenidos de RA*

| **Código:** | **CU 02** |
|---|---|
| **Caso de Uso:** | **Seleccionar contenidos de RA** |
| **Requisito que satisface:** | Intercambiar contenidos de RA |
| **Actor:** | Usuario |
| **Descripción:** | Permite seleccionar los contenidos de RA disponibles |
| **Pre condiciones:** | El usuario debe haber iniciado la aplicación<br><br>El dispositivo debe ser soportado por la aplicación |
| **Post condiciones:** | EL usuario ha seleccionado un nuevo contenido de RA |
| **Flujo:** | 1. Iniciar la aplicación.<br>2. El usuario debe presionar el botón Inicio.<br>3. La aplicación despliega el menú de objetos 3D disponible.<br>4. El usuario debe seleccionar el objeto 3D deseado.<br>5. La aplicación devuelve a su estado inicial el objeto actual.<br>6. Se establece como objeto 3D actual al objeto seleccionado por el usuario. |

Autor: Michael Freire

**Tabla 21.** Caso de Uso: CU 03 Interactuar

| **Código:** | **CU 03** |
|---|---|
| **Caso de Uso:** | **Interactuar** |
| **Requisito que satisface:** | Interactuar mediante clic. |
| **Actor:** | Usuario |
| **Descripción:** | Permite interactuar con los objetos 3D mostrados en pantalla. |
| **Pre condiciones:** | El usuario debe haber iniciado la aplicación.<br><br>El dispositivo debe ser soportado por la aplicación. |
| **Post condiciones:** | EL usuario ha interactuado con el objeto de RA. |
| **Flujo:** | 1. Iniciar la aplicación.<br>2. El usuario debe generar el escenario de RA (Ver CU 01: Generar Escenario de RA).<br>3. El usuario debe mantener enfocado el marcador en todo momento.<br>4. El usuario puede interactuar con los objetos 3D mediante:<br>a. Clic para seleccionar una parte del objeto y visualizar su información (Extends CU 06: Visualizar información).<br>b. Gesto táctil de zoom para redimensionar el objeto (Extends CU 04: Redimensionar Objetos 3D).<br>c. Gesto táctil de compas y botones en pantalla para rotar el objeto (Extends CU 05: Rotar Objetos 3D).<br>5. El usuario ha interactuado con el objeto 3D. |

Autor: Michael Freire

**Tabla 22.** Caso de Uso: CU 04 Redimensionar Objeto 3D

| **Código:** | **CU 04** |
|---|---|
| **Caso de Uso:** | **Redimensionar Objeto 3D** |
| **Requisito que satisface:** | Redimensionar contenidos de RA. |
| **Actor:** | Usuario |
| **Descripción:** | Permite redimensionar los objetos 3D mostrados en pantalla. |
| **Pre condiciones:** | El usuario debe haber iniciado la aplicación.<br><br>El dispositivo debe ser soportado por la aplicación. |
| **Post condiciones:** | El usuario ha redimensionado el objeto de RA |
| **Flujo:** | 1. El usuario debe haber iniciado la aplicación y generado el escenario de RA.<br>2. El usuario mediante gesto táctil de zoom puede redimensionar el objeto.<br>a. Gesto táctil de zoom in para reducir el objeto 3D mostrado en pantalla.<br>b. Gesto táctil de zoom out para reducir el objeto 3D mostrado en pantalla.<br>3. El usuario ha redimensionado el objeto de RA. |

Autor: Michael Freire

**Tabla 23.** Caso de Uso: CU 05 Rotar Objeto 3D

| **Código:** | **CU 05** |
|---|---|
| **Caso de Uso:** | **Rotar Objeto 3D** |
| **Requisito que satisface:** | Rotar contenidos de RA. |
| **Actor:** | Usuario |
| **Descripción:** | Permite rotar los objetos 3D mostrados en pantalla. |
| **Pre condiciones:** | El usuario debe haber iniciado la aplicación.<br><br>El dispositivo debe ser soportado por la aplicación. |
| **Post condiciones:** | El usuario ha rotado el objeto de RA. |
| **Flujo:** | 1. El usuario debe haber iniciado la aplicación y generado el escenario de RA.<br>2. El usuario puede rotar el objeto mediante:<br>a. Gesto táctil de compas hacia izquierda o derecha para rotar el objeto a la posición deseada.<br>b. Dando clic a los botones de rotación dispuestos en pantalla.<br>3. El usuario ha rotado el objeto de RA. |

Autor: Michael Freire

**Tabla 24.** Caso de Uso: CU 06 Visualizar información

| **Código:** | **CU 06** |
|---|---|
| **Caso de Uso:** | **Visualizar Información** |
| **Requisito que satisface:** | Desplegar información relevante, Reproducir audio |
| **Actor:** | Usuario |
| **Descripción:** | Permite desplegar información relevante sobre el objeto 3D mostrado en pantalla. |
| **Pre condiciones:** | El usuario debe haber iniciado la aplicación.<br><br>El dispositivo debe ser soportado por la aplicación. |
| **Post condiciones:** | El usuario visualizado la información el objeto de RA. |
| **Flujo:** | 1. El usuario debe haber iniciado la aplicación y generado el escenario de RA.<br>2. Al seleccionar un objeto la aplicación muestra un cuadro de texto emergente con información relevante.<br>3. Dentro de este cuadro emergente existen botones dispuestos para:<br>a. Botón “A+” y “A-“ para redimensionar el texto en pantalla.<br>b. Botón “Voz” para reproducir en audio el texto mostrado en pantalla.<br>c. Botón “X” para cerrar el cuadro de texto emergente<br>4. El usuario ha visualizado la información del objeto de RA seleccionado. |

Autor: Michael Freire

## 3.13. Diseño.

En la fase de diseño los esfuerzos son centrados en el diseño final del sistema, pudiendo definir una arquitectura para la aplicación y un diagrama de clases que permita tener una vista clara de la estructura del sistema para su desarrollo posterior, sirviendo de guía para la siguiente fase de Implementación.

### 3.13.1. AREL vs Android.

A pesar de que AREL está orientado al desarrollo de aplicaciones de RA cuya principal característica es la independencia de plataforma, sin embargo por ser un leguaje joven, el cual apareció con la versión 5 de Metaio SDK, a la hora de la verdad se encuentra muy limitado en cuanto a funcionalidades se refiere. A continuación se detalla los principales inconvenientes por los que no se seleccionó a AREL como el leguaje de desarrollo de la aplicación de RA:

- ✓ A pesar de que AREL está enfocado como un lenguaje de Desarrollo de Realidad Aumentada multiplataforma, la realidad es que se sigue necesitando trabajar y controlar nativamente en cada plataforma la ejecución y acceso a los recursos utilizados dentro de la aplicación.
- ✓ Así mismo la detección de gestos táctiles no está completamente implementada, teniendo que recurrir al lenguaje nativo de cada plataforma para su correcta implementación y poder obtener el resultado óptimo.
- ✓ El manejo y control de los dispositivos soportado es otro de sus puntos débiles y una vez más es necesario recurrir al lenguaje nativo de cada plataforma si se desea controlar aspectos tales como el tamaño de pantalla soportado, la densidad de pixeles, la versión de sistema operativo mínima, etc.
- ✓ No tiene un motor propio para convertir texto a voz computarizada, teniendo que recurrir a los motores propios de cada plataforma para poder generar dichos contenidos.
- ✓ La facilidad de programación y codificación, teniendo que elaborar grandes rutinas de código para poder realizar ciertas funcionalidades, en comparación a si se desarrollara en lenguaje nativo, o simplemente no se puede realizar determinadas funciones que si está presente en lenguaje nativo de cada plataforma.

No obstante  a pesar de las limitaciones encontradas en AREL sigue siendo una alternativa altamente eficiente para proyectos poco ambiciosos o que no requieran de muchas funcionalidades, se debe tener en cuenta que es la primera versión de AREL por lo que aún le queda mucho camino por recorrer para convertirse en un referente en cuanto al desarrollo de aplicaciones de RA refiere.

### 3.13.2. Arquitectura.


**Figura 32.** Arquitectura de la aplicación
Autor: Arquitectura propuesta por (Metaio, Metaio web site, 2014)

A continuación se detallan cada una de las capas presentes en la arquitectura de la aplicación de RA propuesta por Metaio SDK mediante la utilización del leguaje nativo de Android:

- ✓ **Capa de Aplicación:** Es la encargada de generar la interfaz gráfica de usuario y mostrar el escenario de RA, además recolecta toda la interacción que existe entre el usuario y la aplicación de RA y a su vez pasa toda la información recolectada a la Capa de Android SDK.
- ✓ **Capa de Android SDK Java:** Esta capa contiene todo el conjunto de librerías necesarias para procesar la información generada por la capa de aplicación y sirve de nexo entre la capa de Aplicación y la capa Metaio SDK, pasando todas las instrucciones de RA que no pueden ser procesadas en esta capa.
- ✓ **Capa de Metaio SDK:** Es el núcleo principal de la aplicación de RA, es la encargada del procesamiento del escenario de RA, y mediante la utilización de sensores y sistemas de seguimiento, renderizado y captura es capaz de generar el escenario de RA en el dispositivo.
- ✓ **Capa de SO Android:** El sistema operativo base sobre el cual se ejecuta la aplicación de RA contiene todos los servicios y funciones necesarias para interpretar y ejecutar las instrucciones de código generadas por las capas superiores a más bajo nivel.

### 3.13.3. Diagrama de Clases.

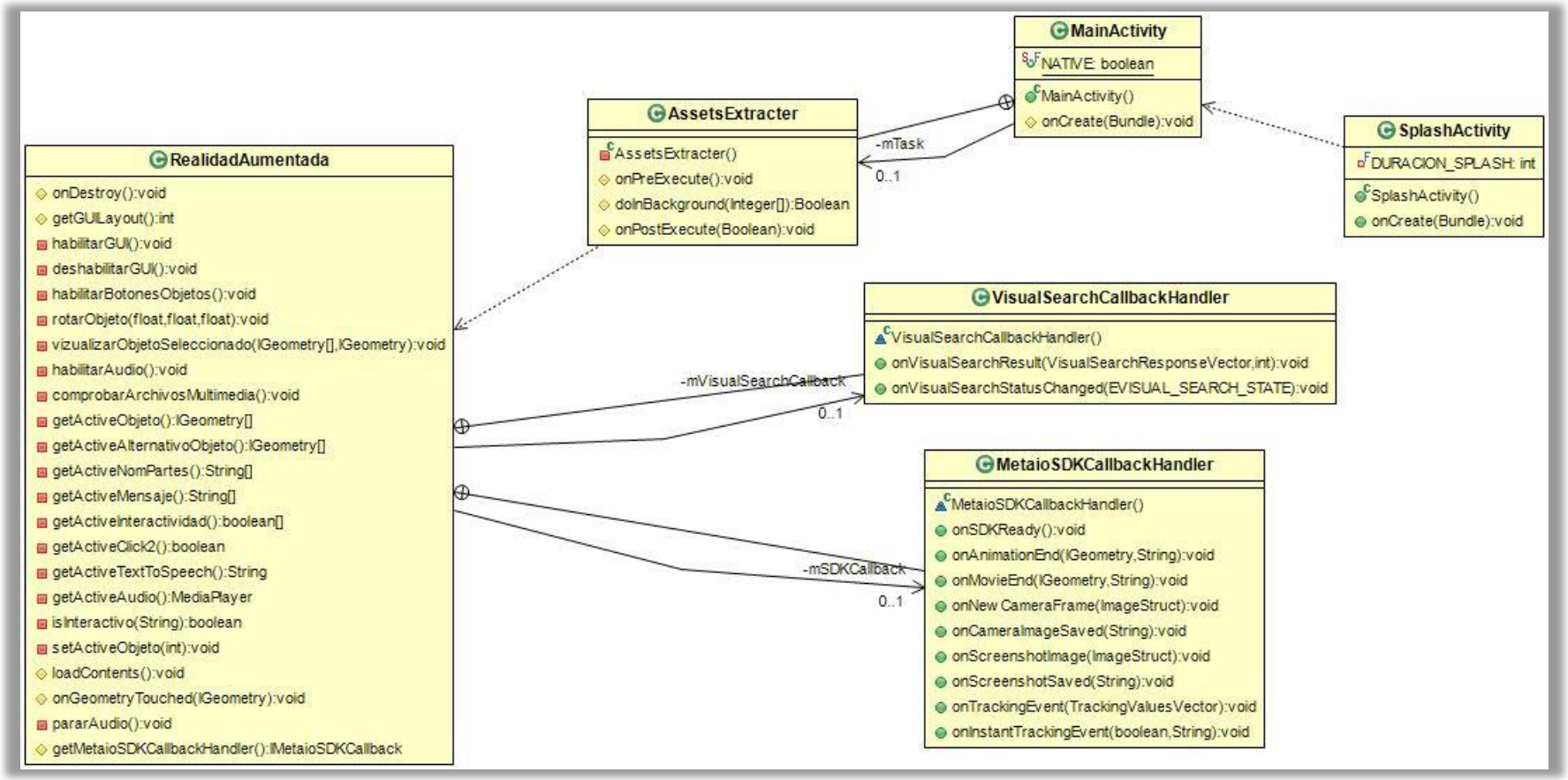



**Figura 33.** Diagrama de Clases
Autor: Michael Freire

## 3.14. Implementación.

La última fase de implementación se basa única y exclusivamente al desarrollo de la aplicación en base a la arquitectura propuesta en la fase anterior, llevando un historial de prototipos, casos de pruebas y bitácora de errores presentados a lo largo del proceso de desarrollo.

### 3.14.1. RA-Desarrollo-Inteligencia.

El nombre seleccionado para la aplicación de RA es RA-Desarrollo-Inteligencia el cual hace referencia a la asignatura Desarrollo de la Inteligencia de la cual han sido extraídos los contenidos 3D utilizados en la presente aplicación, cuyo principal objetivo es servir como herramienta de apoyo al aprendizaje de los estudiantes de la modalidad abierta y a distancia de la UTPL mediante la utilización de objetos 3D interactivos de RA.

#### 3.14.1.1. Interfaz Grafica.

La aplicación de RA-Desarrollo-Inteligencia consta de una interfaz gráfica con una ventana principal, dentro de la cual se encuentran todos los controles necesarios para su ejecución e interacción.

**Ventana Principal**

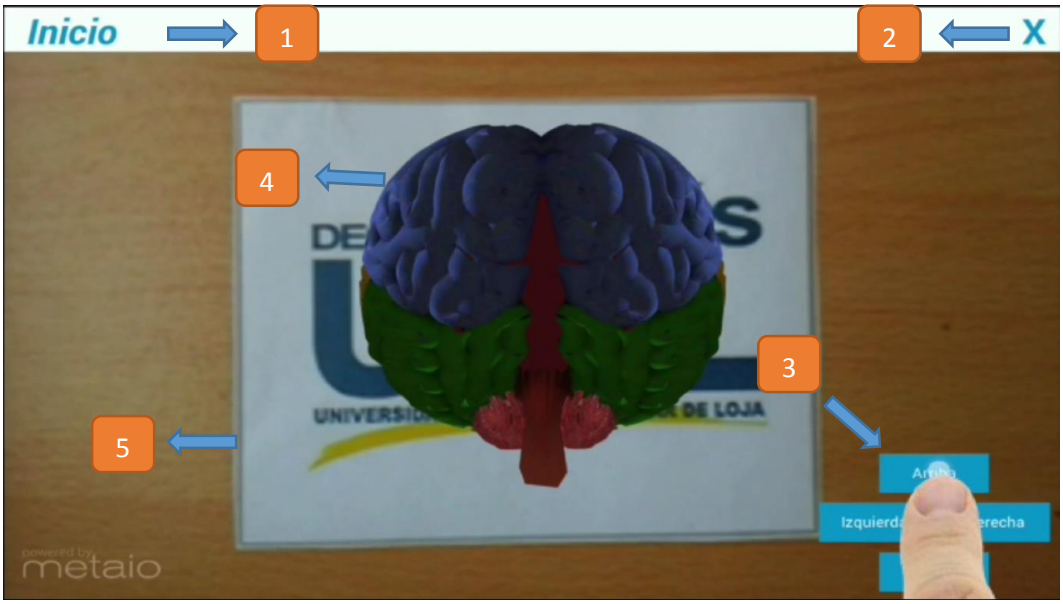



**Figura 34.** Interfaz de la aplicación
Autor: Michael Freire

1. Botón de Inicio, al presionarlo se desplegara el conjunto de objetos disponibles para la asignatura Desarrollo de la Inteligencia.
2. Botón X, permite cerrar la aplicación.
3. Botones de rotación, permite rotar el objeto 3D mostrado en pantalla.
4. Objeto 3D de RA.
5. Marcador o tarjeta RA de la aplicación.

### Menú Inicio

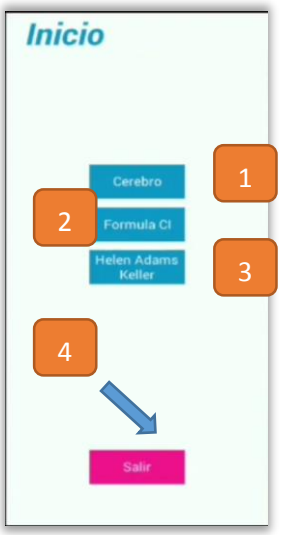



**Figura 35.** Interfaz del menú Inicio
Autor: Michael Freire

1. Botón cerebro, permite cambiar el objeto 3D actual por el objeto 3D cerebro.
2. Botón Formula CI, permite cambiar el objeto 3D actual por el objeto 3D Formula CI.
3. Botón Helen Adams Keller, permite cambiar el objeto 3D actual por el objeto 3D Helen Adams Keller.
4. Botón salir, permite salir de la aplicación.

### Cuadro de Mensaje Desplegable

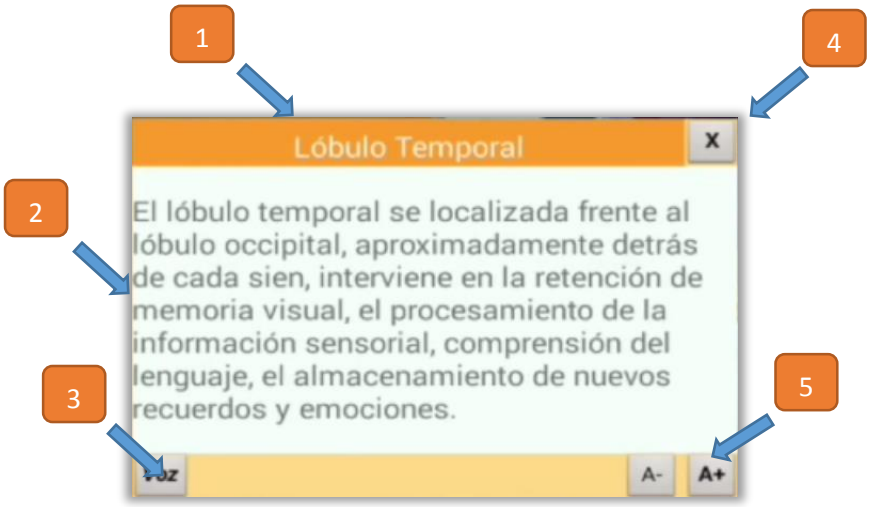



**Figura 36.** Cuadro de texto desplegable
Autor: Michael Freire

1. Nombre del Objeto seleccionado.
2. Mensaje o concepto del objeto seleccionado.
3. Botón Voz, que permite reproducir el audio correspondiente a dicho objeto.
4. Botón X, permite cerrar el cuadro de mensaje desplegable.
5. Botón A- A+, permiten aumentar o disminuir el tamaño de letra del mensaje desplegable.

#### 3.14.1.2. Funcionalidades.

Adicionalmente a los botones dispuestos en pantalla para la interacción, la aplicación de Ra-Desarrollo-Inteligencia permite mediante la utilización de gestos táctiles la interacción con los objetos 3D mostrados en pantalla.

1. Se puede Interactuar con los objetos 3D dando clic sobre estos, lo que provocará que aparezca en pantalla el cuadro de mensaje desplegable con información útil sobre el mismo.

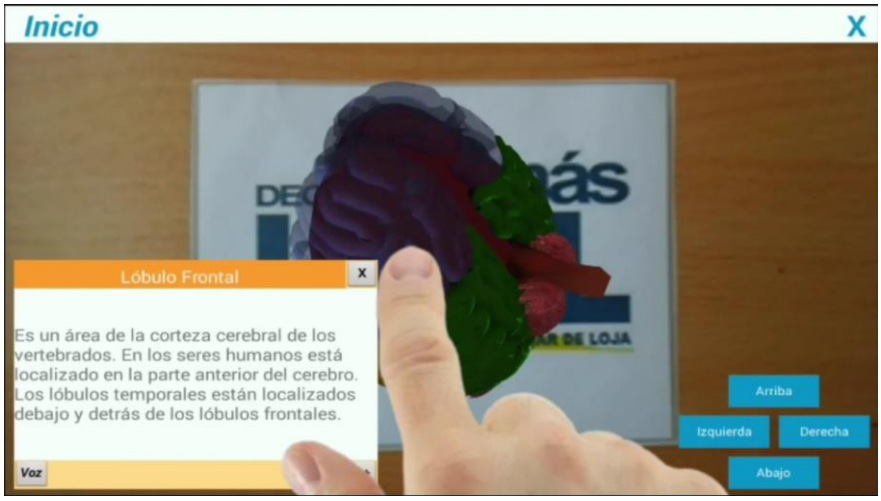



**Figura 37.** Gestos táctil selección
Autor: Michael Freire

2. Se puede hacer Zoom+ y Zoom- con el gesto de pellizco.

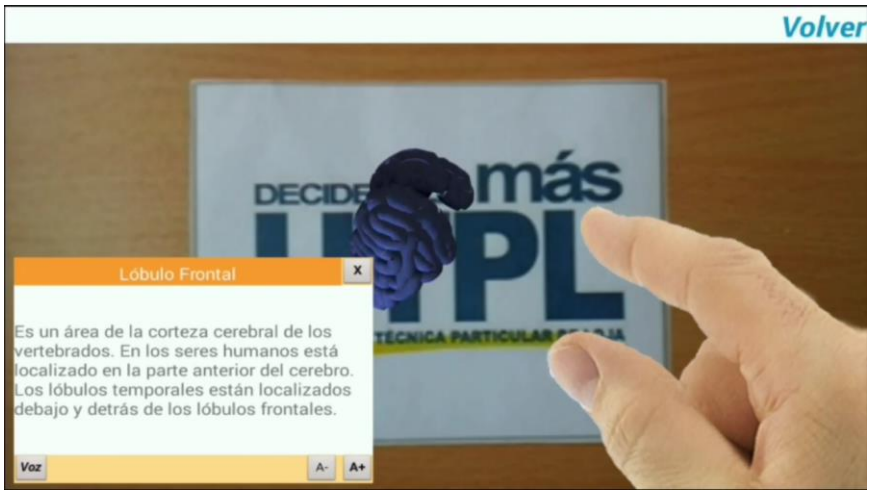



**Figura 38.** Gestos táctil Zoom
Autor: Michael Freire

3. Adicionalmente también se puede girar el objeto 3D con el gesto de compás.


**Figura 39.** Gestos táctil compas (rotación)
Autor: Michael Freire

### 3.14.2. Prototipos.

A lo largo del desarrollo de la aplicación se presentaron varios prototipos que satisfacían los diferentes requisitos recolectados en la fase de análisis de requisitos. A continuación se detalla de forma breve las características implementadas en cada prototipo:

**Prototipo 1:** En el primer prototipo de la aplicación únicamente se puso en funcionamiento lo que era la generación del escenario de Realidad Aumentada con objetos 3D sin ninguna textura presente, es decir la aplicación únicamente cargaba un único objeto sobre el marcador designado sin ningún otro tipo de funcionalidad.

**Prototipo 2:** En el segundo prototipo de la aplicación se integró lo que era la fragmentación de un escenario de RA, es decir se cargaron varios subobjetos los cuales en un conjunto formaban un objeto principal pudiendo seleccionar cada una de las partes, añadiendo un efecto de transparencia al seleccionar dicha parte, y si se volvía a seleccionar la parte resaltada mediante la transparencia esta se mostraba en una vista individual ocultando las demás partes del objeto.

**Prototipo 3:** En el tercer prototipo se trabajó lo que era la implementación de los gestos táctiles para redimensionar el objeto y se dispuso botones y gestos táctiles para la rotación del objeto, además de presentar el cuadro de texto emergente al seleccionar una parte de un subobjeto, y se hicieron avances en lo concernientes a las texturas de los objetos que no eran correctamente cargadas por errores en el diseño de los mismo y, no así, por el SDK de Metaio.

**Prototipo 4:** Se dispuso una barra de menú en la parte superior con dos botones, el botón inicio permite desplegar un menú mediante el cual se puede intercambiar los objetos visualizados en pantalla en tiempo real, mientras que un botón cerrar el cual permitía cerrar la aplicación, además de permitir reproducir el mensaje desplegable en audio ya sea cargado en la aplicación o generado por automáticamente.

**Prototipo 5:** En el último prototipo se trabajó en la detección y corrección de errores con el fin de que la aplicación fuera más estable y subsanar los posibles errores de ejecución que pudieran presentarse. Además de restablecer los objetos al estado inicial al ser intercambiados, y adicionalmente se dieron unos pequeños retoques en la interfaz.

### 3.14.3. Casos de Prueba.

Los casos de prueba se basan en los diversos casos de usos elaborados en las fases anteriores y su principal objetivo es de comprobar el correcto funcionamiento de la aplicación al realizar las acciones descritas en dichos casos de uso.

**Tabla 25.** Caso de Prueba: CP 01 Generar Escenario de RA

| **Código:** | | **CP 01** | |
|---|---|---|---|
| **Caso de Prueba:** | | CU 01: Generar Escenario de RA | |
| **Realizado por:** | | Usuario | |
| **Descripción** | | Comprobar el servicio de generación de Objetos 3D de RA. | |
| **Paso** | **Acción** | **Respuesta esperada del sistema** | **Resp.** |
| **1** | Iniciar Aplicación. | Carga de los servicios de Metaio SDK. | OK |
| **2** | Cargar objetos 3D en memoria. | Si no se cargan todos los contenidos presentar mensaje de error, caso contrario continuar. | OK |
| **3** | Cargar la vista de cámara. | Inicializa la vista de cámara del dispositivo. | OK |
| **4** | Enfocar un marcador. | Si el marcador es el correcto continuar, caso contrario seguir en espera del marcador correcto. | OK |
| **5** | Cargar objeto 3D. | Se carga el objeto 3D de RA en la vista de cámara sobre el marcador escaneado. | OK |

Autor: Michael Freire

**Tabla 26.** Caso de Prueba: CP 02 Seleccionar Contenidos de RA

| **Código:** | | **CP 02** | |
|---|---|---|---|
| **Caso de Prueba:** | | CU 02: Seleccionar Contenidos de RA. | |
| **Realizado por:** | | Usuario | |
| **Descripción** | | Comprobar la funcionalidad del cambio de los objetos 3D en pantalla. | |
| **Paso** | **Acción** | **Respuesta esperada del sistema** | **Resp.** |
| **1** | Presionar botón Inicio. | Se despliega el menú de selección de objetos 3D. | OK |
| **2** | Clic sobre un objetos disponible del menú. | El sistema reinicializa el objeto actual a su estado inicial. | OK |
| **3** | Cambio de objeto 3D. | El sistema cambia el Objeto 3D actual por el objeto seleccionado. | OK |
| **5** | Cargar objeto 3D. | Se carga el nuevo objeto 3D de RA en la vista de cámara sobre el marcador escaneado. | OK |

Autor: Michael Freire

**Tabla 27.** Caso de Prueba: CP 03 Interactuar

| **Código:** | | **CP 03** | |
|---|---|---|---|
| **Caso de Prueba:** | | CU 03: Interactuar | |
| **Realizado por:** | | Usuario | |
| **Descripción** | | Comprobar la interacción de los objetos. | |
| **Paso** | **Acción** | **Respuesta esperada del sistema** | **Resp.** |
| **1** | Dar un clic sobre una parte del Objeto 3D. | El sistema detecta el clic sobre el objeto. | OK |
| **2** | Resaltar la parte seleccionada. | El sistema resalta mediante la transparencia la parte seleccionado. | OK |
| **3** | Cambio de parte seleccionada. | El sistema vuelve la parte actualmente seleccionada a su estado inicial. | OK |
| **5** | Resaltar nueva parte seleccionada. | El sistema transparenta la nueva parte seleccionada. | OK |

Autor: Michael Freire

**Tabla 28.** Caso de Prueba: CP 04 Redimensionar Objeto 3D

| **Código:** | | **CP 04** | |
|---|---|---|---|
| **Caso de Prueba:** | | CU 04: Redimensionar Objeto 3D | |
| **Realizado por:** | | Usuario | |
| **Descripción:** | | Comprobar el correcto funcionamiento de los gestos táctiles para redimensionar los contenidos de RA | |
| **Paso** | **Acción** | **Respuesta esperada del sistema** | **Resp.** |
| **1** | Realizar el Gesto táctil de zoom out. | El sistema detecta el gesto táctil realizado por el usuario. | OK |
| **2** | Redimensionar objeto 3D. | El sistema incrementa el tamaño del objeto 3D en tiempo real. | OK |
| **3** | Realizar el Gesto táctil de zoom in. | El sistema detecta el gesto táctil realizado por el usuario. | OK |
| **4** | Redimensionar objeto 3D. | El sistema disminuye el tamaño del objeto 3D en tiempo real. | OK |

Autor: Michael Freire

**Tabla 29.** Caso de Prueba: CP 05 Rotar Objeto 3D

| **Código:** | | **CP 05** | |
|---|---|---|---|
| **Caso de Prueba:** | | CU 05: Rotar Objeto 3D | |
| **Realizado por:** | | Usuario | |
| **Descripción:** | | Comprobar el correcto funcionamiento de los gestos táctiles y botones dispuestos para rotar los contenidos de RA | |
| **Paso** | **Acción** | **Respuesta esperada del sistema** | **Resp.** |
| **1** | Realizar el Gesto táctil de compas hacia la derecha. | El sistema detecta el gesto táctil realizado por el usuario. | OK |
| **2** | Rotar objeto 3D. | El sistema gira hacia la derecha el objeto 3D en tiempo real. | OK |
| **3** | Realizar el Gesto táctil de compas hacia la izquierda. | El sistema detecta el gesto táctil realizado por el usuario. | OK |
| **4** | Rotar objeto 3D. | El sistema gira hacia la izquierda el objeto 3D en tiempo real. | OK |
| **5** | Se presiona los botones de rotación. | El sistema detecta la pulsación realizado por el usuario. | OK |
| **6** | Rotar objeto 3D. | El sistema gira el objeto 3D hacia la posición seleccionada por el usuario en tiempo real. | OK |

Autor: Michael Freire

**Tabla 30.** Caso de Prueba: CP 06 Visualizar Información

| **Código:** | | **CP 06** | |
|---|---|---|---|
| **Caso de Prueba:** | | CU 06: Visualizar Información | |
| **Realizado por:** | | Usuario | |
| **Descripción:** | | Comprobar el correcto funcionamiento del cuadro de texto emergente con información relevante del objeto | |
| **Paso** | **Acción** | **Respuesta esperada del sistema** | **Resp.** |
| **1** | Se realiza clic sobre una parte del objeto. | El sistema detecta el clic realizado por el usuario. | OK |
| **2** | Despliegue de cuadro de texto. | El sistema despliega el cuadro de texto con la información del objeto seleccionado. | OK |
| **3** | Se presiona los botones para aumentar o disminuir el tamaño de letra (A+ y A-). | El sistema detecta la pulsación realizado por el usuario. | OK |
| **4** | Redimensiona tamaño de letra. | El sistema redimensiona el tamaño de letra de la información presentada. | OK |
| **5** | Se presiona los botones para escuchar el texto (Voz). | El sistema detecta la pulsación realizado por el usuario. | OK |
| **6** | Reproducir Audio. | El sistema reproduce el audio asignado a la parte del objeto seleccionada. | OK |

Autor: Michael Freire

### 3.14.4. Bitácora de errores.

En la bitácora de errores se describen los principales errores encontrados en el desarrollo de la aplicación, las cuales sirven de ayuda a los desarrolladores al mantener un registro de cambios realizados a causa de los mismos además de fuente de consulta para futura referencia en proyectos similares

**Tabla 31.** Bitácora de Errores

| Num. | Tipo | Descripción | Solución |
|---|---|---|---|
| **1** | Diseño de Objeto 3D | Los objetos se cargaban siempre en la posición (0.0.0) y no en la posición requerida. | Tolos los errores encontrados en la carga de los objetos 3D fueron tratados a nivel de diseño pudiendo establecer las posiciones y tamaños de los mismos dentro del propio objeto 3D. |
| **2** | Diseño de Objeto 3D | Los objetos se cargaban con diferentes proporciones y tamaños. | |
| **3** | Diseño de Objeto 3D | Las texturas de los objetos no se visualizaban adecuadamente. | Las texturas de los objetos 3D se las trabajó por separado a nivel de diseño acorde a los requerimientos de Metaio SDK. |
| **4** | Codificación | El sistema no restablecía la parte del objeto seleccionada al dar clic en una nueva parte o subobjeto. | Se guardó en una variable el objeto actual y el nuevo objeto seleccionado a fin de poder realizar comprobaciones y evitar este error. |
| **5** | Codificación | La aplicación se colgaba al intentar reproducir el audio de un objeto si este no tenía asignado un audio personalizado. | Se realiza una comprobación previa y en caso de no existir un audio asignado la aplicación utiliza el motor de texto a audio de Android para genera este contenido. |

| | | | |
|---|---|---|---|
| **6** | Codificación | Al parar una reproducción de audio y volverla a reanudar la aplicación dejaba de responder. | Se realizó comprobaciones adicionales para evitar que el reproductor quede en estado null después de parar o pausar una reproducción. |
| **7** | Codificación | Al intercambiar los objetos visualizados estos no se inicializaban adecuadamente. | Se corrigió el error restableciendo primero todas las propiedades del objeto actual a su estado inicial antes de un intercambio. |
| **8** | Codificación | Al iniciar la aplicación está se colgaba cuando no podía cargar adecuadamente los objetos 3D en memoria. | Antes de inicializar completamente la aplicación esta comprueba que todos los recursos se hayan cargado adecuadamente, caso contario presenta un mensaje indicando el error. |
| **9** | Codificación | La aplicación no detectaba adecuadamente el marcador de la aplicación bajo ciertas condiciones. | Se ajustó el nivel de comprobación de marcador para que sea más permisivo en cuanto a la igualdad del marcador. |
| **10** | Codificación | Al cerrar la aplicación y volverla a iniciar esta tendía a colgarse en ciertos dispositivos. | Se realizaron comprobaciones adicionales antes de cerrar la aplicación con el fin de liberar los recursos utilizados para que al volver a iniciar la aplicación no existan conflictos. |

Autor: Michael Freire

## CONCLUSIONES

- Metaio SDK se destaca de entre los diversos SDK de desarrollo de Aplicaciones de RA por su gran robustez, permitiendo al usuario trabajar en diversas plataformas como: Android, IOS, PC, etc. conjuntamente con los principales tipos de reconocimiento (tracking) existentes.
- Google actualmente proporciona un entorno de desarrollo IDE “Android Studio” para la creación de aplicaciones en Android, sin embargo es posible seguir utilizando otros entornos de desarrollo como es el caso Eclipse, siempre y cuando se instalen los plugin correspondientes
- El lenguaje de programación de RA AREL, a pesar de ofrecer la posibilidad de desarrollo de aplicaciones multiplataforma, aún le queda mucho margen de mejora en cuanto a funcionalidades disponibles, ya que se queda limitado si se lo compara con soluciones trabajadas en lenguaje nativo de cada plataforma.
- La RA como tecnología aplicada a la educación puede ser de gran ayuda a estudiantes que de una u otra forma se encuentran limitados en cuanto a materiales didácticos disponibles en su medio, pudiendo optar por representaciones 3D de RA de contenidos no disponibles en su medio.
- Cuando se desarrolla una aplicación de RA, se debe definir hacia que dispositivos va dirigida, y se debe prestar vital importancia hacia el tamaño de pantalla de dicho dispositivo, ya que en dispositivos dotado con pantallas muy pequeñas es difícil apreciar e interactuar con contenidos de RA.
- Al utilizar marcadores basados en el reconocimiento de imagen se requiere que estas tengan un tamaño y contraste considerable para facilitar su reconocimiento, adicionalmente al utilizar varios marcadores dentro de una misma aplicación es recomendable que estos sean claramente diferenciables entre sí.

## RECOMENDACIONES

- Se recomienda trabajar con el entorno de desarrollo de Eclipse principalmente para aquellos programadores que están dando sus primeros pasos en el desarrollo de aplicaciones para la plataforma de Android, especialmente por su adaptabilidad y comodidad de uso al trabajar en un mismo entorno de desarrollo tanto aplicaciones de escritorio como aplicaciones móviles.
- Al abordar proyectos de RA de gran envergadura se recomienda la utilización una solución basada en el leguaje nativo de la plataforma seleccionada, permitiendo mayor robustez en la solución. Y para proyectos pequeños que no requieran grandes funcionalidades como manejo de archivos, detección de gestos táctiles o motores de audio, etc.; se recomienda utilizar el lenguaje AREL, el cual facilita su portabilidad a varias plataformas.
- Al trabajar con objetos 3D dentro de aplicaciones de RA es recomendable que los temas a modelar sean de granularidad alta, ya que un modelo demasiado detallado tiene un mayor tamaño, y por ende, repercute en el rendimiento de la aplicación.
- Al trabajar con aplicaciones de RA es recomendable enfocar la aplicación a dispositivos con un tamaño de pantalla considerable con el fin de facilitar la inmersión del usuario.
- Para la creación de marcadores se recomienda la utilización de la suite de imagen y fotografía de adobe con el fin de crear marcadores de alta calidad para que sean fácilmente reconocibles por la aplicación de RA.
- A pesar de que Metaio SDK soporta algunos formatos de objetos 3D, es recomendable transformar dichos objetos al formato mfbx, el cual, al ser un formato propio de Metaio se encuentra optimizado para su funcionamiento con Metaio SDK

# REFERENCIAS BIBLIOGRÁFICAS


Apple. (2014). *Apple Developer web site*. Obtenido de https://developer.apple.com/xcode/downloads/

Aumentaty. (2014). *El valor de la Realidad Aumentada.* Obtenido de http://www.aumentaty.com/

Basogain, X., Olabe, M., Espinoza, K., Roueche, C., & Olabe, J. (2007). *Realidad Aumentada en la Educación: una tecnología emergente.* Obtenido de Escuela Superior de Ingeniería de Bilbao: http://www.anobium.es/docs/gc_fichas/doc/6CFJNSalrt.pdf

Castle, R. (2009). *Active Vision Laboratory*. Obtenido de http://www.robots.ox.ac.uk/~bob/research/research_objectslam.html

De San Martin Oliva, C. R. (2009). *Portal Huarpe.* Obtenido de http://www.portalhuarpe.com.ar/Seminario09/archivos/MetodologiaICONIX.pdf

Eclipse. (2014). *Eclipse web site*. Obtenido de http://www.eclipse.org/

Edwards, G. (s/a). *University of Manchester*. Obtenido de http://www.lysator.liu.se/~eru/research/

Eventos Realidad Aumentada . (2012). *Eventia Turismo y Publicidad*. Obtenido de http://grupoeventia.files.wordpress.com/2010/06/realidadaumentada.png

Fundación Telefonica. (2011). *Internet, Realidad Aumentada una nueva lente para ver el mundo.* Obtenido de http://www.fundacion.telefonica.com/es/que_hacemos/media/publicaciones/Realidad_Aumentada_Completo.pdf

Google. (2014). *Android Developer web site*. Obtenido de http://developer.android.com/index.html

Google. (2014). *Google Glass web site*. Obtenido de https://www.google.com/glass/start/

Google. (2014). *Google Sky Map web site*. Obtenido de http://www.google.com/sky/about.html

Greiner, M. (2013). *Augmented Realty by Metaio.* Obtenido de http://www.hs-augsburg.de/~john/mobile-experience/workshops/Metaio_AR/metaio.pdf

Honig, Z. (2013). *Engadget*. Obtenido de http://www.engadget.com/2013/10/09/avegant-retinal-hmd/

LeanAR. (2014). *LeanAR web site*. Obtenido de http://www.learnar.org/

Loup, A. (2012). *Realidad Aumentada.* Obtenido de Universidad Católica "Nuestra Señora de Asunción": http://dragodsm.com.ar/pdf/dragodsm-temas-de-interes-reconocimiento-realidad-aumentada-09-2012.pdf

Lozano, N. (2013). *Comunidad IEBS* . Obtenido de http://comunidad.iebschool.com/mividasinmovil/2013/09/29/los-dispositivos-moviles-soporte-de-vida/

Metaio. (2014). *Metaio Developer Portal*. Obtenido de https://dev.metaio.com/arel/overview/

Metaio. (2014). *Metaio web site*. Obtenido de http://www.metaio.com

Microsoft. (2014). *Visual Studio web site*. Obtenido de http://www.visualstudio.com/

Nintendo. (2014). *Nintendo web site*. Obtenido de http://www.nintendo.com/consumer/systems/3ds/es_la/ar_download.jsp

Ockendon. (2014). *Ockendon web site*. Obtenido de http://www.ockendon.net/halluxangles_homepage.htm

Olwal, A. (2010). *An Introduction to Augmented Reality.* Obtenido de KTH - Department of Numerical Analysis and Computer Science: http://www.csc.kth.se/~alx/courses/DT2140/olwal_introduction_to_ar_2010_03_05.pdf

Ortiz, F. (2014). *Un Mundo Aumentado*. Obtenido de http://unmundoaumentado.com/2014/01/07/metaio-va-a-empezar-a-integrar-realidad-aumentada-en-3d-en-el-sdk-de-intel-real-sense/

Quest Visual. (2014). *Word Lents web site*. Obtenido de http://questvisual.com/

Renukdas, P., Ghundiyal, R., Gadgil, H., & Pathare, V. (2013). *Markerless Augmented Reality Android App for Interior Decoration*. Obtenido de Department of Computer Engineering, MES College of Engineering.: http://www.ijngca.com/Papers/IJNGCA18032013.pdf

Silva, R., Oliveira, J., & Giraldi, G. (2003). *Introduction to Augmented Reality.* Obtenido de National Laboratory for Scientific Computation: http://www.lncc.br/~jauvane/papers/RelatorioTecnicoLNCC-2503.pdf

Sony. (2012). *Playstation Blog*. Obtenido de http://blog.latam.playstation.com/2012/02/23/tarjetas-de-realidad-aumentada-para-tu-ps-vita/

Unity. (2014). *Unity web site*. Obtenido de http://unity3d.com

Vuforia. (2014). *Vuforia web site.* Obtenido de https://www.vuforia.com/

Wikitude. (2014). *Wikitude web site*. Obtenido de http://www.wikitude.com/

Zeis, A. (2014). *Android Central*. Obtenido de http://www.androidcentral.com/epson-moverio-bt-200-smart-glasses-now-available

**ANEXOS**

# MANUAL DEL PROGRAMADOR

## RA-Desarrollo-Inteligencia


### Descripción breve

El siguiente manual tiene la finalidad de guiar al programador a través de las principales rutinas de código empleadas en el desarrollo de la aplicación de RA



Michael Freire

mfffreire@utpl.edu.ec

**Introducción.**

El presente Manual de Programador tiene la finalidad de guiar al programador o administrador a través del flujo básico de información dentro de la aplicación de Realidad Aumentada RA, en el mismo se describe detalladamente las principales clases, funcionalidades y atributos utilizados para la implementación de la aplicación, así mismo se describe como realizar ciertas tareas administrativas dentro de la aplicación, con el fin de que se pueda reutilizar la misma dentro de otros componentes académicos.

**Plataforma.**

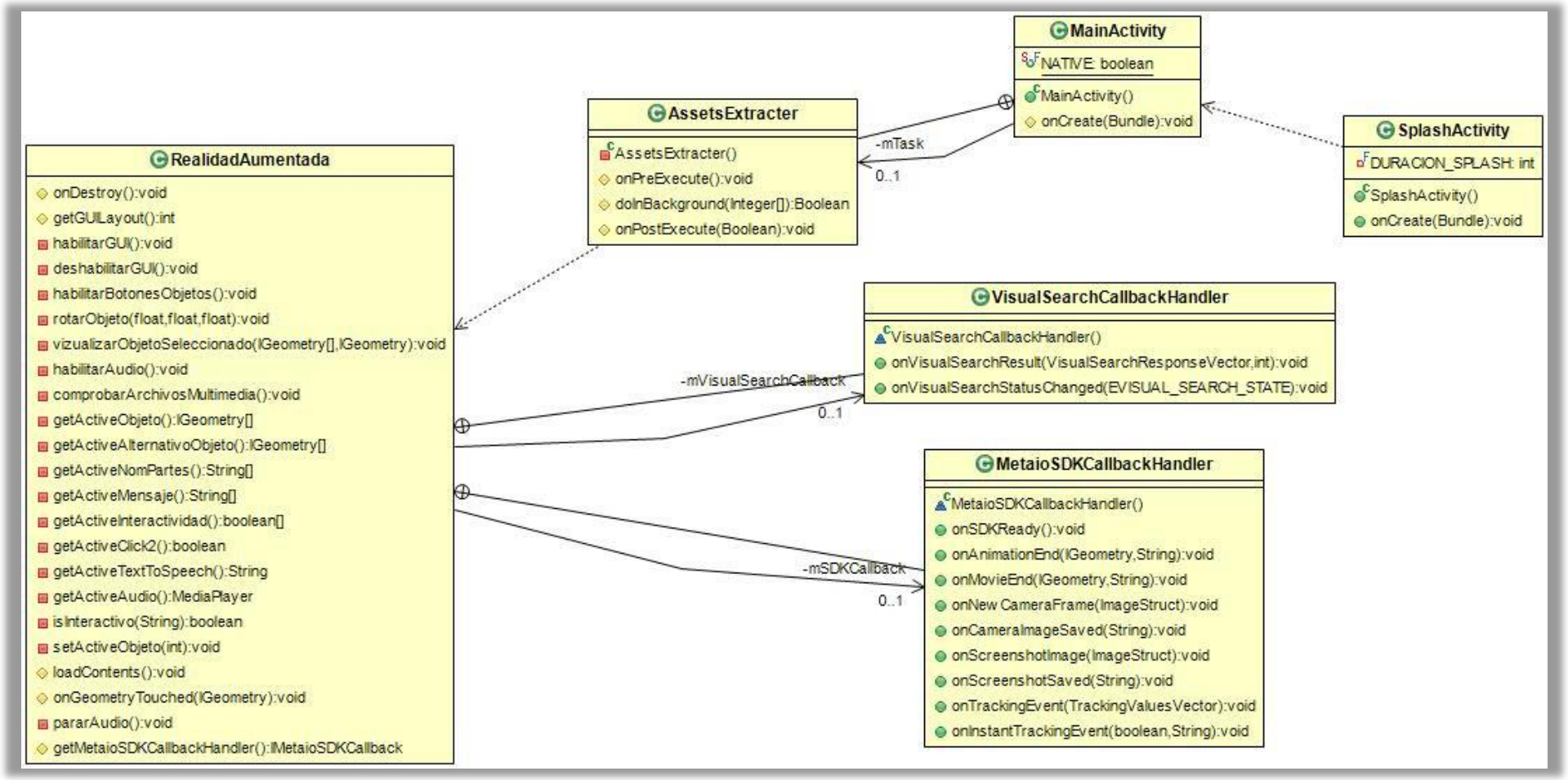
La presente aplicación de RA ha sido desarrolla para el Sistema Operativo SO Android y está enfocada principalmente para dispositivos móviles dotados con una pantalla de entre 7 a 10 pulgadas (Tablet). Para el desarrollo de la aplicación de RA se ha tomado como base la versión 4.4.2 de Android, y tiene soporte desde Android 4.0 en adelante.

**Tecnologías Utilizadas .**

La presente aplicación ha sido desarrollada con el SDK de Android (Java) conjuntamente con el SDK de Metaio (Realidad Aumentada), el entorno de desarrollo elegido para la realización del presente proyecto de fin de carrera es Eclipse. A continuación se muestra un detalle de las tecnologías utilizadas:

- ✓ Plataforma de desarollo: Eclipse Luna versión 4.4.1.
- ✓ Android SDK: Android SDK versión r24.1.2.
- ✓ Metaio SDK: Metaio SDK versión 5.5.2.
- ✓ Java JDK: Java JDK version 8u20.

**Diagrama de Clases.**



**Descripción de Clases.**

| Nombre de la Clase | MainActivity |
|---|---|
| **Detalles** | Clase principal de inicio para cualquier aplicación desarrollada en Android. |
| **Características** | Bolean Native: True, si la aplicación es nativa de Android o False, si se utilizara una implementación AREL. |
| **Comportamiento** | MainActivity(): Constructor de la clase MainActivity<br><br>onCreate(): Método de inicio de la aplicación, es llamado inmediatamente al iniciar la aplicación. |

| Nombre de la Clase | AssetsExtracter |
|---|---|
| **Detalles** | Clase interna o anidada, en la que se detalla las rutinas de códigos necesarias para la extracción de archivos almacenados en el directorio assets. |
| **Características** | No tiene. |
| **Comportamiento** | AssetsExtracter(): Constructor de la clase AssetsExtracter.<br><br>onPreExecute(): Rutina de código que se ejecuta antes de empezar a extraer recursos multimedia localizados en assets.<br><br>doInBackground(): Método utilizado para cargar los archivos, se ejecuta en segundo plano.<br><br>OnPostExecute(): Rutina de código que se ejecuta después de extraer los recursos multimedia localizados en assets. |

| Nombre de la Clase | SplashActivity |
|---|---|
| **Detalles** | Clase que contiene la rutina de código necesaria para mostrar un layout o ventana de carga al iniciar la aplicación (actualmente no se encuentra activa). |
| **Características** | Int DURATION_SPLASH: Determina la duración del primer Splash o imagen de carga que se ejecuta al iniciar la aplicación, se expresa en milisegundos. |
| **Comportamiento** | SplashActivity():Constructor de la clase SplashActivity<br><br>onCreate: Método que se ejecuta al iniciar esta actividad. |

| Nombre de la Clase | VisualSearchCallbackHandler |
|---|---|
| **Detalles** | Clase interna o anidada, en la que se detalla las rutinas de código necesarias para la búsqueda objetos 3D mediante VisualSearch. |
| **Características** | No tiene. |
| **Comportamiento** | onVisualSearchResult():Este método se llama después de haber iniciado con éxito una nueva búsqueda.<br><br>onVisualSearchStatusChanged():Este método es llamado cada vez que el estado de la búsqueda cambia. |

| Nombre de la Clase | MetaioSDKCallback |
|---|---|
| **Detalles** | Clase interna o anidada, en la que se detalla las rutinas de código necesarias para la ejecución de una escena de RA a través de Metaio SDK. |
| **Características** | No tiene. |

| | |
|---|---|
| **Comportamiento** | onSDKReady(): Este método se ejecuta cuando el SDK de Meatio está listo, es decir, cuando ya ha cargado la escena de RA.<br><br>onAnimationEnd(): Este método se ejecuta una vez que alguna animación ha finalizado.<br><br>onMovieEnd(): Este método se ejecuta una vez que un video ha finalizado.<br><br>onNewCameraFrame(): Devuelve el siguiente frame o imagen tomado por la cámara.<br><br>onCameraImagedSaved(): Este método se ejecuta una vez que se ha guardado una imagen capturada por la cámara.<br><br>onScreenhotsaved():Devuelve una captura de pantalla en formato ImageStruct..<br><br>onTrackingEvent(): Este método se ejecuta automáticamente cada vez que exista un cambio en el sistema de tracking o seguimiento.<br><br>onInstandTrackingEvent(): Este método cuando se active un evento de instant tracking o seguimiento instantáneo. |

| **Nombre de la Clase** | **RealidadAumentada** |
|---|---|
| **Detalles** | Clase en la que se ejecuta y controla toda la interactividad de la escena de RA a través de Metaio SDK, contiene todas las rutinas de código necesarias para cargar, controlar y visualizar los objetos 3D. |
| **Características** | String strNomObj: Nombre del objeto (este nombre se mostrara en el botón de la interfaz).<br><br>String [] strNomPartesObj: Contiene el nombre de cada uno de los subobjetos.<br><br>String [] strMensajeObj: Contiene la información desplegable de cada uno de los subobjetos. |

| | |
|---|---|
| | String [] strRutasObj: Contiene las rutas de cada uno de los archivos de los subobjetos.<br><br>String [] strRutasAltObj: Contiene las rutas de cada uno de los archivos de los subobjetos alternativos en caso de tenerlos.<br><br>Integer [] rutasAudioObj: Contiene las rutas de cada uno de los archivos de audio de los subobjetos.<br><br>boolean [] interactividadObj: Determina si el subobjeto es interactivo o estático; true: interactivos; false: estático.<br><br>boolean activarObj: Determina si el todo el objeto se encuentra activo o inactivo; true: activo; false: inactivo.<br><br>float escalaObj: Determina la escala de todo el objeto.<br><br>boolean click2Obj: Determina si la interactividad del subobjeto es de 1 clic o 2 clic; true: 2 clic; false: 1 clic.<br><br>IGeometry [] obj: Guarda el objeto 3D para trabajarlo dentro de la aplicación.<br><br>IGeometry [] altObj: Guarda el objeto 3D alternativo en caso de haberlo para trabajarlo dentro de la aplicación.<br><br>MediaPlayer [] audioObj: Guarda la ruta de los audios del objeto para trabajarlos de dentro de la aplicación.<br><br>int objeto: Guarda el número del objeto actual.<br><br>int subObjeto: Guarda el número del subobjeto actual.<br><br>IGeometry objAnt: Guarda el subobjeto anterior para realizar comparaciones y activar o desactivar dicho objeto.<br><br>boolean error: Determina si hubo un error al cargar los archivos de los objetos y audios por defecto es false; true existe un error; false no existe errores. |

| | |
|---|---|
| | MediaPlayer audioClick: Guarda el audio que se reproduce al seleccionar un subobjeto.<br><br>MediaPlayer audioObjeto: Guarda el audio del objeto actual para trabajarlo dentro de la aplicación.<br><br>TextToSpeech tts: Motor de texto a audio el cual se utiliza si un subobjeto no tiene un archivo de audio asignado.<br><br>String strTextSpeech: Guarda el texto que será leído por el motor de texto a audio. |
| **Comportamiento** | onCreate(Bundle savedInstanceState): Método de inicio de la aplicación, es llamado inmediatamente al crear una instancia de la clase RealidadAumentada.<br><br>onDestroy(): Este método es llamado al terminar la ejecución de la aplicación.<br><br>getGUILayout(): Obtiene el layout de la aplicación.<br><br>onTouch(View v, MotionEvent event) Método ejecutado al producirse un clic en la pantalla.<br><br>onBackPressed() Método ejecutado al presionar el botón regresar del dispositivo android.<br><br>onbtnAmasButtonClick(View v) Evento del botón para aumentar el tamaño de letra del mensaje desplegable.<br><br>onbtnAmenosButtonClick(View v) Evento del botón para disminuir el tamaño de letra del mensaje desplegable.<br><br>onbtnXButtonClick(View v) Evento del botón cerrar, el cual permite cerrar el mensaje desplegable.<br><br>onbtnVozButtonClick(View v) Evento del botón voz el cual permite reproducir el audio del subobjeto. |

| | |
|---|---|
| | onbtnInicioButtonClick(View v) Evento del botón inicio de la interfaz permite desplegar los objetos disponibles.<br><br>onbtnSalirButtonClick(View v) Evento del botón cerrar aplicación el cual permite salir de la aplicación.<br><br>onbtnVolverButtonClick(View v) Evento del botón volver el cual permite salir de la vista personalizada del objeto.<br><br>onbtnAceptarButtonClick(View v) Evento del botón aceptar del cuadro de dialogo para cerrar la aplicación.<br><br>onbtnCancelarButtonClick(View v) Evento del botón cancelar del cuadro de dialogo para cerrar la aplicación.<br><br>onbtnDerechaButtonClick(View v): Evento del botón girar a la derecha, permite girar a la derecha el objeto.<br><br>onbtnIzquierdaButtonClick(View v) Evento del botón girar a la izquierda, permite girar a la izquierda el objeto.<br><br>onbtnArribaButtonClick(View v): Evento del botón girar hacia arriba, permite girar hacia arriba el objeto.<br><br>onbtnAbajoButtonClick(View v): Evento del botón girar hacia abajo, permite girar hacia abajo el objeto.<br><br>onbtnObjeto1ButtonClick(View v): Evento del botón objeto 1, permite cambiar el objeto actual por el objeto numero 1.<br><br>onbtnObjeto2ButtonClick(View v): Evento del botón objeto 2, permite cambiar el objeto actual por el objeto numero 2.<br><br>onbtnObjeto3ButtonClick(View v): Evento del botón objeto 3, permite cambiar el objeto actual por el objeto numero 3.<br><br>onbtnObjeto4ButtonClick(View v): Evento del botón objeto 4, permite cambiar el objeto actual por el objeto numero 4. |

| | |
|---|---|
| | onbtnSalir2ButtonClick(View v): Evento del botón salir que se encuentra en la interfaz, permite salir de la aplicación.<br><br>habilitarGUI() Método que permite habilitar la interfaz gráfica de usuario una vez que Metaio SDK está cargado.<br><br>deshabilitarGUI(): Método q permite deshabilitar la interfaz gráfica de usuario antes de cerrar la aplicación.<br><br>habilitarBotonesObjetos(): Habilita los botones de los objetos si estos tienen como estado: activo.<br><br>rotarObjeto(float f, float g, float h): Permite rotar el objeto mostrado en pantalla en la dirección deseada.<br><br>vizualizarObjetoSeleccionado(IGeometry[] objAct, IGeometry geometry): Permite intercambiar el objeto actual por el objeto seleccionado en la interfaz.<br><br>habilitarAudio(): Habilita los controles de audio y el motor de texto a voz de la aplicación.<br><br>comprobarArchivosMultimedia(): Método que comprueba si todos los archivos de la aplicación han sido cargados correctamente.<br><br>getActiveObjeto(): Permite obtener el objeto actual mostrado en pantalla.<br><br>getActiveAlternativoObjeto():Permite obtener el objeto alternativo actual mostrado en pantalla.<br><br>getActiveNomPartes(): Permite obtener el nombre de los subobjetos mostrados en pantalla.<br><br>getActiveMensaje(): Permite obtener el mensaje desplegable de los subobjetos mostrados en pantalla.<br><br>getActiveInteractividad(): Permite obtener el tipo de interactividad que tiene el subobjeto. |

| | |
|---|---|
| | getActiveClick2() Permite obtener si la interactividad del objeto es de 1 clic o 2 clic.<br><br>getActiveTextToSpeech(): Permite obtener el texto del mensaje mostrado en pantalla para pasarlo a voz en caso de que el subobjeto no tenga un archivo de audio personalizado.<br><br>getActiveAudio(): Permite obtener el audio personalizado del subobjeto actual.<br><br>isInteractivo(String name): Devuelve si un subobjeto es interactivo o no.<br><br>setActiveObjeto(int modelIndex) Permite cambiar el objeto actual por el objeto deseado mediante el número de objeto.<br><br>loadContents(): Método que carga todos los archivos multimedia que serán utilizados dentro de la aplicación.<br><br>onGeometryTouched(IGeometry geometry) Evento que se ejecuta al presionar o dar clic sobre un subobjeto.<br><br>pararAudio(): evento que permite parar el audio que se encuentre reproduciendo.<br><br>onSDKReady() Este método se ejecuta cuando el SDK de Meatio está listo, para trabajar. |

**Código Fuente de la aplicación.**

A continuación se detalla el código fuente de las principales clases, métodos y funciones de la aplicación RA-Desarrollo-Inteligencia, en las cuales se puede apreciar las principales rutinas de código para la carga e interacción con los objetos 3D:

**Clase: Mainactivity.**

```
public class MainActivity extends Activity
{
	//TODO: Select the template, i.e. Native Android or AREL
	public static final boolean NATIVE = true;
	/**
	 * Task that will extract all the assets
	 */
	private AssetsExtracter mTask;
	@Override
	protected void onCreate(Bundle savedInstanceState)
	{
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
		// Enable metaio SDK debug log messages based on build
configuration
		MetaioDebug.enableLogging(BuildConfig.DEBUG);
		// extract all the assets
		mTask = new AssetsExtracter();
		mTask.execute(0);

	}
	/**
	 * This task extracts all the assets to an external or internal location
	 * to make them accessible to metaio SDK
	 */
	private class AssetsExtracter extends AsyncTask<Integer, Integer, Boolean>
	{
		@Override
		protected void onPreExecute()
		{
		}
		@Override
		protected Boolean doInBackground(Integer... params)
		{
			try
			{
				// Extract all assets and overwrite existing files if
debug build
			AssetsManager.extractAllAssets(getApplicationContext(),
BuildConfig.DEBUG);
			}
			catch (IOException e)
			{
				MetaioDebug.log(Log.ERROR, "Error extracting assets:
"+e.getMessage());
				MetaioDebug.printStackTrace(Log.ERROR, e);
				return false;
			}
			return true;
		}
		@Override
		protected void onPostExecute(Boolean result)
		{
			if(NATIVE)
			{
				// create native template and present it
```

```
                        Intent intent = new Intent(getApplicationContext(),
RealidadAumentada.class);
                        startActivity(intent);
                    }
                    else
                    {
                        // create AREL template and present it
                        final String arelConfigFilePath =
AssetsManager.getAssetPath(getApplicationContext(), "arelConfig.xml");
                        MetaioDebug.log("arelConfig to be passed to intent:
"+arelConfigFilePath);
                        Intent intent = new Intent(getApplicationContext(),
ARELViewActivity.class);
                        intent.putExtra(getPackageName()+".AREL_SCENE",
arelConfigFilePath);
                        startActivity(intent);

                    }
                    finish();
            }
        }
}
```

**Clase: Splashactivity.**

```
public class SplashActivity extends Activity {

    // Duración en milisegundos que se mostrará el splash
    private final int DURACION_SPLASH = 5000; // 5 segundos
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Tenemos una plantilla xml donde mostraremos la información que
queramos (logotipo, etc.)
        setContentView(R.layout.template);

        new Handler().postDelayed(new Runnable(){
            public void run(){
             // Cuando pasen los 5 segundos, pasamos a la actividad principal de
la aplicación
             Intent intent = new Intent(SplashActivity.this,
MainActivity.class);
             startActivity(intent);
             finish();
            };
        }, DURACION_SPLASH);
    }
}
```

**Clase: RealidadAumentada.**

**Declaración de variables.**

```
private String strNomObj1 = "Cerebro";
      private String [] strNomPartesObj1 = {"Lóbulo Parietal","Lóbulo
Occipital", "Lóbulo Frontal", "Lóbulo Temporal","Cerebelo",""};
      private String [] strMensajeObj1 = {
                  "Está ubicado en las partes medias y laterales de la cabeza,
los mayores entre los que forman el cráneo. Se trata de la zona cerebral que
está encargada especialmente de recibir las sensaciones de tacto, calor, frío,
presión, dolor, y coordinar el equilibrio.",
                  "Es el centro de nuestro sistema visual de la percepción.
Recibe información visual de esta área, desde donde esta información va a otras
zonas cerebrales que se especializan en temas como la identificación de
palabras.",
                  "Es un área de la corteza cerebral de los vertebrados. En los
seres humanos está localizado en la parte anterior del cerebro. Los lóbulos
temporales están localizados debajo y detrás de los lóbulos frontales.",
                  "El lóbulo temporal se localizada frente al lóbulo occipital,
aproximadamente detrás de cada sien, interviene en la retención de memoria
visual, el procesamiento de la información sensorial, comprensión del lenguaje,
el almacenamiento de nuevos recuerdos y emociones.",
                  "Es una región del encéfalo cuya función principal es de
integrar las vías sensitivas y las vías motoras. Existe una gran cantidad de
haces nerviosos que conectan el cerebelo con otras estructuras encefálicas y con
la médula espinal.",
                   ""};
      private String [] strRutasObj1 =
{"parietal.zip","occipital.zip","frontal.zip","temporal.zip","cerebelo.zip",
"centro.zip"};
      private String [] strRutasAltObj1 = {null,null,null,null,null,null};
      private Integer [] rutasAudioObj1 = {R.raw.lobulo_pariental,
R.raw.lobulo_occipital, R.raw.lobulos_frontales, R.raw.lobulo_temporal ,
R.raw.cerebelo, null };
      private boolean [] interactividadObj1 = {true,true,true,true,true,false};
      private boolean activarObj1 = true ;
      private float escalaObj1 = 1f;
      private boolean click2Obj1 = true;

      private IGeometry [] obj1;
      private IGeometry [] altObj1;

      private MediaPlayer [] audioObj1;

private String strNomObj2 = "Formula CI";

      private String [] strNomPartesObj2 = {"Coeficiente Intelectual","Edad
Cronologica","Edad Mental","Constante",""};
      private String [] strMensajeObj2 = {
                  "El cociente intelectual, o  “CI” en forma abreviada, es una
puntuación, resultado de alguna de las pruebas estandarizadas diseñadas para
valorar la inteligencia.",
                  "La Edad Cronológica se encuentra definida por la edad
natural de la persona",
                  "Se la mide a través de instrumentos como test de
inteligencia, usualmente se utiliza los: \n- Test de Raven.- mide la
```

```
inteligencia general se da principalmente sobre imágenes. \n- Test de Wesley.-
evalúa la parte verbal (numérica, lógica); y la manipulable (rompecabezas,
figuras)",
                    "El valor constante 100 es la medida numérica estándar para
obtener el Coeficiente Intelectual. ",
                    ""
                    };
      private String [] strRutasObj2 =
{"coeficiente.zip","edad_cronologica.zip","edad_mental.zip","constante.zip","ray
a.zip"};
      private String [] strRutasAltObj2 = {null,null,null,null,null};
      private Integer [] rutasAudioObj2 = {R.raw.coeficiente_intelectual,
R.raw.edad_cronologica , R.raw.edad_natural, R.raw.constante, null};
      private boolean [] interactividadObj2 = {true,true,true,true,false};
      private boolean activarObj2 = true;
      private float escalaObj2 = 0.5f;
      private boolean click2Obj2 = true;

      private IGeometry [] obj2;
      private IGeometry [] altObj2;
      private MediaPlayer [] audioObj2;


      private String strNomObj3 = "Helen Adams Keller";

      private String [] strNomPartesObj3 = {"Helen Adams Keller","Ojo","Oido"};

      private String [] strMensajeObj3 = {
                    "(Alabama, 27 de junio de 1880 - Connecticut, 1 de junio de
1968) fue una escritora, oradora y activista política sordo-ciega
estadounidense. A la edad de 19 meses, sufrió una grave enfermedad que le
provocó la pérdida total de la visión y la audición. \nA lo largo de toda su
vida, redactó una multiplicidad de artículos y más de una docena de libros sobre
sus experiencias y modos de entender la vida, entre ellos “La historia de mi
vida (1903) y Luz en mi oscuridad (1927)”. \nHellen es un ícono en el desarrollo
atípico de la inteligencia debido a los logros alcanzados durante su vida pese a
sus discapacidades. Dio conferencias en 25 países y viajo alrededor del mundo.
Su cociente intelectual era equivalente al 92,85%; en una escala en la cual el
57,14% e inferior, representa inteligencias con evidentes problemas de
subdesarrollo normal. El 71,42%  representa a la inteligencia normal y el
porcentaje igual a un superdotado es igual a 100%.",
                    "Cuando cumplió siete años, sus padres decidieron buscar una
instructora y fue así como el Instituto Perkins para Ciegos les envió a una
joven especialista en problemas de visión y con mucha experiencia en el trabajo
con personas ciegas, Anne Sullivan, que se encargó de su formación y logró un
avance en la educación especial centrándose específicamente en su discapacidad
visual. Helen continuó viviendo al lado de Sullivan hasta la muerte de esta en
1936.",
                    "Después de graduarse de la escuela secundaria en Cambridge,
Keller ingresó en el Radcliffe College, donde recibió una licenciatura,
convirtiéndose en la primera persona sordo-ciega en obtener un título
universitario. Durante su juventud, comenzó a apoyar al socialismo y en 1905, se
unió formalmente al Partido Socialista, pesar de su discapacidad auditiva Helen
era una de las activistas más representativas y tenía una de las mayores
acogidas.",
                    };
      private String [] strRutasObj3 = {"hellen.zip","ojo.zip","oido.zip"};
      private String [] strRutasAltObj3 = {null,"ojo2.zip","oido2.zip"};
```

```
	private Integer [] rutasAudioObj3 = { R.raw.hellen, R.raw.ojos,
R.raw.oido};
	private boolean [] interactividadObj3 = {true,true,true};
	private boolean activarObj3 = true ;
	private float escalaObj3 = 0.8f;
	private boolean click2Obj3 = false;

	private IGeometry [] obj3;
	private IGeometry [] altObj3;

	private MediaPlayer [] audioObj3;

	private MetaioSDKCallbackHandler mSDKCallback;
	private VisualSearchCallbackHandler mVisualSearchCallback;

	private int objeto = 1
	private int subObjeto = 0;
	private IGeometry objAnt = null;
	private boolean error = false;


	//deteccion movimiento
	private GestureHandlerAndroid mGestureHandler;
	private int mGestureMask;

	//Controles de la GUI
	private EditText txtObjeto;
	private EditText txtMensaje;
	private TextView btnInico;
	private TextView btnVolver;
	private TextView btnSalir;
	private Button btnObjeto1;
	private Button btnObjeto2;
	private Button btnObjeto3;
	private Button btnObjeto4;
	private View layoutPrincipal;
	private View layoutSplash;
	private View layoutMensaje;
	private View layoutDialogo;
	private View layoutBtnRotacion;
	private View layoutMenuIzq;
	private View layoutMenu;

	//sonido
	private MediaPlayer audioClick;
	private MediaPlayer audioObjeto;

	//Motor de voz
	private TextToSpeech tts =null;

	private String strTextSpeech = "";
```

**Eventos.**

```
       ////////////////////////////////////////////////////////
       //Eventos Generales de la aplicacion --> layout Mensaje//
       ////////////////////////////////////////////////////////

       @Override
       public void onCreate(Bundle savedInstanceState){

              super.onCreate(savedInstanceState);

              mSDKCallback = new MetaioSDKCallbackHandler();
              mVisualSearchCallback = new VisualSearchCallbackHandler();

              //Agrega la deteccion de movimiento
              mGestureMask =
GestureHandler.GESTURE_PINCH|GestureHandler.GESTURE_ROTATE;
              mGestureHandler = new GestureHandlerAndroid(metaioSDK,
mGestureMask);

              if (metaioSDK != null)

       metaioSDK.registerVisualSearchCallback(mVisualSearchCallback);
       }

       @Override
       protected void onDestroy(){
              super.onDestroy();
              mSDKCallback.delete();
              mSDKCallback = null;
              mVisualSearchCallback.delete();
              mVisualSearchCallback = null;
              tts.stop();
              if(audioObjeto != null)
                     audioObjeto.stop();
       }

       @Override
       protected int getGUILayout(){
              // referencia el Layout a esta actividad
              return R.layout.template;
       }

       @Override
       public boolean onTouch(View v, MotionEvent event){
              super.onTouch(v, event);
              mGestureHandler.onTouch(v, event);
              return true;
       }

       @Override
       public void onBackPressed() {
              if (tts != null){
                     layoutDialogo.setVisibility(View.VISIBLE);
                     deshabilitarGUI();
              }
       }
```

```
	////////////////////////////////////////////////////
	//Eventos de los botones del Mensaje --> layout Mensaje//
	////////////////////////////////////////////////////

	public void onbtnAmasButtonClick(View v){

	txtMensaje.setTextSize(TypedValue.COMPLEX_UNIT_PX,txtMensaje.getTextSize()
+1);
	}

	public void onbtnAmenosButtonClick(View v){

	txtMensaje.setTextSize(TypedValue.COMPLEX_UNIT_PX,txtMensaje.getTextSize()
-1);
	}

	public void onbtnXButtonClick(View v){
		layoutMensaje.setVisibility(View.GONE);
	}

	public void onbtnVozButtonClick(View v){

		if(audioObjeto != null){
			if(audioObjeto.isPlaying()){
				audioObjeto.pause();
				audioObjeto.seekTo(0);
				if(audioObjeto == getActiveAudio())

					return;
			}
		}else
			if(tts.isSpeaking()){
				tts.stop();
				if(strTextSpeech.compareTo(getActiveTextToSpeech()) ==
0)
					return;
			}

		strTextSpeech = getActiveTextToSpeech();
		audioObjeto = getActiveAudio();

		if(audioObjeto != null)
			audioObjeto.start();
		else
			tts.speak( strTextSpeech, TextToSpeech.QUEUE_FLUSH, null );

	}


	////////////////////////////////////////////////////////////
	//Eventos de los botones de la barra de menu  --> layout Menu//
	////////////////////////////////////////////////////////////

	public void onbtnInicioButtonClick(View v){
		layoutMensaje.setVisibility(View.GONE);
		if(layoutMenuIzq.getVisibility() != View.VISIBLE)
			layoutMenuIzq.setVisibility(View.VISIBLE);
		else
```

```
                layoutMenuIzq.setVisibility(View.GONE);
        }

        public void onbtnSalirButtonClick(View v){
                layoutDialogo.setVisibility(View.VISIBLE);
                deshabilitarGUI();
        }

        public void onbtnVolverButtonClick(View v){
                btnInico.setVisibility(View.VISIBLE);
                btnSalir.setVisibility(View.VISIBLE);
                btnVolver.setVisibility(View.GONE);
                layoutMensaje.setVisibility(View.GONE);
                if (getActiveAlternativoObjeto()[subObjeto] != null)
                        getActiveAlternativoObjeto()[subObjeto].setVisible(false);
                objAnt = null;
                setActiveObjeto(objeto);
        }


        ////////////////////////////////////////////////////////
        //Eventos de los botones de cerrar APP --> layout Dialogo//
        ////////////////////////////////////////////////////////

        public void onbtnAceptarButtonClick(View v){
                finish();
        }

        public void onbtnCancelarButtonClick(View v){
                layoutDialogo.setVisibility(View.GONE);
                layoutMenu.setVisibility(View.VISIBLE);
                layoutBtnRotacion.setVisibility(View.VISIBLE);
        }


        ////////////////////////////////////////////////////////
        //Eventos de los botones de Rotacion --> layout Rotacion//
        ////////////////////////////////////////////////////////

        public void onbtnDerechaButtonClick(View v){
                rotarObjeto(0f,0f,0.1f);
        }

        public void onbtnIzquierdaButtonClick(View v){
                rotarObjeto(0f,0f,-0.1f);
        }

        public void onbtnArribaButtonClick(View v){
                rotarObjeto(-0.1f,0f,0f);
        }

        public void onbtnAbajoButtonClick(View v){
                rotarObjeto(0.1f,0f,0f);
        }


        ////////////////////////////////////////////////////////
        //Eventos de los botones de Objetos --> layout MenuIzq//
        ////////////////////////////////////////////////////////
```

```
	public void onbtnObjeto1ButtonClick(View v){
		objeto = 1;
		subObjeto = 0;
		setActiveObjeto(objeto);
		txtObjeto.setText("");
		txtMensaje.setText("");
	}
	public void onbtnObjeto2ButtonClick(View v){
		objeto = 2;
		subObjeto = 0;
		setActiveObjeto(objeto);
		txtObjeto.setText("");
		txtMensaje.setText("");
	}

	public void onbtnObjeto3ButtonClick(View v){
		objeto = 3;
		subObjeto = 0;
		setActiveObjeto(objeto);
		txtObjeto.setText("");
		txtMensaje.setText("");
	}

	public void onbtnObjeto4ButtonClick(View v){
		objeto = 4;
		subObjeto = 0;
		setActiveObjeto(objeto);
		txtObjeto.setText("");
		txtMensaje.setText("");
	}

	public void onbtnSalir2ButtonClick(View v)   {
		layoutDialogo.setVisibility(View.VISIBLE);
		deshabilitarGUI();
	}
```

**Métodos y funciones.**

```
private void habilitarGUI() {

		txtObjeto  = (EditText) findViewById(R.id.txtTitulo);
		txtMensaje = (EditText) findViewById(R.id.txtTexto);

		btnInico = (TextView) findViewById(R.id.btnInicio);
		btnVolver = (TextView) findViewById(R.id.btnVolver);
		btnSalir = (TextView) findViewById(R.id.btnSalir);

		layoutPrincipal = mGUIView.findViewById(R.id.layoutPrincipal);
		layoutSplash = mGUIView.findViewById(R.id.layoutSplash);
		layoutMensaje = mGUIView.findViewById(R.id.layoutMensaje);
		layoutMenu = mGUIView.findViewById(R.id.layoutMenu);
		layoutDialogo = mGUIView.findViewById(R.id.layoutDialogo);
		layoutBtnRotacion = mGUIView.findViewById(R.id.layoutRotacion);
```

```
            layoutMenuIzq =  mGUIView.findViewById(R.id.layoutMenuIzq);

            layoutPrincipal.setBackgroundColor(Color.TRANSPARENT);
            layoutSplash.setVisibility(View.GONE);
            layoutMenu.setVisibility(View.VISIBLE);
            layoutBtnRotacion.setVisibility(View.VISIBLE);

            habilitarBotonesObjetos();

      }

      private void deshabilitarGUI() {
            layoutMenu.setVisibility(View.GONE);
            layoutMenuIzq.setVisibility(View.GONE);
            layoutBtnRotacion.setVisibility(View.GONE);
            layoutMensaje.setVisibility(View.GONE);
      }

      private void habilitarBotonesObjetos() {
            if(activarObj1){
                  btnObjeto1 = (Button) findViewById(R.id.btnObjeto1);
                  btnObjeto1.setText(strNomObj1);
                  btnObjeto1.setVisibility(View.VISIBLE);
            }
            if(activarObj2){
                  btnObjeto2 = (Button) findViewById(R.id.btnObjeto2);
                  btnObjeto2.setText(strNomObj2);
                  btnObjeto2.setVisibility(View.VISIBLE);
            }
            if(activarObj3){
                  btnObjeto3 = (Button) findViewById(R.id.btnObjeto3);
                  btnObjeto3.setText(strNomObj3);
                  btnObjeto3.setVisibility(View.VISIBLE);
            }
            if(activarObj4){
                  btnObjeto4 = (Button) findViewById(R.id.btnObjeto4);
                  btnObjeto4.setText(strNomObj4);
                  btnObjeto4.setVisibility(View.VISIBLE);
            }
      }


      ///////////////////////
      // Metodos de Apoyo //
      ///////////////////////


      private void rotarObjeto(float f, float g, float h) {
            IGeometry[] objAct  = getActiveObjeto();
            IGeometry[] objAlt = getActiveAlternativoObjeto();

            if ((objAlt[subObjeto] != null) && (objAlt[subObjeto].isVisible()))
// preguntar si es visible y solo se rota cuando sea visible
                  objAlt[subObjeto].setRotation(new Rotation(f,g,h), true);
            else
                  for(int i = 0; i < objAct.length; i++){
                        objAct[i].setRotation(new Rotation(f,g,h), true);

            }
```

```
	}

	private void vizualizarObjetoSeleccionado(IGeometry[] objAct, IGeometry
geometry) {

		btnInico.setVisibility(View.GONE);
		btnSalir.setVisibility(View.GONE);
		btnVolver.setVisibility(View.VISIBLE);

		IGeometry[] objAlt = getActiveAlternativoObjeto();

		for(int i = 0; i < objAct.length; i++){
			objAct[i].setVisible(false);
		}

		if (objAlt[subObjeto] == null)
			geometry.setVisible(true);
		else
			objAlt[subObjeto].setVisible(true);
	}

	private void habilitarAudio() {
		audioClick = MediaPlayer.create(this, R.raw.click);
		audioClick.setVolume(0.1f, 0.1f);
		tts = new TextToSpeech( this, this );
	}


	private void comprobarArchivosMultimedia() {
		if (error){
			AlertDialog.Builder alertDialog = new
AlertDialog.Builder(this);
			alertDialog.setTitle("ERROR");
			alertDialog.setMessage( "Error al cargar los archivos
Multimedia\nConsulte al Administrador");
			alertDialog.setCancelable(false);
			alertDialog.setPositiveButton("Ok",new
DialogInterface.OnClickListener() {
				public void onClick(DialogInterface dialog, int id) {
				 finish();
				}
			});
			alertDialog.show();
		}
	}


	private IGeometry[] getActiveObjeto() {
		if (objeto == 1)
			return obj1;
		if (objeto == 2)
			return obj2;
		if (objeto == 3)
			return obj3;
		if (objeto == 4)
			return obj4;
		return null;
	}
```

```
	private IGeometry[] getActiveAlternativoObjeto() {
		if (objeto == 1)
			return altObj1;
		if (objeto == 2)
			return altObj2;
		if (objeto == 3)
			return altObj3;
		if (objeto == 4)
			return altObj4;
		return null;
	}

	private String [] getActiveNomPartes() {
		if (objeto == 1)
			return strNomPartesObj1;
		if (objeto == 2)
			return strNomPartesObj2;
		if (objeto == 3)
			return strNomPartesObj3;
		if (objeto == 4)
			return strNomPartesObj4;
		return null;
	}

	private String[] getActiveMensaje() {
		if (objeto == 1)
			return strMensajeObj1;
		if (objeto == 2)
			return strMensajeObj2;
		if (objeto == 3)
			return strMensajeObj3;
		if (objeto == 4)
			return strMensajeObj4;
		return null;
	}

	private boolean [] getActiveInteractividad() {
		if (objeto == 1)
			return interactividadObj1;
		if (objeto == 2)
			return interactividadObj2;
		if (objeto == 3)
			return interactividadObj3;
		if (objeto == 4)
			return interactividadObj4;
		return null;
	}

	private boolean getActiveClick2() {
		if (objeto == 1)
			return click2Obj1;
		if (objeto == 2)
			return click2Obj2;
		if (objeto == 3)
			return click2Obj3;
		if (objeto == 4)
			return click2Obj4;
		return false;
	}
```

```
	private String getActiveTextToSpeech() {
		if (objeto == 1)
			return strMensajeObj1[subObjeto];
		if (objeto == 2)
			return strMensajeObj2[subObjeto];
		if (objeto == 3)
			return strMensajeObj3[subObjeto];
		if (objeto == 4)
			return strMensajeObj4[subObjeto];
		return null;
	}

	private MediaPlayer getActiveAudio() {
		if (objeto == 1)
			return audioObj1[subObjeto];
		if (objeto == 2)
			return audioObj2[subObjeto];
		if (objeto == 3)
			return audioObj3[subObjeto];
		if (objeto == 4)
			return audioObj4[subObjeto];
		return null;
	}

	private boolean isInteractivo(String name) {

		boolean [] interactividadAct = getActiveInteractividad();
		String[] strNomPartesAct = getActiveNomPartes();

		for(int i = 0; i < interactividadAct.length; i++){
			if(name.compareTo(strNomPartesAct[i]) == 0)
				if (interactividadAct[i]){
					subObjeto = i;
					return true;
				}
		}


		return false;
	}

	private void setActiveObjeto(int modelIndex){

		pararAudio();

		if(activarObj1){
			for(int i = 0; i < obj1.length; i++){
				obj1[i].setVisible(modelIndex == 1);
				obj1[i].setScale(escalaObj1);
				obj1[i].setRotation(new Rotation(0f,0f,0f));

			}
		}
		if(activarObj2){
			for(int i = 0; i < obj2.length; i++){
				obj2[i].setVisible(modelIndex == 2);
				obj2[i].setScale(escalaObj2);
				obj2[i].setRotation(new Rotation(0f,0f,0f));
```

```
                }
            }
            if(activarObj3){
                for(int i = 0; i < obj3.length; i++){
                    obj3[i].setVisible(modelIndex == 3);
                    obj3[i].setScale(escalaObj3);
                    obj3[i].setRotation(new Rotation(0f,0f,0f));
                }
            }
            if(activarObj4){
                for(int i = 0; i < obj4.length; i++){
                    obj4[i].setVisible(modelIndex == 4);
                    obj4[i].setScale(escalaObj4);
                    obj4[i].setRotation(new Rotation(0f,0f,0f));
                }
            }


            mSDKCallback.onTrackingEvent(metaioSDK.getTrackingValues());
      }

      @Override
      protected void loadContents()   {

            try   {

                  // recoge la ruta del archivo de configuracion de Tracking o
Seguimiento
                  String trackingConfigFile =
AssetsManager.getAssetPath(getApplicationContext(),
"TrackingData_MarkerlessFast.xml");

                  // Assigning tracking configuration
                  boolean result =
metaioSDK.setTrackingConfiguration(trackingConfigFile);
                  MetaioDebug.log("Tracking data loaded: " + result);

                  // Rutina para cargar el objeto 1
                  if(activarObj1){
                        obj1 = new IGeometry[strNomPartesObj1.length];
                        audioObj1 = new MediaPlayer[strNomPartesObj1.length];

                        altObj1 = new IGeometry[strNomPartesObj1.length];
                        for(int i = 0; i < obj1.length; i++){
                              final String Model =
AssetsManager.getAssetPath(getApplicationContext(), strRutasObj1[i]);

                              if (Model != null){
                                    obj1[i] =
metaioSDK.createGeometry(Model);
                                    if (obj1[i] != null){
                                          //carga el Objeto 3D
                                          obj1[i].setScale(escalaObj1);

      obj1[i].setName(strNomPartesObj1[i]);
                                          mGestureHandler.addObject(obj1[i],
3);
                                          //carga el Audio
                                          if(rutasAudioObj1[i] != null)
```

```
                                                audioObj1 [i] =
MediaPlayer.create(this, rutasAudioObj1[i]);
                                          else
                                                audioObj1 [i] = null;
                                    }else
                                          MetaioDebug.log(Log.ERROR, "Error
al cargar la geometria de "+ strRutasObj1[i]+ ": " + obj1[i]);
                                    }
                                    //cargar Obj Alternativo
                                    if( strRutasAltObj1[i] != null){
                                          final String ModelAlt =
AssetsManager.getAssetPath(getApplicationContext(), strRutasAltObj1[i]);
                                          if (ModelAlt != null){
                                                altObj1[i] =
metaioSDK.createGeometry(ModelAlt);
                                                if (altObj1[i] != null){

      altObj1[i].setScale(escalaObj1);
altObj1[i].setName(strNomPartesObj1[i]);
            altObj1[i].setVisible(false);
                                                }
                                          }else
                                                MetaioDebug.log(Log.ERROR, "Error
al cargar la geometria alternativa de "+ strRutasAltObj1[i]+ ": " + obj1[i]);
                                    }else
                                          altObj1[i] = null;
                              }
                        }

                        // Rutina para cargar el objeto 2
                        if(activarObj2){
                              obj2 = new IGeometry[strNomPartesObj2.length];
                              audioObj2 = new MediaPlayer[strNomPartesObj2.length];

                              altObj2 = new IGeometry[strNomPartesObj2.length];

                              for(int i = 0; i < obj2.length; i++){
                                    final String Model =
AssetsManager.getAssetPath(getApplicationContext(), strRutasObj2[i]);

                                    if (Model != null){

                                          obj2[i] =
metaioSDK.createGeometry(Model);

                                          if (obj2[i] != null){
                                                //carga el Objeto 3D
                                                obj2[i].setScale(escalaObj2);

      obj2[i].setName(strNomPartesObj2[i]);
      mGestureHandler.addObject(obj2[i], 3);
                                                //carga el Audio
                                                if(rutasAudioObj2[i] != null)
                                                      audioObj2 [i] =
MediaPlayer.create(this, rutasAudioObj2[i]);
                                                else
                                                      audioObj2 [i] = null;
                                          }else
```

```
                                        MetaioDebug.log(Log.ERROR, "Error
al cargar la geometria de "+ strRutasObj2[i]+ ": " + obj2[i]);
                                }
                                //cargar Obj Alternativo
                                if( strRutasAltObj2[i] != null){
                                        final String ModelAlt =
AssetsManager.getAssetPath(getApplicationContext(), strRutasAltObj2[i]);
                                        if (ModelAlt != null){
                                                altObj2[i] =
metaioSDK.createGeometry(ModelAlt);
                                                if (altObj2[i] != null){

      altObj2[i].setScale(escalaObj2);
      altObj2[i].setName(strNomPartesObj2[i]);
             altObj2[i].setVisible(false);
                                                }
                                        }else
                                                MetaioDebug.log(Log.ERROR, "Error
al cargar la geometria alternativa de "+ strRutasAltObj2[i]+ ": " + obj1[i]);
                                }else
                                        altObj2[i] = null;
                        }
                }

                // Rutina para cargar el objeto 3
                if(activarObj3){
                        obj3 = new IGeometry[strNomPartesObj3.length];
                        audioObj3 = new MediaPlayer[strNomPartesObj3.length];

                        altObj3 = new IGeometry[strNomPartesObj3.length];
                        for(int i = 0; i < obj3.length; i++){
                                final String Model =
AssetsManager.getAssetPath(getApplicationContext(), strRutasObj3[i]);

                                if (Model != null){
                                        obj3[i] =
metaioSDK.createGeometry(Model);

                                        if (obj3[i] != null){
                                                //carga el Objeto 3D
                                                obj3[i].setScale(escalaObj3)
                                        obj3[i].setName(strNomPartesObj3[i]);
                                                mGestureHandler.addObject(obj3[i],
3);
                                                // carga el Audio
                                                if(rutasAudioObj3[i] != null)
                                                        audioObj3 [i] =
MediaPlayer.create(this, rutasAudioObj3[i]);
                                                else
                                                        audioObj3 [i] = null;
                                        }else
                                                MetaioDebug.log(Log.ERROR, "Error
al cargar la geometria de "+ strRutasObj3[i]+ ": " + obj3[i]);
                                }
                                //cargar Obj Alternativo
                                if( strRutasAltObj3[i] != null){
                                        final String ModelAlt =
AssetsManager.getAssetPath(getApplicationContext(), strRutasAltObj3[i]);
                                        if (ModelAlt != null){
```

```
                                          altObj3[i] =
metaioSDK.createGeometry(ModelAlt);
                                          if (altObj3[i] != null){

      altObj3[i].setScale(escalaObj3);
      altObj3[i].setName(strNomPartesObj3[i]);
      altObj3[i].setVisible(false);
                                          }
                                    }else
                                          MetaioDebug.log(Log.ERROR, "Error
al cargar la geometria alternativa de "+ strRutasAltObj3[i]+ ": " + obj1[i]);
                              }else
                                    altObj3[i] = null;
                        }
                  }

                  // Rutina para cargar el objeto 4
                  if(activarObj4){
                        obj4 = new IGeometry[strNomPartesObj4.length];
                        audioObj4 = new
MediaPlayer[strNomPartesObj4.length];
                        altObj4 = new IGeometry[strNomPartesObj4.length];
                        for(int i = 0; i < obj4.length; i++){
                              final String Model =
AssetsManager.getAssetPath(getApplicationContext(), strRutasObj4[i]);

                              if (Model != null){

                                    obj4[i] =
metaioSDK.createGeometry(Model);

                                    if (obj4[i] != null){
                                          //cargar el Objeto 3D
                                          obj4[i].setScale(escalaObj4);
                                    obj4[i].setName(strNomPartesObj4[i]);
      mGestureHandler.addObject(obj4[i], 3);
                                          // carga el Audio
                                          if(rutasAudioObj4[i] != null)
                                                audioObj4 [i] =
MediaPlayer.create(this, rutasAudioObj4[i]);
                                          else
                                                audioObj4 [i] = null;
                                    }else
                                          MetaioDebug.log(Log.ERROR, "Error
al cargar la geometria de "+ strRutasObj4[i]+ ": " + obj4[i]);
                              }
                              //cargar Obj Alternativo
                              if( strRutasAltObj4[i] != null){
                                    final String ModelAlt =
AssetsManager.getAssetPath(getApplicationContext(), strRutasAltObj4[i]);
                                    if (ModelAlt != null){
                                          altObj4[i] =
metaioSDK.createGeometry(ModelAlt);
                                          if (altObj4[i] != null){

      altObj4[i].setScale(escalaObj4);
      altObj4[i].setName(strNomPartesObj4[i]);
      altObj4[i].setVisible(false);
                                          }
```

```
                                    }else
                                          MetaioDebug.log(Log.ERROR, "Error
al cargar la geometria alternativa de "+ strRutasAltObj4[i]+ ": " + obj1[i]);
                                }else
                                    altObj4[i] = null;
                            }
                        }
                        setActiveObjeto(objeto);
                }
                catch (Exception e)
                {
                        error = true;
                }
        }

        @Override
        protected void onGeometryTouched(IGeometry geometry){

                if(isInteractivo(geometry.getName())){ //preguntamos si el objeto
es interactivo
                        audioClick.start();//reproducimos el sonido de click al
seleccionar un objeto
                        IGeometry[] objAct  = getActiveObjeto();

                        if(objAnt != null) // quitamos la transaparencia al objeto
anterior
                                objAnt.setTransparency(0f);

                        if(getActiveClick2()){// preguntamos si la interactividad es
de 2 click o de 1 click
                                if((objAnt != null) &&
(geometry.getName().compareTo(objAnt.getName()) == 0)){//preguntamos si el
objeto es el mismo que el anterior
                                        geometry.setTransparency(0f);

        vizualizarObjetoSeleccionado(objAct,geometry);//vizualizamos el objeto
                                }else{
                                        pararAudio(); //paramos el audio si se esta
intercambiando de objeto
                                        geometry.setTransparency(0.5f);//Si es un objeto
diferente se lo hace transparente
                                }
                        }else{

        vizualizarObjetoSeleccionado(objAct,geometry);//vizualizamos el objeto
                        }

                        objAnt = geometry;// guardamos el objeto actual

                        String[] strNomPartesAct = getActiveNomPartes();
                        String[] strMensajeAct = getActiveMensaje();

                        txtObjeto.setText(strNomPartesAct[subObjeto]); // presentamos
el nombre del objeto
                        txtMensaje.setText(strMensajeAct[subObjeto]); // presentamos
el mensaje del objeto
                        //vizualizamos el cuadro del mensaje desplegable
                        layoutMensaje.setVisibility(View.VISIBLE);
                        layoutMenuIzq.setVisibility(View.GONE);
```

```
		}
		MetaioDebug.log("Template.onGeometryTouched: " + geometry);
	}


	private void pararAudio() {
		//parar audio
		if(audioObjeto != null){
			if(audioObjeto.isPlaying()){
				audioObjeto.pause();
				audioObjeto.seekTo(0);
			}
		}else
			if(tts != null){
				if(tts.isSpeaking()){
					tts.stop();
				}
			}
	}

	@Override
	protected IMetaioSDKCallback getMetaioSDKCallbackHandler(){
		return mSDKCallback;
	}

	final class MetaioSDKCallbackHandler extends IMetaioSDKCallback
	{
		@Override
		public void onSDKReady(){

			MetaioDebug.log("The SDK is ready");
			// show GUI
			runOnUiThread(new Runnable(){
				@Override
				public void run(){
					comprobarArchivosMultimedia();
					habilitarGUI();
					habilitarAudio();
				}
			});
		}

		@Override
		public void onAnimationEnd(IGeometry geometry, String
animationName){
			MetaioDebug.log("animation ended" + animationName);
		}

		@Override
		public void onMovieEnd(IGeometry geometry, String name)   {
			MetaioDebug.log("movie ended" + name);
		}

		@Override
		public void onNewCameraFrame(ImageStruct cameraFrame){
			MetaioDebug.log("a new camera frame image is delivered" +
cameraFrame.getTimestamp());
		}
```

```
		@Override
		public void onCameraImageSaved(String filepath){
			MetaioDebug.log("a new camera frame image is saved to" +
filepath);
		}

		@Override
		public void onScreenshotImage(ImageStruct image){
			MetaioDebug.log("screenshot image is received" +
image.getTimestamp());
		}

		@Override
		public void onScreenshotSaved(String filepath){
			MetaioDebug.log("screenshot image is saved to" + filepath);
		}

		@Override
		public void onTrackingEvent(TrackingValuesVector trackingValues){
			for (int i=0; i<trackingValues.size(); i++)
			{
				final TrackingValues v = trackingValues.get(i);
				MetaioDebug.log("Tracking state for COS
"+v.getCoordinateSystemID()+" is "+v.getState());
			}
		}
```

# MANUAL DEL ADMINISTRADOR

RA-Desarrollo-Inteligencia

## Descripción breve

El Manual del Administrador comprende la guía de instalación del Entorno de Desarrollo para Android y la administración de la aplicación de Realidad Aumentada.

Michael Freire
mfffreire@utpl.edu.ec

**TABLA DE CONTENIDOS**

## 1. INTRODUCCION.

Para mayor compresión el presente manual del administrador está dividido en dos partes, la primera parte tiene la finalidad de guiar al usuario a través de la instalación de todos los componentes necesarios para el desarrollo de aplicaciones de Realidad Aumentada RA para Android mediante el SDK de Metaio. Mientras, que la segunda parte aborda la administración de la aplicación de RA desarrollada en el actual proyecto de fin de carrera a nivel de programación.

## 2. INSTALACIÓN DEL ENTORNO DE DESARROLLO.

Para el desarrollo de la aplicación de RA se ha utilizado al entorno de desarrollo de Eclipse por su gran adaptabilidad y usabilidad. A continuación se detalla los pasos necesarios para instalar y configurar Eclipse para el desarrollo de aplicaciones móviles de RA basadas en Metaio.

**Nota**: Todos los recursos y archivos presentes en este manual serán adjuntados al mismo en un CD.

### 2.1. Componentes necesarios.

Para la puesta en marcha del entorno de desarrollo parara aplicaciones móviles de RA se requiere los siguientes componentes:

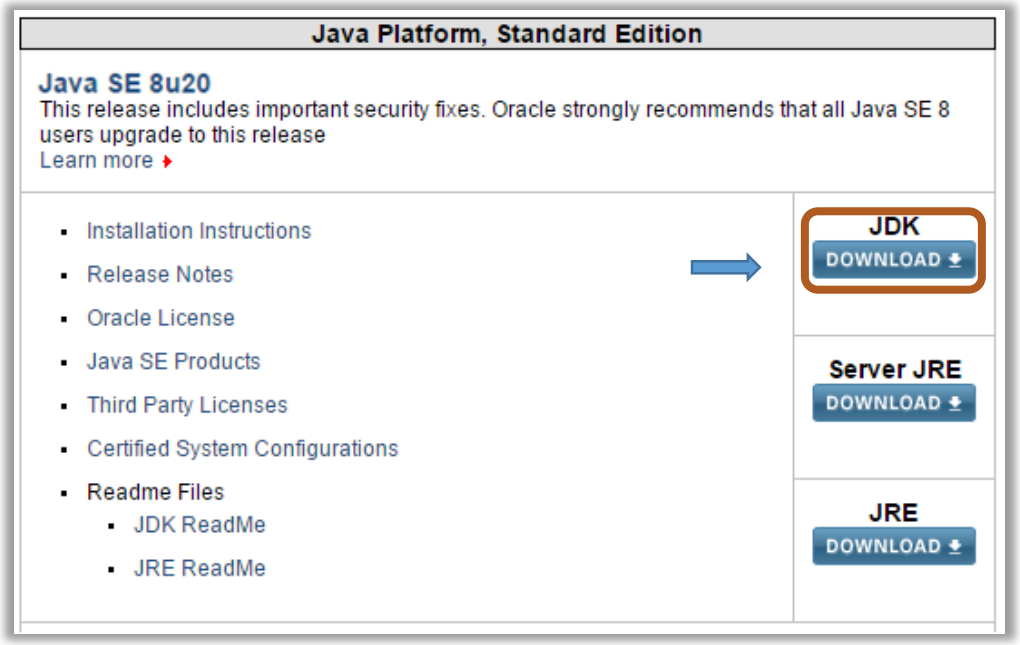
- ✓ JDK de Java.
- ✓ Entorno de desarrollo Eclipse.
- ✓ SDK de Android.
- ✓ SDK de Metaio para Android.
- ✓ Configuración del dispositivo móvil.
- ✓ Compilación y ejecución de ejemplos de Metaio SDK.

**Nota:** Toda la instalación de los diversos componentes está hecha bajo el Sistema Operativo Windows 7 de 32 bits.

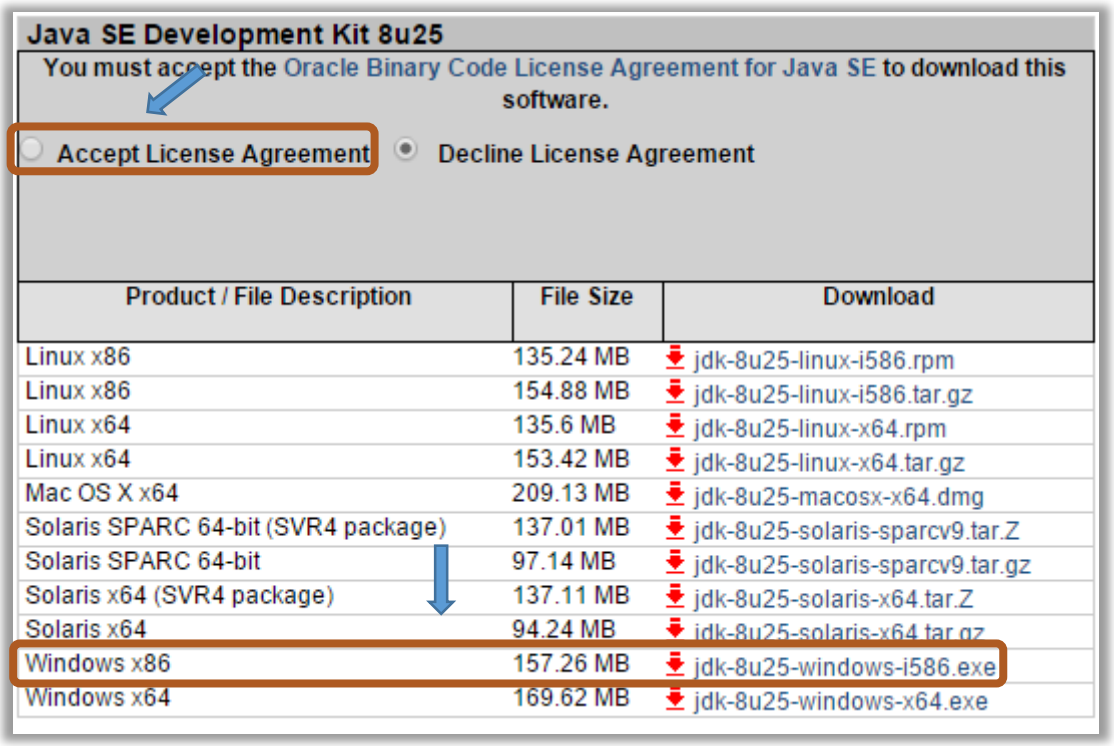
### 2.2. Instalación JDK de Java.

Para instalar el JDK de Java, se realiza lo siguiente:

1. Ingresar al siguiente enlace y se procede a descargar la última versión del JDK de java.
   - ✓ http://www.oracle.com/technetwork/java/javase/downloads/index.html




2. En la siguiente ventana se acepta la licencia del JDK de Java y posteriormente se procede a descargar el Java SE Development Kit (JDK) correspondiente al sistema operativo que se esté utilizando, en este caso es la versión Windows x86 (32 bits).




3. A continuación se procede a instalar el JDK de Java para lo cual se debe ejecutar el fichero descargado "jdk-8u25-windows-i586.exe" y se lo instala con las configuraciones por defecto.


4. Esperamos a que la instalación termine y listo ya se tiene el JDK de Java instalado en el equipo.

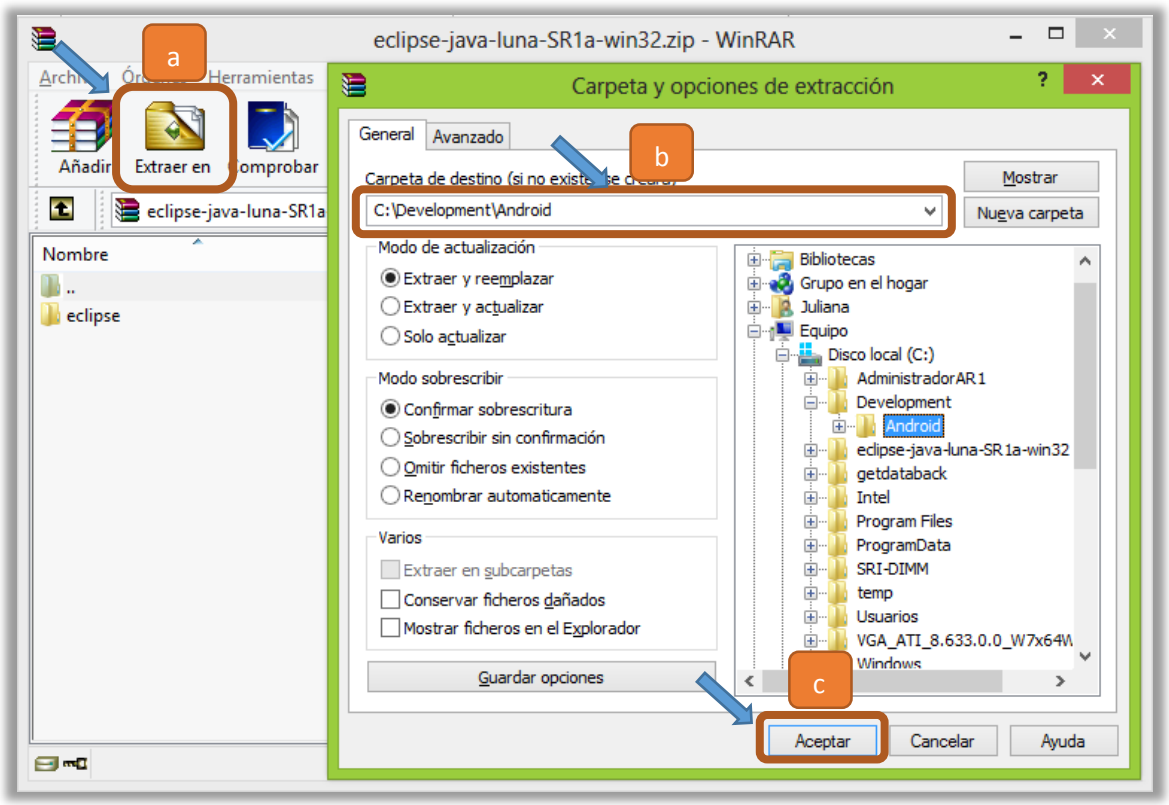
### 2.3. Instalación de Eclipse.

Antes de empezar con la instalación de Eclipse primeramente se debe crear el siguiente directorio “C:\Development\Android\” en el cual se instalara todos los componentes del entorno de desarrollo de RA para Android. Una vez creado el directorio se procede a instalar Eclipse, para lo cual se debe realizar los siguientes pasos:

1. Descargar la última versión Eclipse Windows (en este caso es Windows 32 Bit) desde el siguiente enlace:

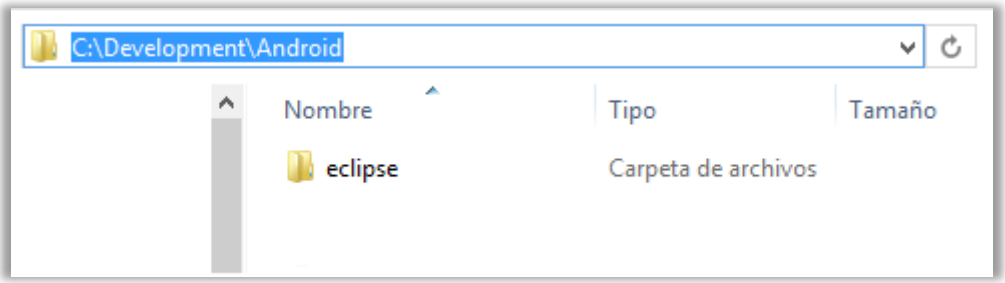
   http://www.eclipse.org/downloads/


2. A continuación se procede a descomprimir el archivo descargado “eclipse-java-luna-SR1a-win32.zip” en el directorio anteriormente creado “C:\Development\Android\”. Si se tiene instalado el programa Winrar simplemente se da doble clic al archivo y se mostrara una ventana donde se procede a descomprimirlo de la siguiente forma:

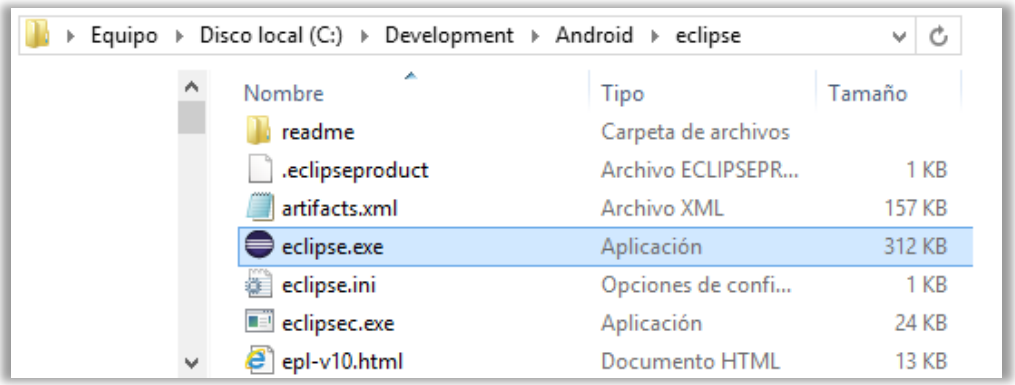



a. Se da Clic en el botón “Extraer en”, luego
b. Se debe seleccionar la ruta de descompresión del archivo C:\Development\Android\”.
c. Por último se da clic en Aceptar y listo.

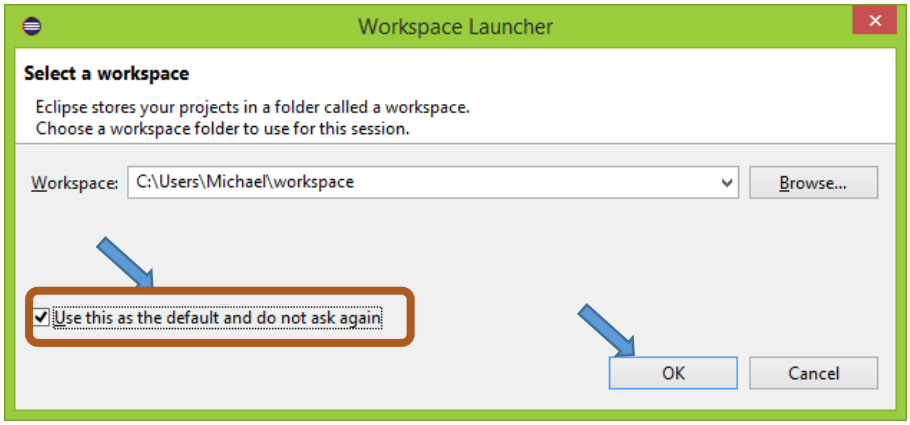
3. Esperamos que finalice la descompresión del archivo que contiene el entorno de desarrollo de Eclipse y al final debería aparecer la carpeta “eclipse” dentro del directorio C:\Development\Android\”.de la siguiente forma:




4. Posteriormente se debe ejecutar el entorno de desarrollo de eclipse, para lo cual se da doble clic en el archivo “eclipse.exe” que se encuentra dentro de la carpeta eclipse.




5. Por último se configura en la ventana emergente el directorio para el espacio de trabajo (donde se guardaran los proyectos de Android) dejamos el directorio que aparece por defecto, y damos un visto a la opción “Use this as a default and do not ask again” y después a “OK”.




### 2.4. Instalación de Android SDK.

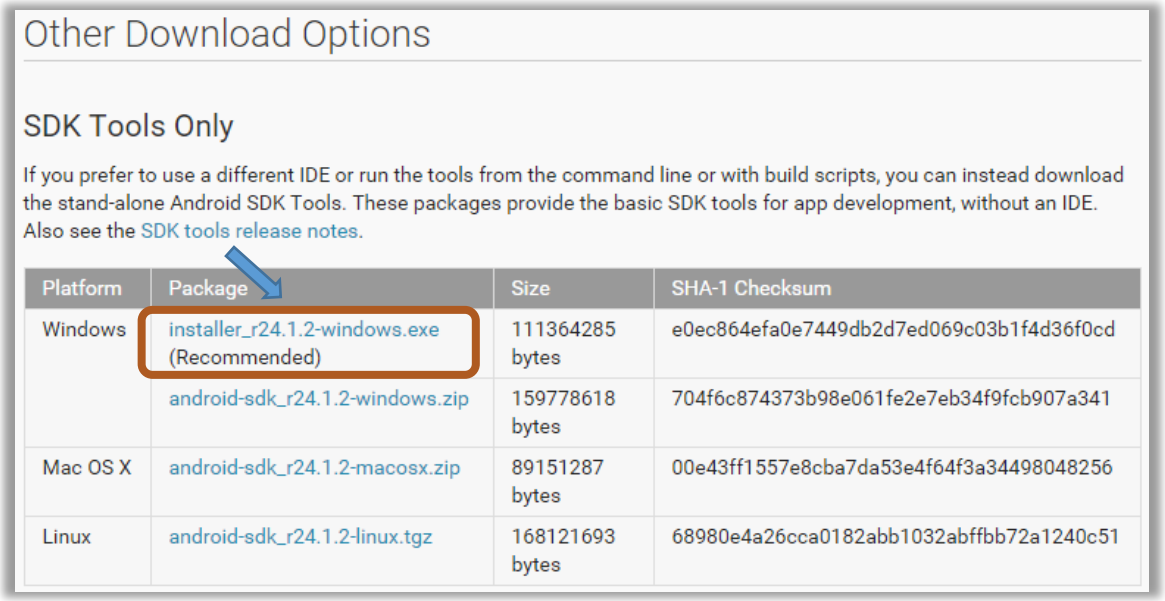
Para el correcto funcionamiento del SDK de Android se debe instalar y configurar los siguientes componentes:

1. Android SDK Tools.
2. ADT plugin.
3. Configurar Android SDK Manager.

### 2.4.1. Instalar Android SDK Tools.

Para la instalación de Android SDK Tools se debe realizar los siguientes pasos:

1. Primeramente se debe descargar el Android SDK Tools para Windows 32 bits desde el siguiente enlace:
   - ✓ http://developer.android.com/sdk/index.html#Other



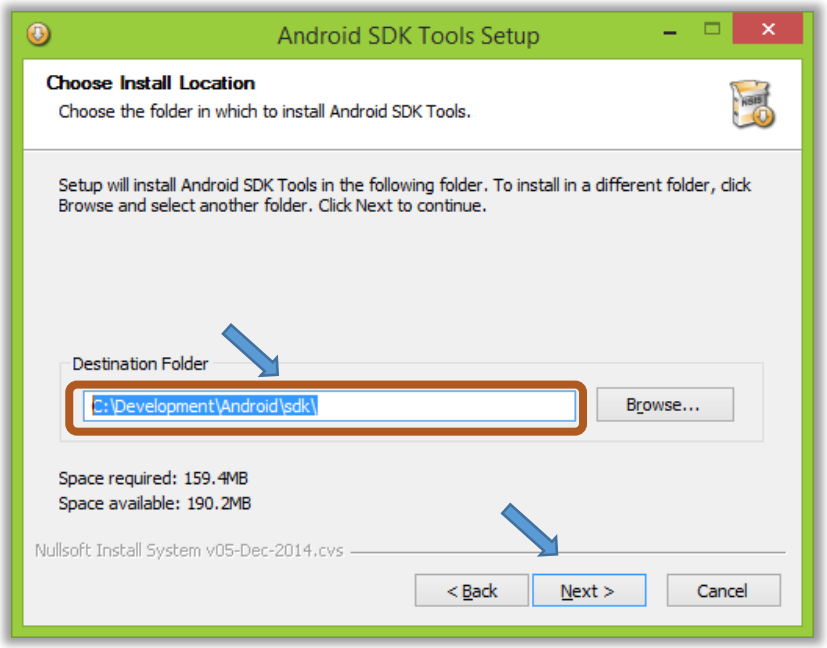
Other Download Options

SDK Tools Only

If you prefer to use a different IDE or run the tools from the command line or with build scripts, you can instead download the stand-alone Android SDK Tools. These packages provide the basic SDK tools for app development, without an IDE. Also see the SDK tools release notes.

| Platform | Package | Size | SHA-1 Checksum |
|---|---|---|---|
| Windows | installer_r24.1.2-windows.exe (Recommended) | 111364285 bytes | e0ec864efa0e7449db2d7ed069c03b1f4d36f0cd |
| | android-sdk_r24.1.2-windows.zip | 159778618 bytes | 704f6c874373b98e061fe2e7eb34f9fcb907a341 |
| Mac OS X | android-sdk_r24.1.2-macosx.zip | 89151287 bytes | 00e43ff1557e8cba7da53e4f64f3a34498048256 |
| Linux | android-sdk_r24.1.2-linux.tgz | 168121693 bytes | 68980e4a26cca0182abb1032abffbb72a1240c51 |

2. A continuación se procede a ejecutar el archivo descargado “installer_r24.1.2-windows.exe”, y se lo instala con sus opciones por defecto y cuando pregunte el directorio de instalación, se debe seleccionar el directorio de instalación del entorno de desarrollo “C:\Development\Android\sdk\”.

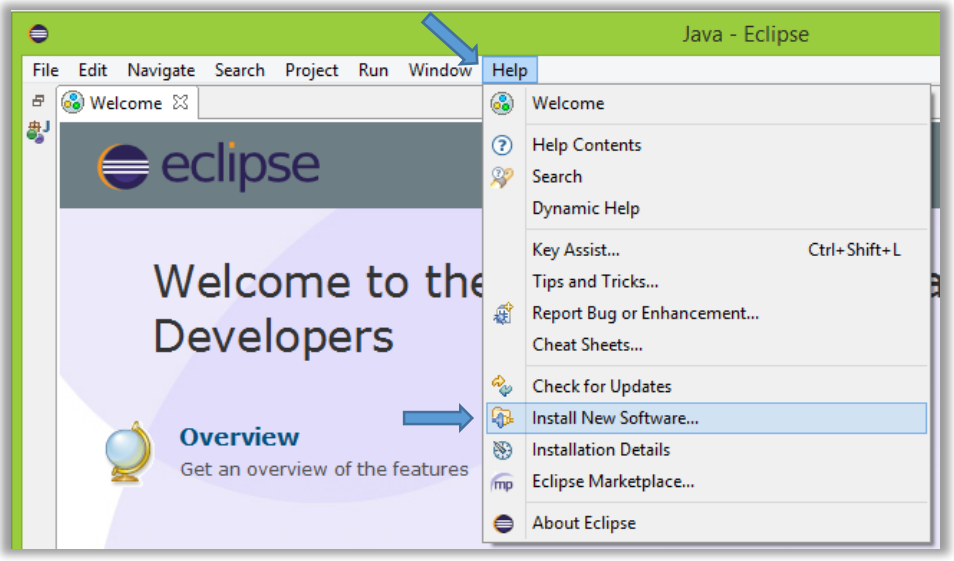



3. Se espera hasta que el proceso haya terminado y listo.

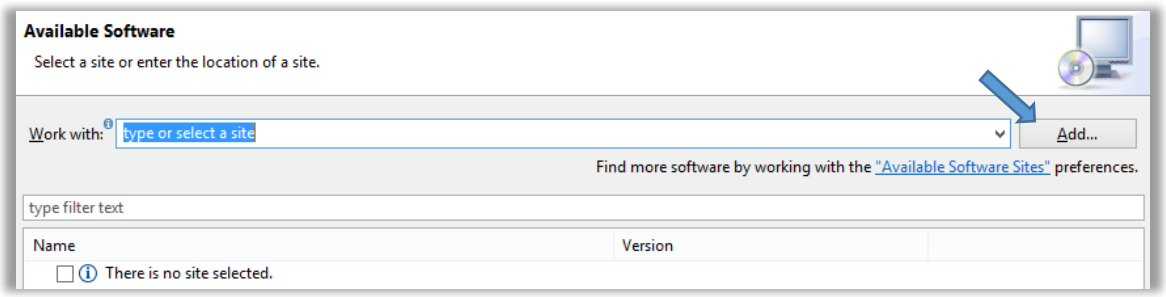
### 2.4.2. Instalación del ADT Plugin para Ecplipse.

Para la instalación del ADT Plugin se realiza lo siguiente:

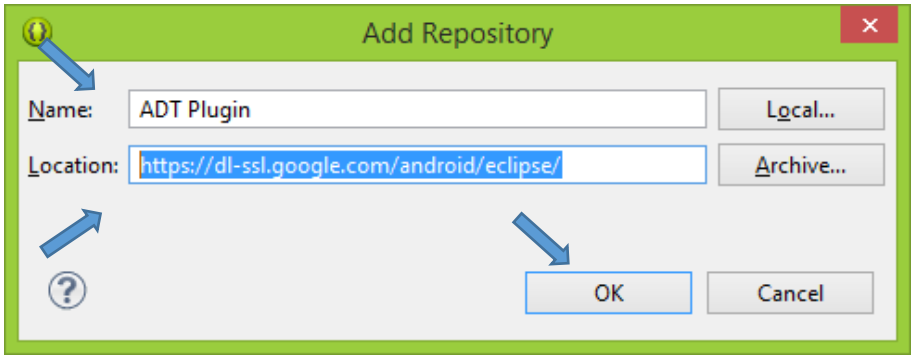
1. Se debe ejecutar el entorno de desarrollo de Eclipse ubicado en:
   - ✓ C:\Development\Android\eclipse\eclipse.exe
2. Una vez abierto Eclipse se selecciona "Help" en la barra de menú principal y posteriormente se debe seleccionar la opción "Install New Software".



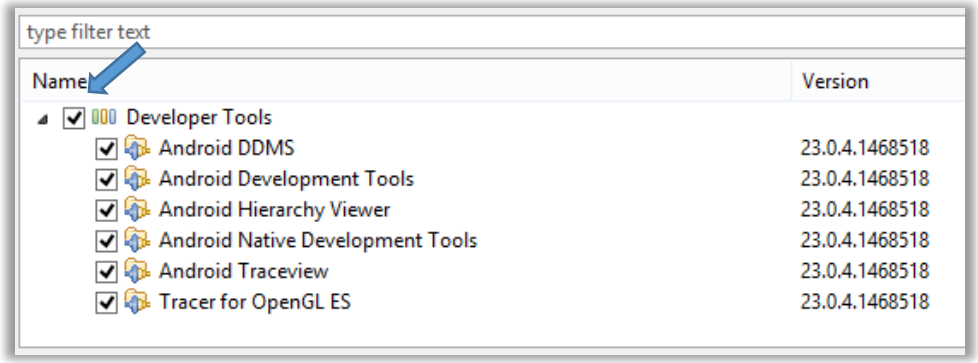
3. Posteriormente en la ventana emergente se selecciona la opción “Add”.



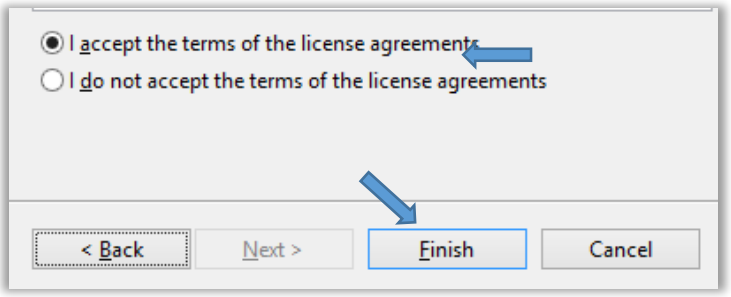

4. A continuación en la nueva ventana emergente se ingresa lo siguiente y se debe dar clic en el botón “OK”:
   - ✓ **Name:** ADT Plugin
   - ✓ **Location:** https://dl-ssl.google.com/android/eclipse/




5. Más adelante aparecerá una ventana de instalación donde se debe seleccionar el componente que se instalara “Developement Tools” y luego clic en el botón “Next”.




6. A continuación cuando nos pida aceptar los términos de la licencia se selecciona “I accept the terms of the license agreements”y se da clic en “Finish”.



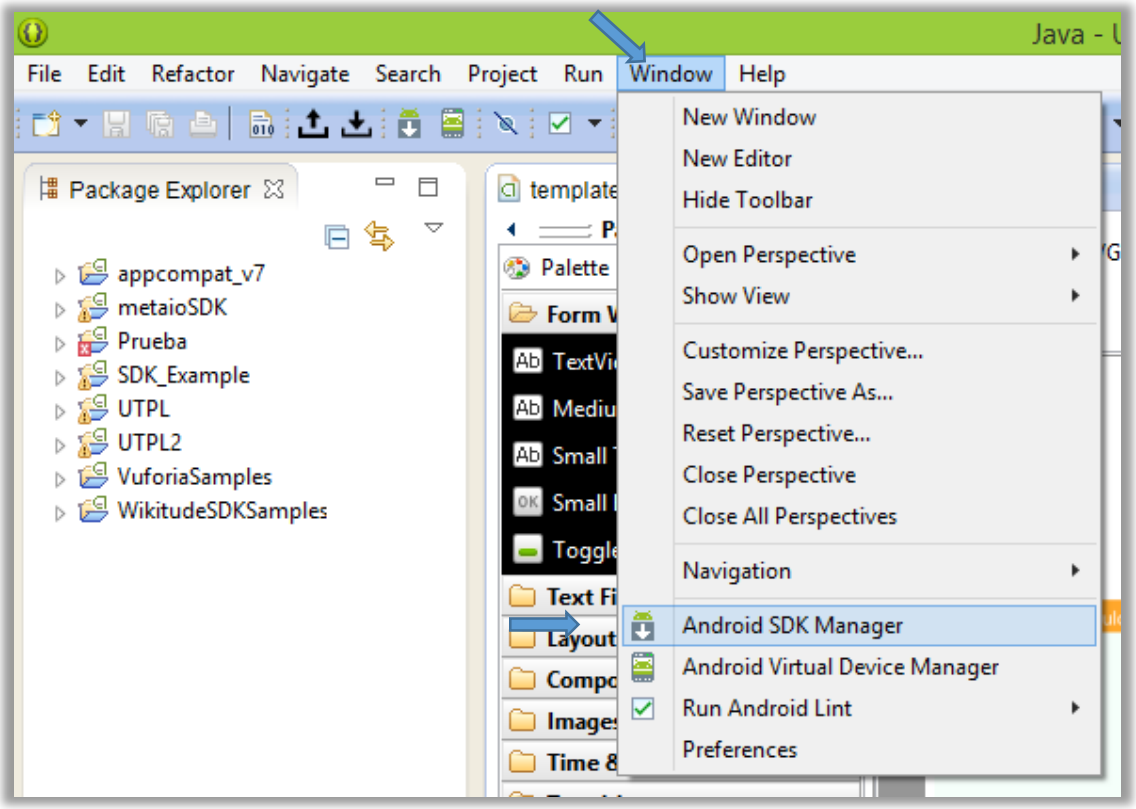

**Nota:** Si aparece algún mensaje de advertencia simplemente se lo acepta y se continúa normalmente con la instalación.

7. Una vez finalizado el proceso de instalación reiniciamos Eclipse y aparecerá la ventana de bienvenida del ADT plugin y continuación se debe seleccionar la opción “Use existing SDKs**”** y buscamos el directorio donde se instaló Android SDK que en nuestro caso es “C:\Development\Android\sdk\” (el resto de opciones se dejan las que aparecen por defecto) y posteriormente damos clic en el botón “Next” y por ultimo a “Finish”.

### 2.4.3. Configurar Android SDK Manager.

Si se ha instalado la última versión de Android SDK debería tener todos los componentes necesarios instalados, pero por obvias razones es recomendable asegurarse de que todos los componentes necesarios se encuentren instalados correctamente, para lo cual se realizara los siguientes pasos:

1. Ejecutamos Eclipse, el cual se encuentra en la siguiente ruta:
   - ✓ C:\Development\Android\eclipse\eclipse.exe
2. Posteriormente en la barra de menú se selecciona la opción “Windows” y posteriormente seleccionamos “Android SDK Manager”.




2. En la siguiente ventana emergente nos aseguramos de que se encuentre instalados o actualizados los siguientes componentes:

En Tools:

- ✓ Android SDK Platform-tools

En Android 4.4 (API 19):

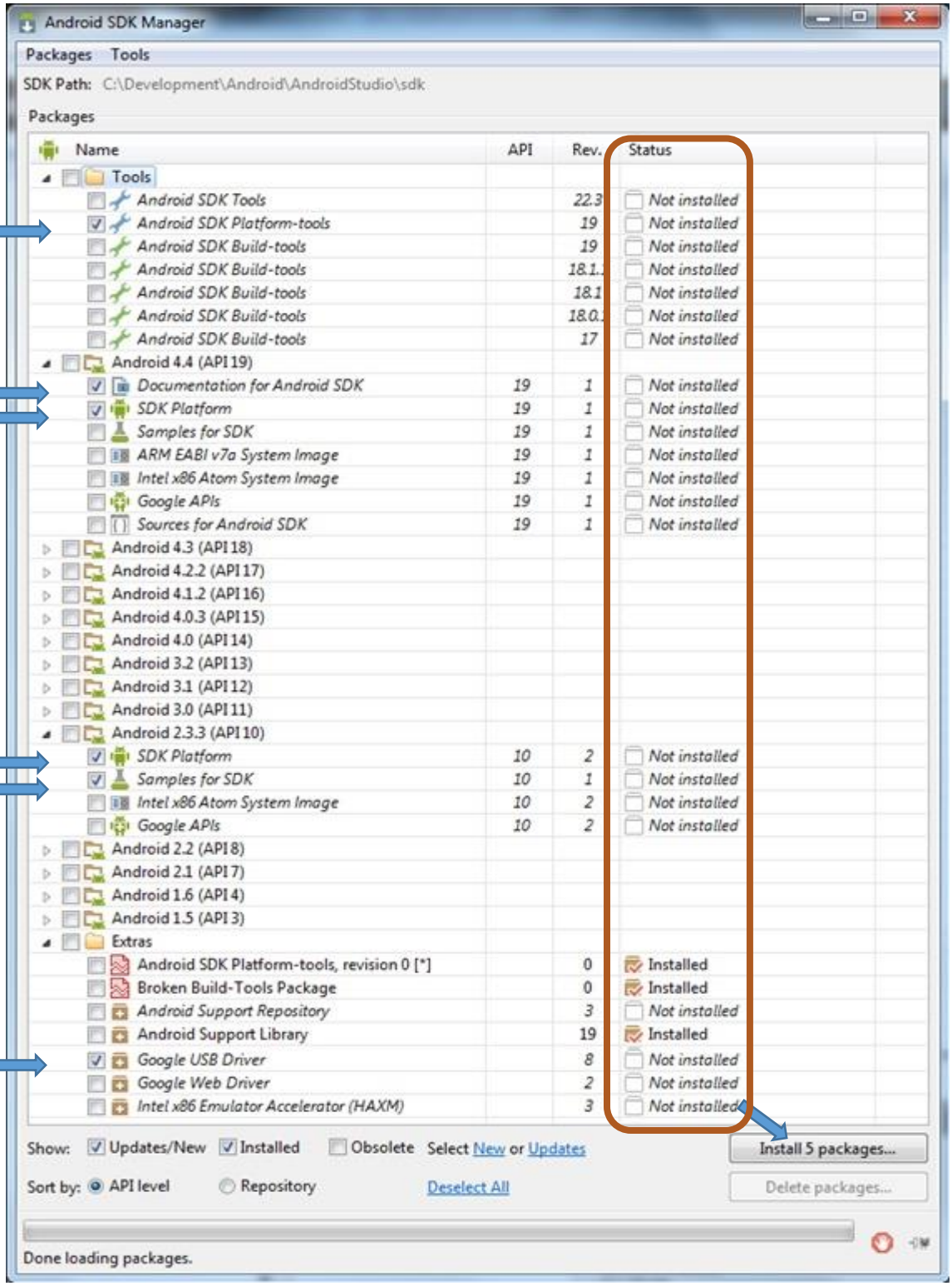
- ✓ Documentation for Android SDK
- ✓ SDK Platform

En Android 2.3.3 (API 10):

- ✓ SDK Platform
- ✓ Samples for SDK (opcional)

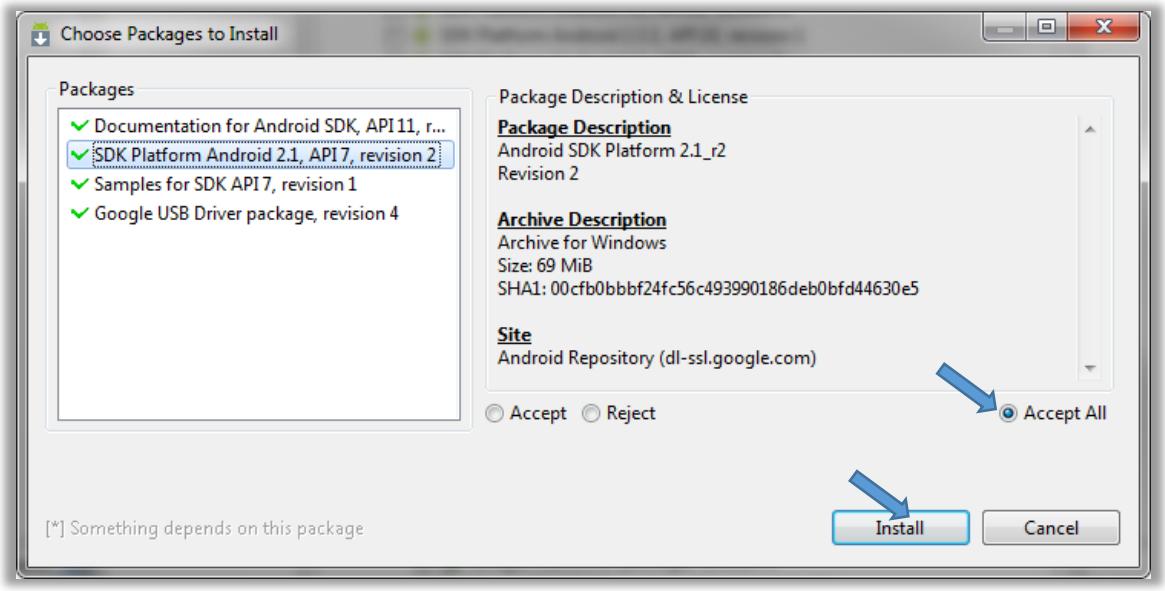
En Extras:

- ✓ Google USB Driver package




En caso de que algún componente no aparezca como instalado, se procede a instalarlos de la siguiente forma.

1. Marcamos todos los componentes faltantes y a continuación se da clic en el botón a “Install Packages”.
2. Posteriormente se debe dar un visto en las correspondientes licencias y se debe marcar la opción “Accept All” y por último se da clic al botón “install” tal y como se muestra en la siguiente imagen:




3. Se espera a que termine el proceso y listo

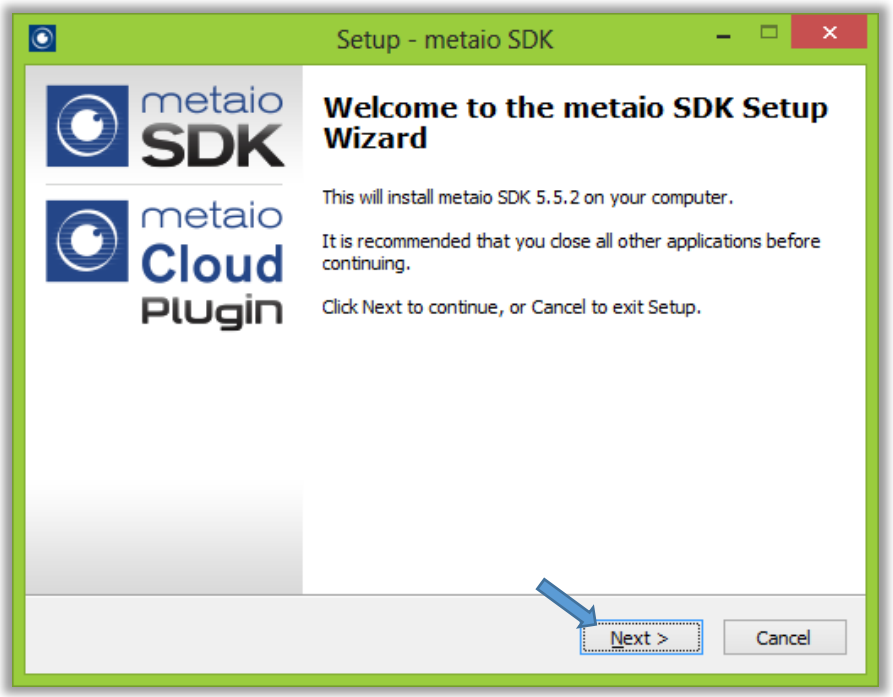
### 2.5. Instalación de Metaio SDK para Android.

A continuación se detallan los pasos necesarios para instalar Metaio SDK para Android:

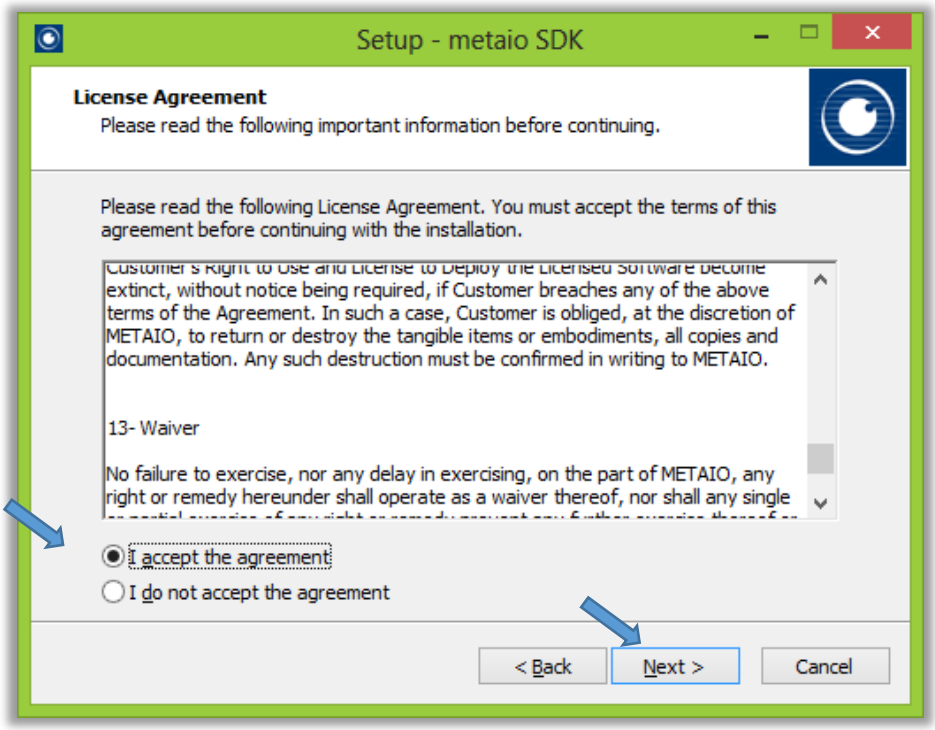
1. Primeramente se debe registrar en la web de Metaio, para luego proceder a descargar el SDK de Metaio para Windows desde el siguiente enlace:

   ✓ https://my.metaio.com/index.php

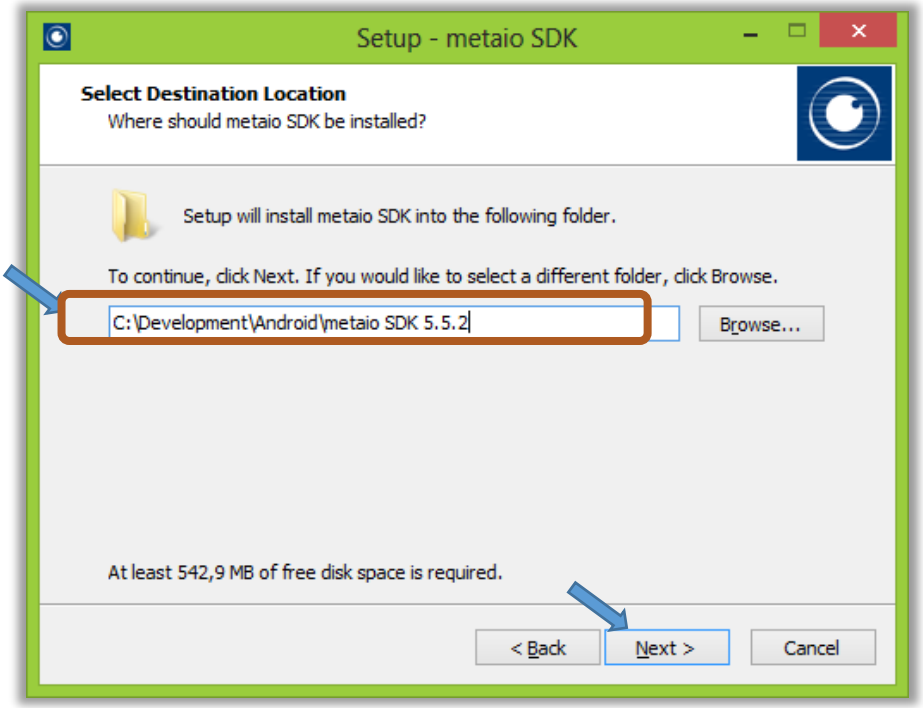

2. Seguidamente se ejecuta el archivo de instalación de Metaio y se procede a instalarlo, tal como se muestra a continuación.




3. Posteriormente se debe aceptar la respectiva licencia de Metaio, tal como se muestra a continuación:



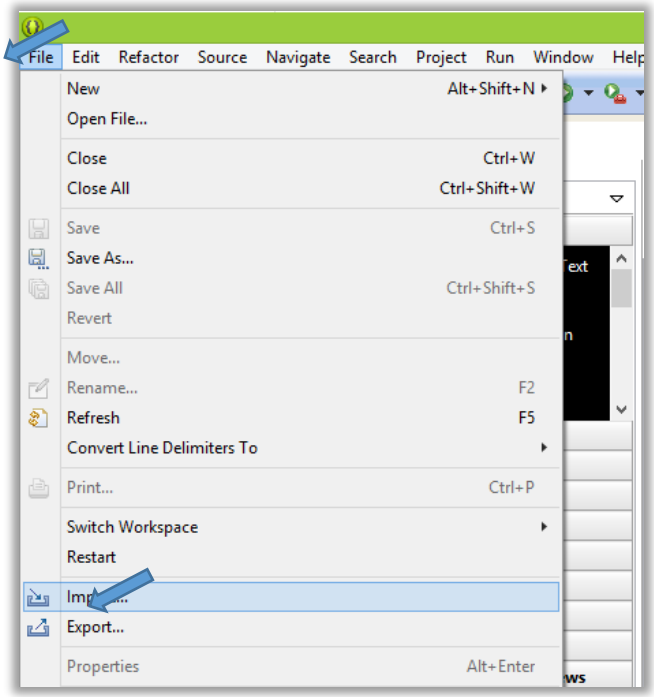

4. A continuación se debe seleccionar el directorio de instalación, en este caso se utilizara el directorio “C:\Development\Android\” y después a “Next”.



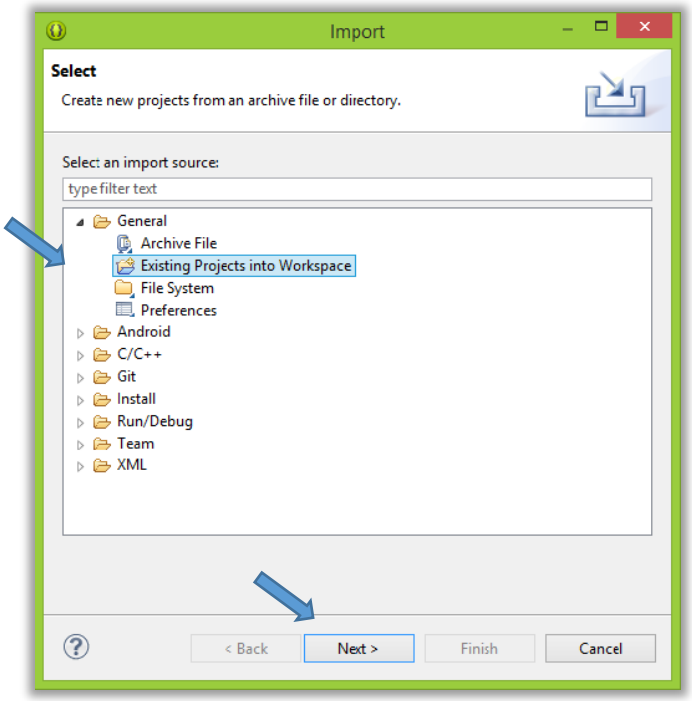
5. Por último se espera que termine la instalación y listo.

Una vez instalado Metaio SDK se procede a importar dicho SDK en el entorno de desarrollo de Eclipse, de la siguiente forma:

1. Ejecutamos Eclipse y en el barra de menú se selecciona “File” y después “Import” tal y como se muestra a continuación

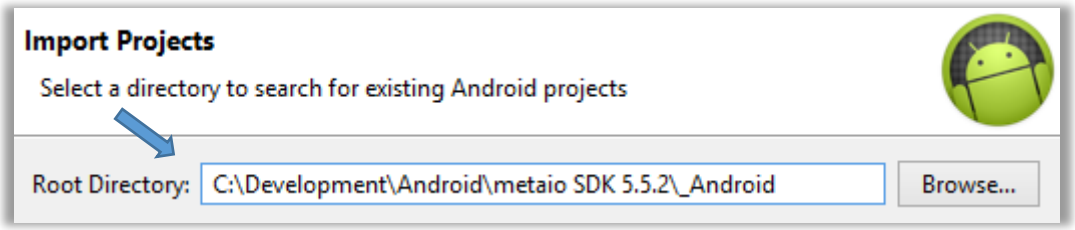



2. Posteriormente en la ventana emergente se selecciona “Existing Projects into Workspace” y después se da clic al botón “Next”.

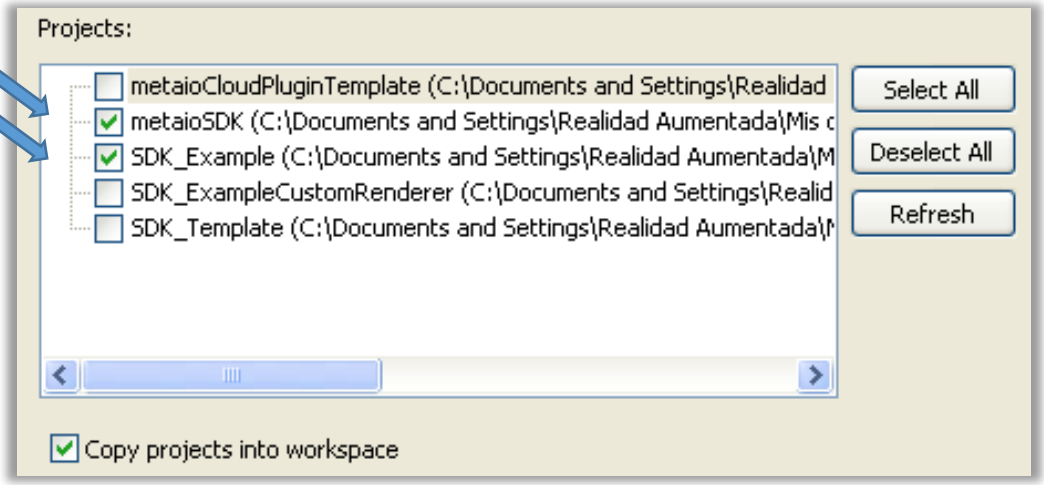



3. A continuación en la siguiente ventana se da clic en “Browse..” y buscamos la carpeta “_Android” que se encuentra en el directorio de instalación de Metaio SDK y damos a aceptar. En este caso es la siguiente:

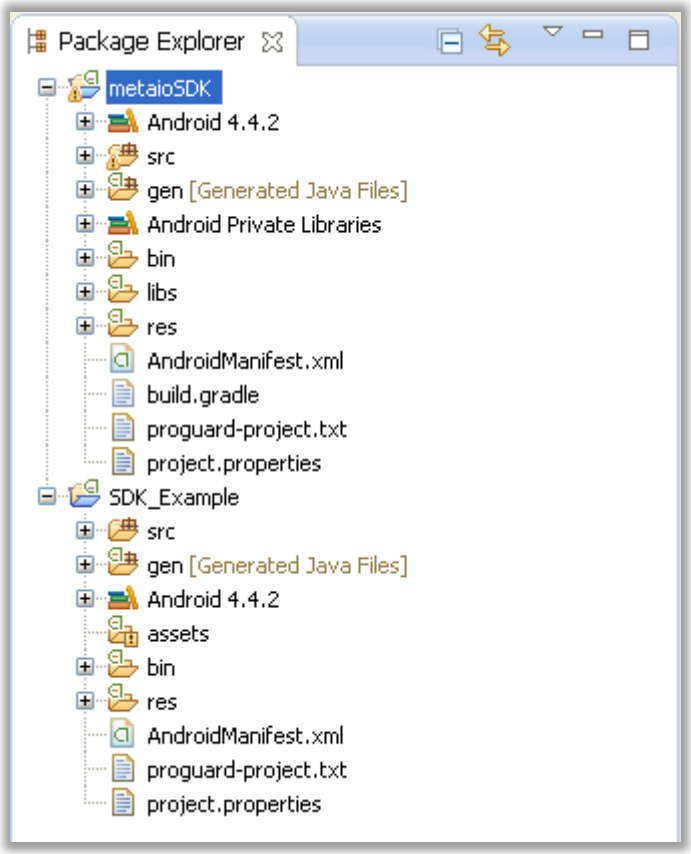
   ✓ C:\Development \Android\metaio SDK 5.5.2_Android



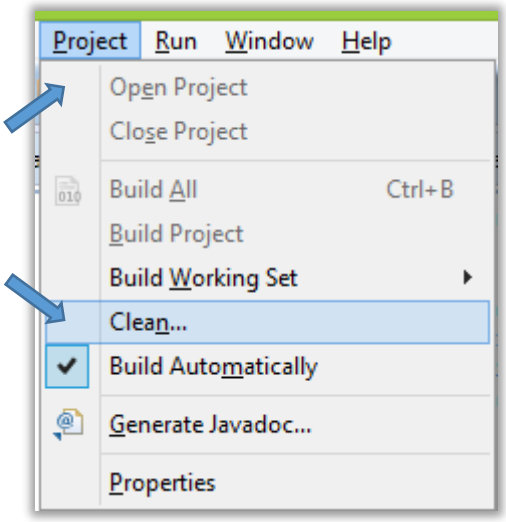

4. Posteriormente se selecciona los proyectos “MetaioSDK” y “SDK_Example” y marcamos la casilla “Copy projects into workspace” y luego se da clic al botón “Finalizar”.



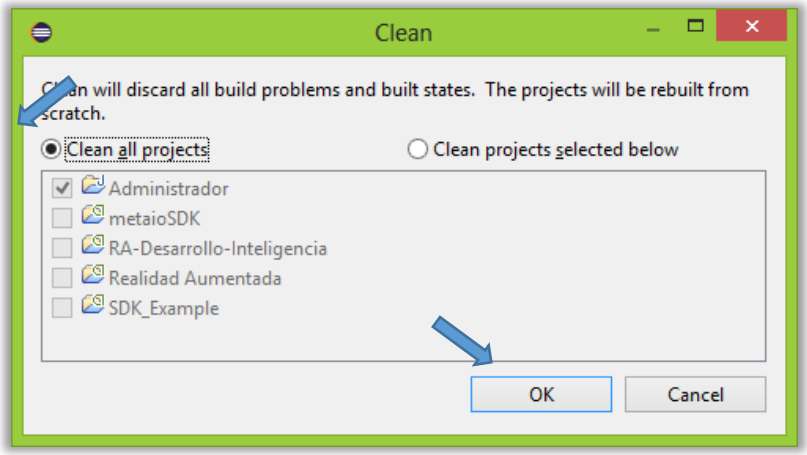

5. Posteriormente comprobamos que los proyectos se agregaron correctamente en eclipse, si todo salió correctamente debería mostrarse algo así en el explorador de paquetes de Eclipse:




6. Luego ejecutamos la opcion de “Clean…” que se encuentra en la barra de menu principal en la pestaña Project.




7. Por ultimo se selecciona la opcion “Clean all projects” co lo cual se actaulizaran todas las referencias de los proyectos cargados dentro del entorno de desarrollo Eclipse.

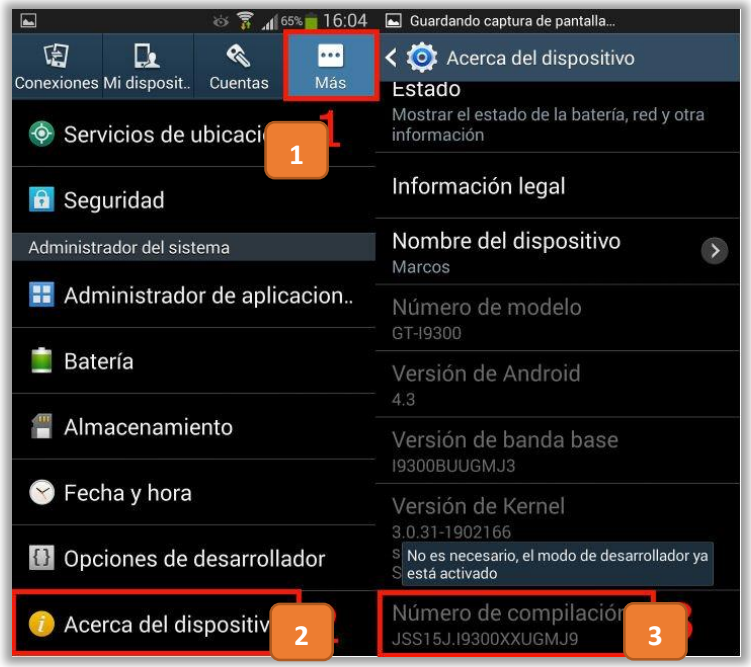



**Nota:** En caso de ocurrir algun error debemos asegurarnos que tenemos instalada la misma version de Metaio SDK o en su defecto utilizar los archivos de instalacion proporcionado con el presente manual.

### 2.6. Configuración del Dispositivo Móvil.

Para el correcto funcionamiento de las aplicaciones de RA se recomienda ejecutar las aplicaciones directamente en el dispositivo de destino y no en el emulador de Android. A continuación se muestra los pasos necesarios para configurar el dispositivo móvil para que trabaje conjuntamente con el entorno de desarrollo de Eclipse:

1. Primeramente se debe activar las opciones de desarrollador en el dispositivo, para lo cual se debe ir a Ajustes → Acerca del dispositivo y **pulsar siete veces sobre el número de compilación**. Una vez hecho nos saldrá el mensaje ¡Ahora eres un desarrollador! y aparecerá las “Opciones de desarrollador” en el menú ajustes.




2. Seguidamente se ingresa al menú “Opciones de desarrollador” y marcamos la casilla “depuración USB” en el dispositivo,

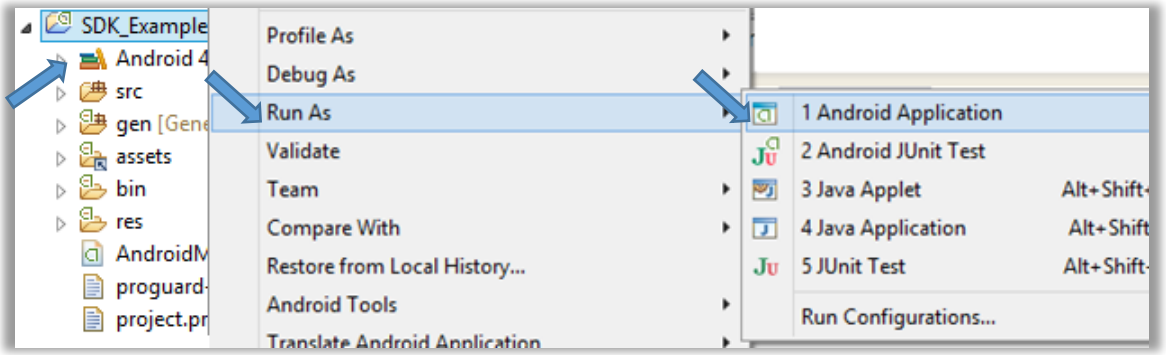
3. Por último se debe conectar el dispositivo Android al PC mediante el cable USB y se espera a que automáticamente se instalen los drivers necesarios para el dispositivo.

### 2.7. Compilación y Ejecución de ejemplos de Metaio SDK.

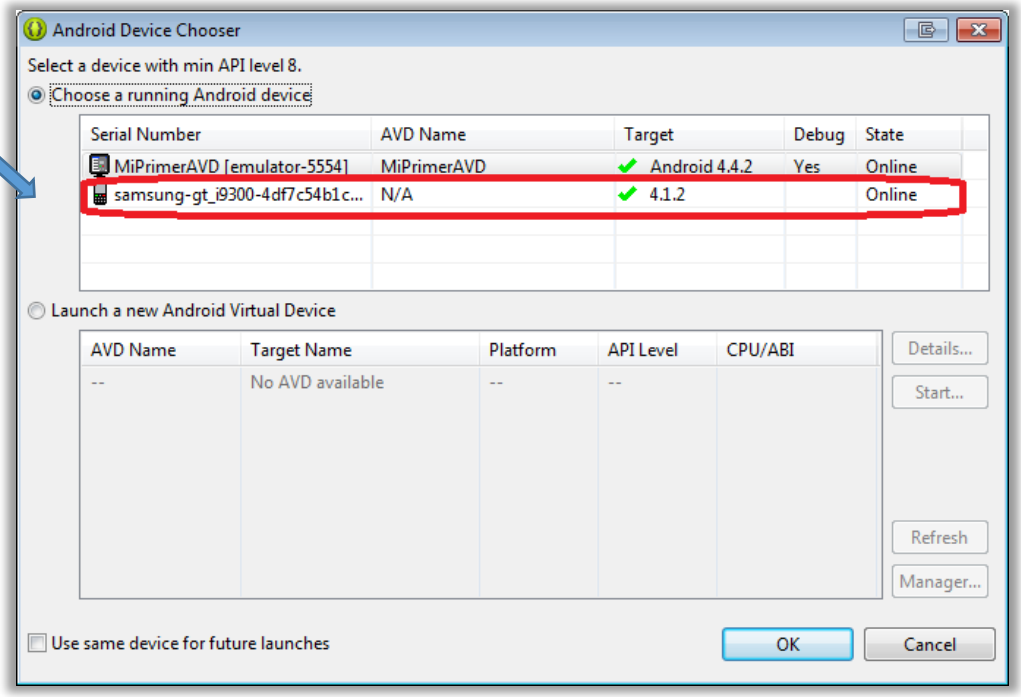
Una vez instalado y configurado Android development tool y Metaio SDK se procede a ejecutar el proyecto de ejemplo de Metaio SDK con el fin de comprobar que toda la instalación se ha realizado correctamente.

Con un dispositivo configurado en modo desarrollador y conectado al computador se procede a correr lo ejemplos de Metaio SDK para lo cual se realiza los siguientes pasos:

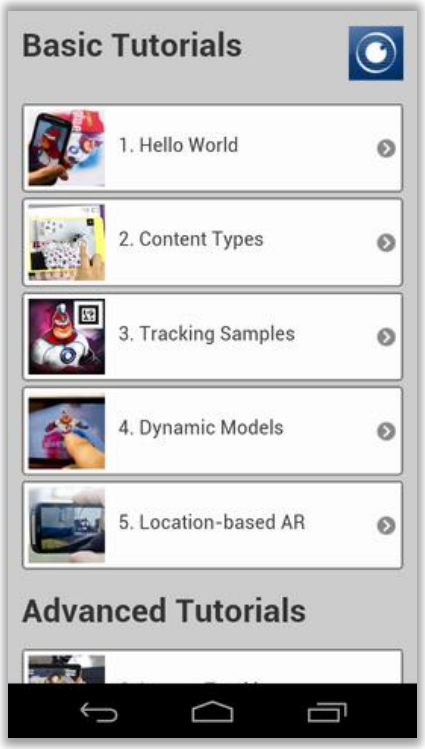
1. Se selecciona el proyecto SDK_Example en el explorador de paquetes y se da clic derecho para desplegar la opciones y a continuación se selecciona “Run As”, seguido de “Android Application”




2. Posteriormente se debe seleccionar el dispositivo Android de destino de entre la lista de dispositivos y luego a “OK”.




3. Si todo ha salido bien, la aplicación debería ejecutarse automáticamente en el dispositivo seleccionado y mostrar la siguiente pantalla de inicio:



## 3. ADMINISTRACION DE LA APLICACIÓN.

En el siguiente capítulo se detallara todas las funciones necesarias para adaptar la aplicación de RA del componente académico Desarrollo de la Inteligencia hacia cualquier otro componente académico con diferentes recursos. A continuación se detalla el conjunto de funciones que se abordara en este capítulo.

- ✓ Importación del proyecto de RA.
- ✓ Cambiar de Información de la Aplicación.
- ✓ Cambiar de Gráficos de la Aplicación.
- ✓ Cambio de los Objetos 3D y Audio.
- ✓ Cambios en el Código de la Aplicación.
- ✓ Exportación del Proyecto de RA.

### 3.1. Importación del proyecto de RA.

Para poder importar la carpeta del proyecto de RA al entorno de desarrollo de Eclipse se debe realizar los mismos pasos que se realizó para importar el SDK de Metaio, solo que esta vez, en lugar de indicar la ruta o directorio de instalación del SDK se procede a seleccionar la ruta en donde se encuentra almacenada la carpeta del proyecto de RA.

Adicionalmente se debe verificar si Metaio SDK está presente, caso contrario se debe importar nuevamente, ya que la aplicación de RA requiere que Metaio SDK se encuentre cargado en Eclipse para funcionar adecuadamente.

En este caso como se desea modificar la aplicación para adaptarla a otro componente académico, y con el objetivo de facilitar las tareas de administración de la aplicación de RA se ha creado un proyecto adicional listo para la modificación, el cual se encuentra limpio de archivos innecesarios y el código de programación se encuentra optimizado para su fácil edición. Por consiguiente si se desea crear una nueva aplicación de RA personalizada se debe utilizar este proyecto listo para la edición.

**Nota:** La carpeta del actual proyecto de RA y el proyecto listo para modificar se encuentra dentro del CD proporcionado conjuntamente con este manual.

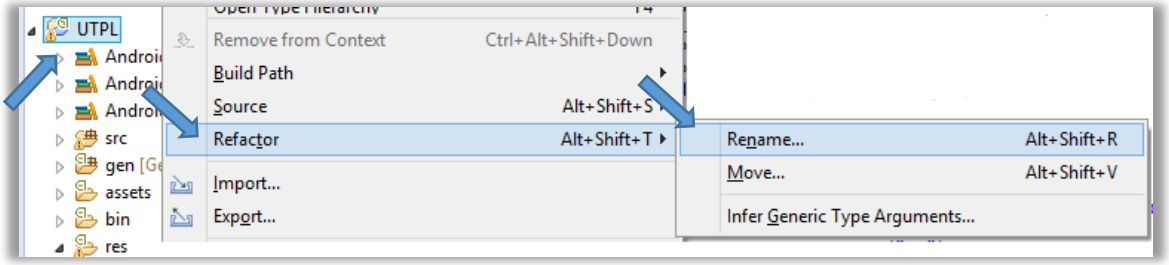
### 3.2. Cambio de Información de la Aplicación.

A continuación se describen los pasos necesarios para el cambio de información de la aplicación:

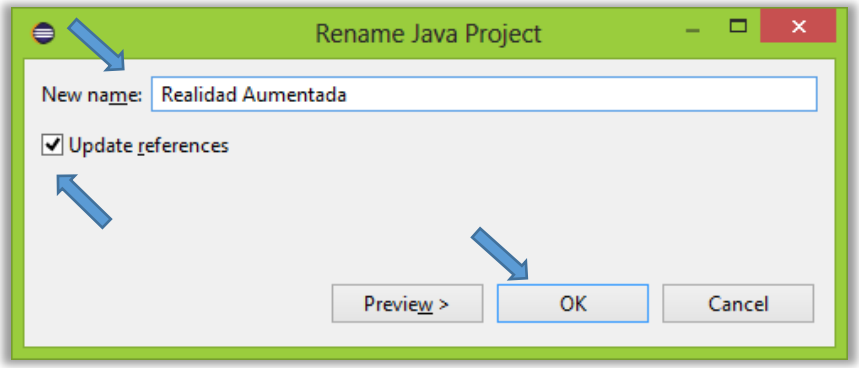
#### 3.2.1. Cambio de Nombre e ID de la aplicación.

Para realizar el cambio de nombre e ID de Aplicación (Nombre del Package) se debe continuar con los siguientes pasos:

1. Primeramente se debe ubicar la carpeta del proyecto en el explorador de paquetes y se da clic derecho sobre esta y a continuación se debe seleccionar la opción “Refactor” y posteriormente Rename.

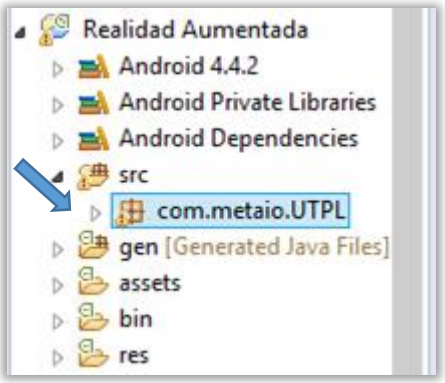



2. Posteriormente se presentara una ventana emergente donde se debe ingresar el nuevo nombre de la aplicación, y se debe asegurar que la opción “Update References” está marcada y luego se da clic en “OK”.




1. Por último se utiliza la opción “Clean…” que se encuentra en la barra de menú de la pestaña “Project” para actualizar las referencias, tal y como se hizo al importar el proyecto.

2. Para cambiar el ID de Aplicación (Package) se debe repetir los pasos del 1 al 3, solo que esta vez en lugar de seleccionar la carpeta del proyecto se selecciona el Package de la aplicación.

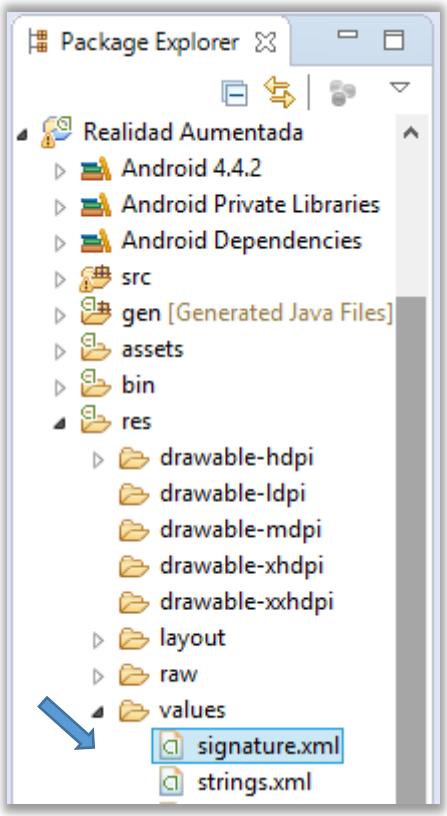



**Nota:** para el nombre del package generalmente se utiliza una nomenclatura que contiene la dirección de web inversa de la empresa más el nombre de la aplicación (Ejm: ec.com.example.RealidadAumentada).

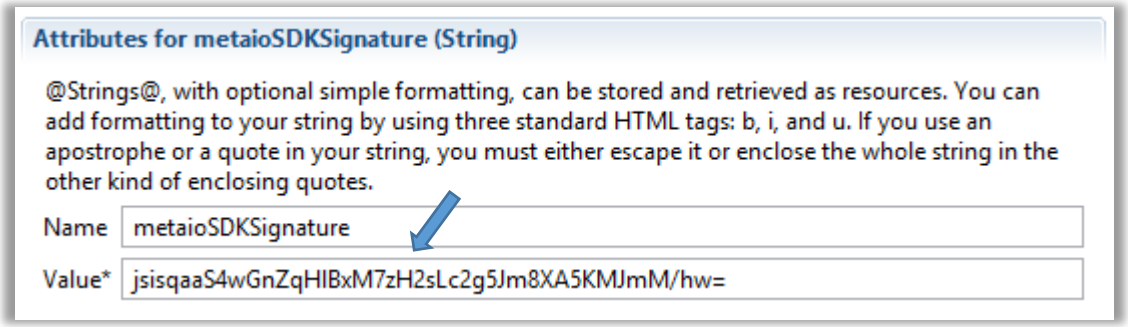
### 3.2.2. Cambiar el String de Licencia de Metaio.

Para cambiar el String de Licencia de Metaio se debe realizar lo siguiente:

1. Navegamos dentro del proyecto hasta el archivo XML llamado “Signature” y se procede a dar doble clic sobre este.
2. doble clic sobre este.



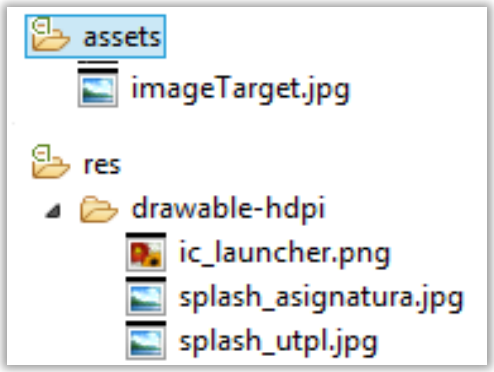

3. Posteriormente en el panel de a lado se cambia el String de licencia de Metaio por el nuevo String de licencia tal y como se muestra a continuación.




**Nota:** Para generar un nuevo String de Licencia Metaio remitirse al manual de Administración de la aplicación Administrador AR en la Sección 5.5

### 3.3. Cambio de gráficos de la Aplicación.

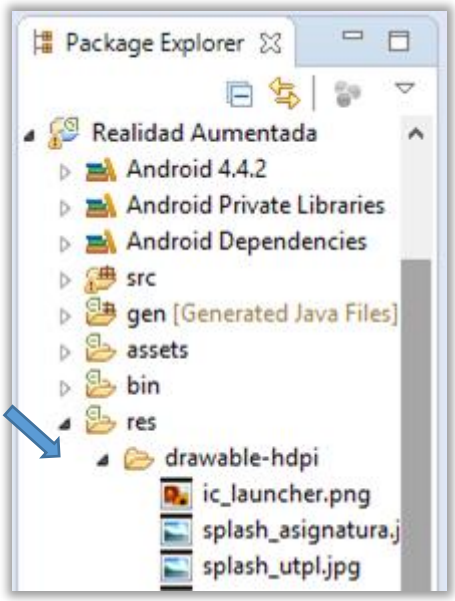
La aplicación de RA cuenta con 4 gráficos u imágenes en total, el icono de la aplicación, la tarjeta RA y 2 pantallas de carga que corresponde a los siguientes archivos. Donde:



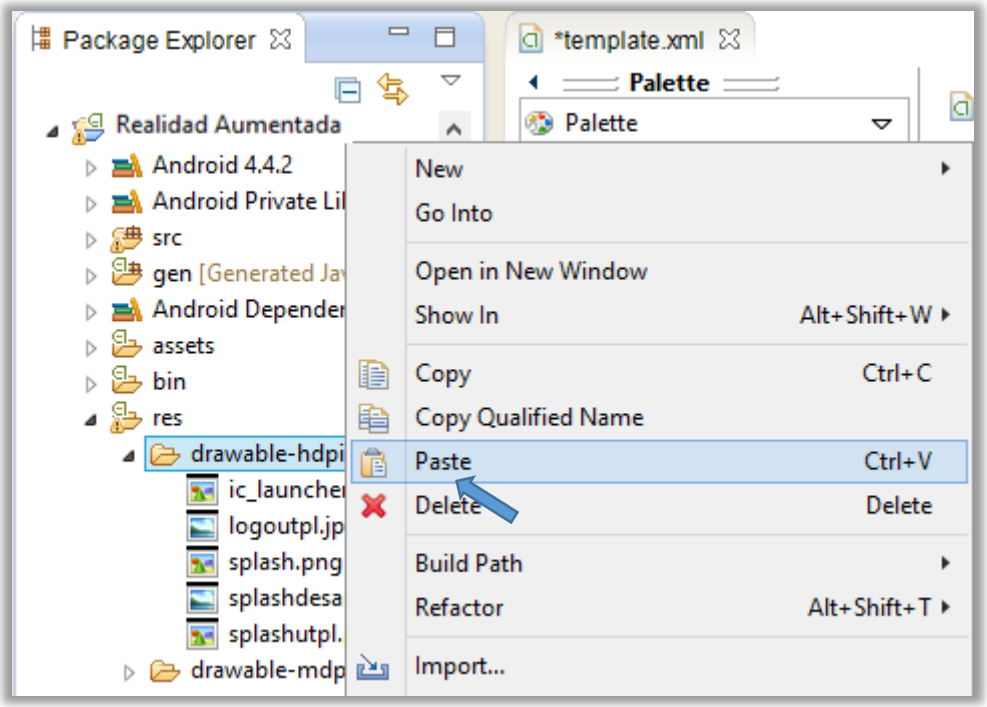

- ✓ imageTarget.jpg corresponde a la Tarjeta RA (240x240).
- ✓ **ic_launcher.png** corresponde al icono de la aplicación (72x72).
- ✓ **splash_utpl.jpg** corresponde a la pantalla de carga 1(800x320).
- ✓ **splash_asignatura.jpg** corresponde a la pantalla de carga 2 (800x320).

Los nuevos gráficos de la aplicación deben tener exactamente el mismo nombre con su mismo formato o extensión de archivo, adicionalmente deben tener una resolución mínima de 240x240 en el casado de la tarjeta RA, 72x72 pixeles para el icono y 800x320 pixeles para las pantallas de carga. Para cambiar estos gráficos dentro de la aplicación se realiza lo siguiente:

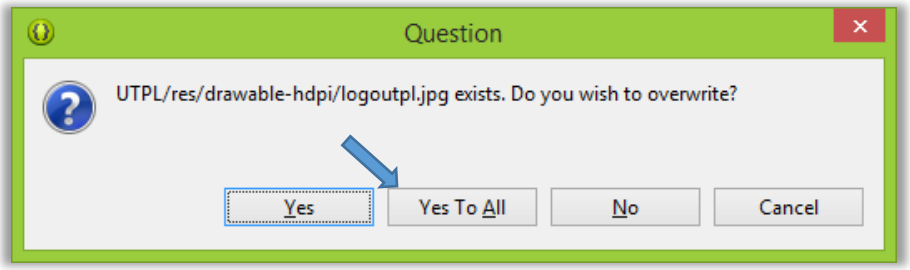
1. En el explorador de paquetes se debe localizar la carpeta del proyecto y la desagregamos luego ubicamos la carpeta “res” y dentro de esta la carpeta “drawable-hdpi”



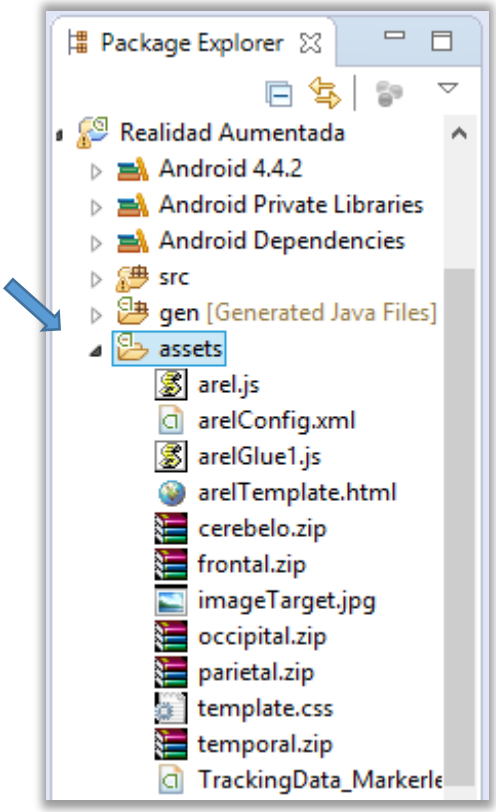

2. Ahora simplemente se copia los nuevos gráficos dando clic derecho en la carpeta “drawable-hdpi” y se seleccionando la opción “Paste” de la siguiente forma:



3. Aceptar el mensaje de sobrescribir y listo




Para cambiar la tarjeta de RA se debe realizar exactamente los mismos pasos, solo que en vez de copiarlos en la carpeta “drawable-hdpi” los copiamos en la carpeta “assets”.




**Nota:** No se debe olvidar que lo archivos deben tener exactamente el mismo nombre y extensión de archivo o sino la aplicación no será capaz de reconocerlos.

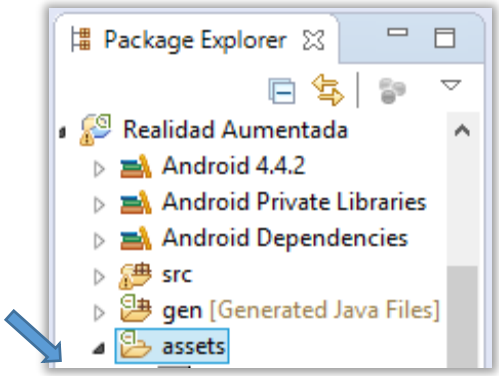
### 3.4. Cambio de los Objetos 3D y Audio.

Antes de realizar el cambio de los objetos 3D se debe realizar la respectiva preparación de la misma, para realizar esta tarea se debe remitir al manual de administración de la aplicación Administrador AR  en el sección 5, donde se encuentran los pasos a seguir.

Adicionalmente si se desea incluir archivos de audio personalizados estos deben de estar en formato mp3 y tener un bitrate máximo de 128kbps.

#### 3.4.1. Cambio de archivos de objetos 3D.

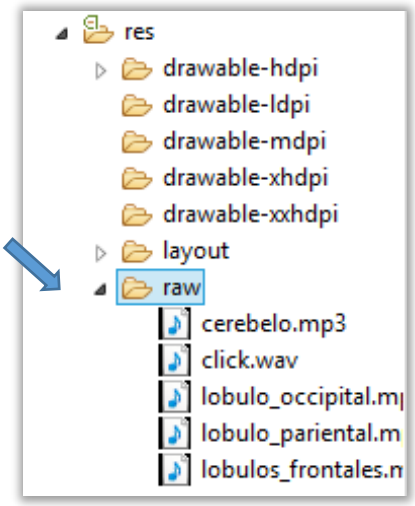
Con todos los objetos 3D listos para utilizarlos en la aplicación de RA simplemente se los debe copiar a la carpeta “assets” tal y como se copiaron los gráficos o imágenes.




**Nota**: Evite usar caracteres especiales o espacios en los nombres de archivos.

#### 3.4.2. Cambio de archivos de audio.

Para cambiar los archivos de audio de la aplicación de RA simplemente se los copia en la carpeta “raw” que se encuentra dentro de la carpeta “res” tal y como se ha estado copiando los distintos archivos.

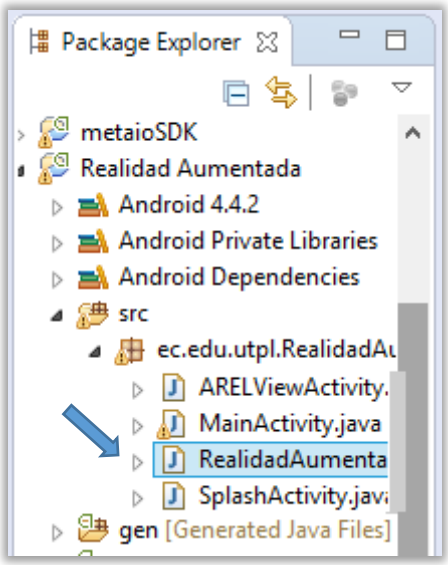



**Nota**: como en el caso anterior se recomienda no utilizar caracteres especiales o espacios en el nombre de los archivos ni letras mayúsculas.

### 3.5. Cambios en el Código de la Aplicación.

Una vez sé que se tiene todos los recursos necesarios cargados en el proyecto se debe realizar las siguientes modificaciones dentro del código de la aplicación de la siguiente forma:

1. Abrimos el proyecto llamado UTPL en el explorador de paquetes y localizamos el archivo UTPL.java:



2. Una vez abierto el archivo RealidadAumentada.java localizamos en el inicio el fragmento de código que se muestra a continuación.


3. Como se observa esta es la rutina de ingreso para el objeto número 1, pudiendo encontrar después de este, 3 rutinas de código más para poder ingresar un máximo de 4 objetos.

### 3.5.1. Descripción de las variables principales.

A continuación se explica cada una de las variables que intervienen en esta rutina de código.

- ✓ **strNomObj1 (String):** Nombre del objeto (este nombre se muestra en el botón de la interfaz).
- ✓ **strNomPartesObj1 (String[ ])**: Nombre de cada sub-objeto perteneciente al objeto principal.
- ✓ **strMensajeObj1 (String[ ])**: Contiene los mensajes de cada uno de los sub-objetos del objeto principal.
- ✓ **strRutasObj1 (String [ ])**: Contiene los nombres de cada uno de los archivos de los objetos 3D incluyendo su extensión.
- ✓ **strRutasAltObj1 (String[ ])**: Contiene los nombres de cada uno de los archivos de los objetos 3D alternativos incluyendo su extensión
- ✓ **rutasAudioObj1 (Integer [ ]):** Contiene los nombres de cada uno de los archivos de audio.

- ✓ **interactividadObj1 (boolean[ ]):** Determina si el sub-objeto es interactivo o no (“true” para interactivo y “false” para no interactivo).
- ✓ **activarObj1 (boolean):** Esta variable determina si el objeto 3D está activo dentro de la aplicación (“true” para activo y “false” para inactivo).
- ✓ **escalaObj1 (float):** Determina el tamaño del objeto mostrado en pantalla (normalmente no cambiar).
- ✓ **click2Obj1 (boolean):** Determina si la interactividad de los objetos es con 2 clic o 1 clic (“true” 2 clic, “false” 1 clic).
- ✓ **obj1, altObj1 y audObj1 (IGeometry, IGeometry, MediaPlayer):** Son variables auxiliares para la generación de los objetos (no se editan).

### 3.5.2. Modificaciones de código para ingreso de objetos.

A continuación mediante un ejemplo explicara los cambios a realizar para el ingreso del objeto “Cerebro” el cual consta de 2 partes Hemisferio Izquierdo y Hemisferio Derecho, cuyos archivos HemisferioIzq.zip, HemisferioDer.zip, audioizq.mp3 y objAltDer.zip ya han sido previamente copiados al proyecto. A continuación se detalla cada uno de los cambios que se realiza dentro del fragmento de código:

1. Primeramente se cambia el contenido de la variable **strNomObj1** por el nombre de del objeto, en este caso “Cerebro”.

```
87          private String strNomObj1 = "Cerebro";
```

2. Posteriormente se procede a cambiar el contenido arreglo **strNomPartesObj1** en el cual se pondrá los nombres reales de cada parte, es decir los nombres que se visualizara dentro de la aplicación en este caso Hemisferio Izquierdo y Hemisferio Derecho.

```
89          private String [] strNomPartesObj1 = {"Hemisferio Izquierdo", "Hemisferio Derecho"};
```

3. Seguidamente se realiza lo mismo con el arreglo **strMensajeObj1** en el cual se pondrá los mensajes que acompañaran a cada parte en el cuadro de mensaje desplegable, en ese caso serán “Mensaje Hemisferio Izquierdo” y “Mensaje Hemisferio Derecho” respectivamente.

```
91        private String [] strMensajeObj1 = {
92                "Mensaje Hemisferio Izquierdo",
93                "Mensaje Hemisferio Derecho"};
```

4. Luego se procede a cambiar el contenido del arreglo **strRutasObj1** y en su lugar se pone el nombre de los archivos que contienen los sub-objetos copiados previamente HemisferioIzq.zip y HemisferioDer.zip.

```
95        private String [] strRutasObj1 = {"HemisferioIzq.zip","HemisferioDer.zip"};
```

**5.** A continuación si el sub-objeto tiene un objeto alternativo se debe ingresarlo en la variable **strRutasAltObj1** con el nombre del archivo que contiene el objeto alternativo caso contrario se coloca null. En este caso como el sub-objeto “Hemisferio Izquierdo” no tiene objeto alternativo pondremos “null” y en el caso del “Hemisferio Derecho” que si tiene un objeto alternativo escribimos el nombre del archivo de dicho objeto “objAltDer.zip”.

```
97        private String [] strRutasAltObj1 = {null,"objAltDer.zip"};
```

**6.** **Más adelante** si el sub-objeto tiene un audio personalizado se debe ingresarlo en la variable **audioObj1** con el nombre del archivo que contiene el audio caso contrario se coloca  null. En este caso como el subobjeto “Hemisferio Izquierdo” tiene un audio alternativo se escribe el nombre del archivo de la siguiente forma anteponemos en prefijo “R.raw”+”nombre de archivo sin extensión” quedando de la siguiente forma “R.raw.audioizq”, y en el caso del “Hemisferio Derecho” que no tiene un audio personalizado se debe escribir “null”

```
99        private Integer [] rutasAudioObj1 = {R.raw.audioizq, null};
```

**7.** Posteriormente se debe cambiar el contenido de la variable interactividadObj1 para indicar si los subobjetos “Hemisferio Izquierdo” y “Hemisferio derecho” serán interactivos o no interactivos. En este caso como ambos objetos son interactivos colocamos “true” para ambos, en caso de que no fuesen interactivos se debería colocar false.

```
101       private boolean [] interactividadObj1 = {true,true};
```

8. A continuación se debe modificar la variable **activarObj1** con el valor de **“true”** para poner el objeto “Cerebro” como activo y esté se encuentre disponible dentro de la aplicación,

```
103         private boolean activarObj1 = true;
```

9. **Luego** se debe modificar la variable **escalaObj1** el cual es un valor decimal de tipo float que normalmente, se lo debe dejar en 1.0f mara que mantenga la misma escala que tuvo a nivel de diseño, en este ejemplo se lo modificara a 2.5f para escalarlo a un tamaño 2.5 veces más grande.

```
105         private float escalaObj1 = 2.5f;
```

10. Por último se debe modificar la variable click2Obj1 para indicar si la interactividad de los objetos es mediante 2 clics “true” o mediante un solo clic “false”, en este caso se pone "true”.

```
107         private boolean click2Obj1 = true;
```

**Nota:** Se debe tener especial cuidado en el nombre de los archivos, ya que un nombre mal escrito puede ocasionar que la aplicación no inicie adecuadamente.

A continuación se muestra el fragmento de código terminado con el cual se carga el objeto Cerebro y sus dos subobjetos Hemisferio Derecho y Hemisferio izquierdo.

*UTPL.java

```
84
85      // rutina paara objeto numero 1
86
87          private String strNomObj1 = "Cerebro";
88          private String [] strNomPartesObj1 = {"Hemisferio Izquierdo", "Hemisferio Derecho"};
89          private String [] strMensajeObj1 = {
90                  "Mensaje Hemisferio Izquierdo",
91                  "Mensaje Hemisferio Derecho"};
92          private String [] strRutasObj1 = {"HemisferioIzq.zip","HemisferioDer.zip"};
93          private String [] strRutasAltObj1 = {null,"objAltDer.zip"}; //null si no tiene subobj alternativo
94          private Integer [] rutasAudioObj1 = {R.raw.audioizq, null}; //null si no tiene audio personalizado
95          private boolean [] interactividadObj1 = {true,true};//true si el subobj es interactivo
96          private boolean activarObj1 = true; // cambiamos a true para activar el objeto
97          private float escalaObj1 = 2.5f; //escala de objetos
98          private boolean click2Obj1 = true; // true si intercatividad es de 2 click
99          private IGeometry [] obj1;          ///
100         private IGeometry [] altObj1;       /// variables de apoyo para el objeto
101         private MediaPlayer [] audioObj1;   ///
```

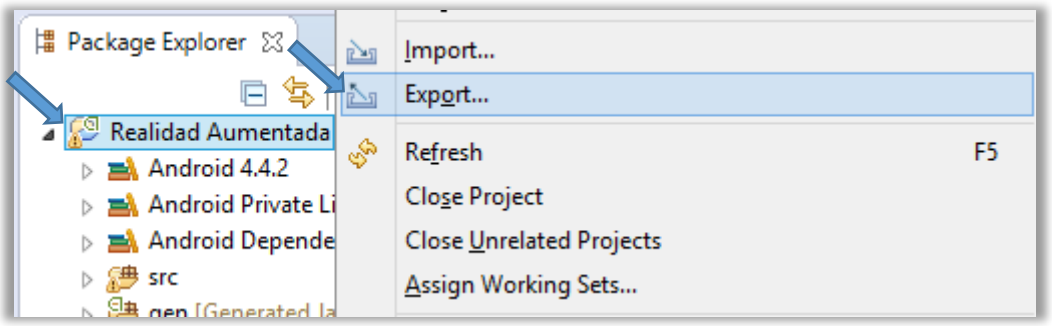
Recuerde que todos las rutas nombres y mensajes de las partes de los objetos deben estar en el mismo orden, es decir si se tiene un sub-objeto1 en la primera posición del arreglo en las demás variables esa misma posición corresponderá a ese subobjeto, nunca intercambiar el orden caso contrario puede ocasionar problemas dentro de la aplicación.

### 3.6. Exportación del Proyecto de RA.

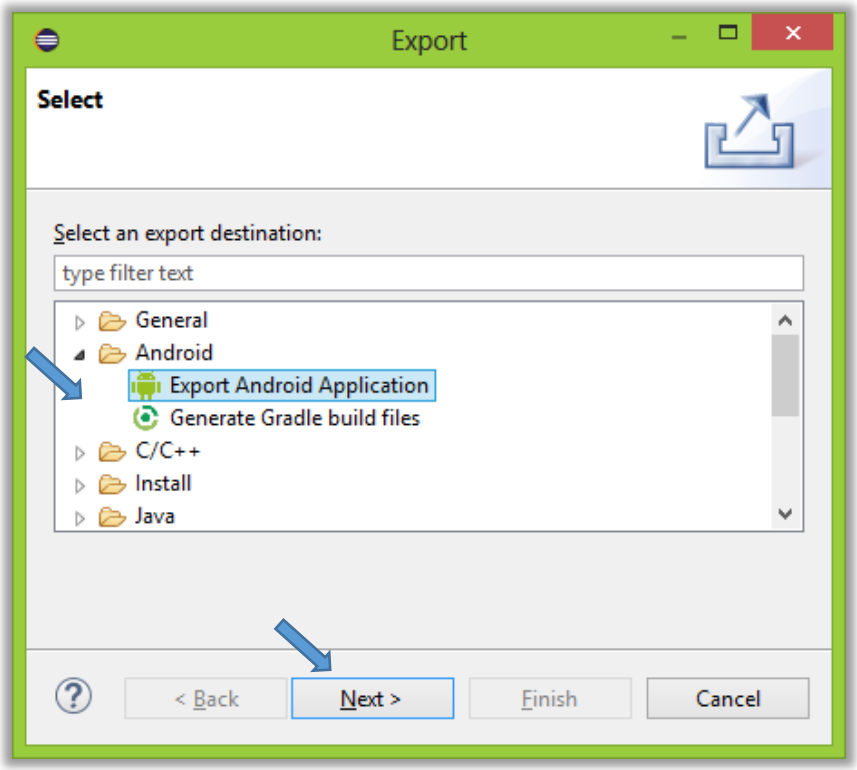
Una vez que se realiza los cambios pertinentes dentro de la aplicación de RA se procede a exportarlo como un archivo de instalación de Android, es decir, un “APK”. Así también se puede ejecutar la aplicación mediante un dispositivo conectado y configurado tal y como se explicó anteriormente en el presente manual.

A continuación se detalla los pasos a seguir para exportar el proyecto de Ecplise como un archivo de instalación Android:

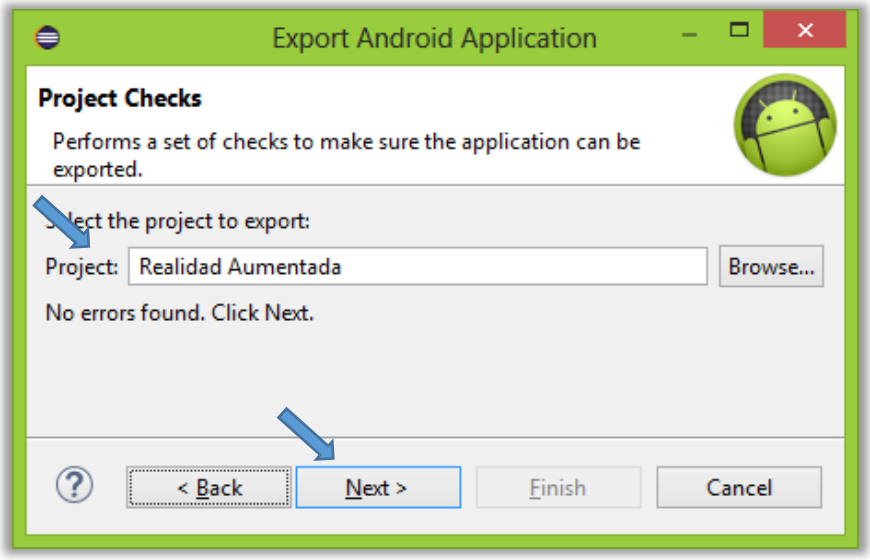
1. Primeramente se debe seleccionar el proyecto a exportar y posteriormente dando clic derecho sobre el mismo aparecerá un menú de opciones en el cual se debe escoger la opción “Export”



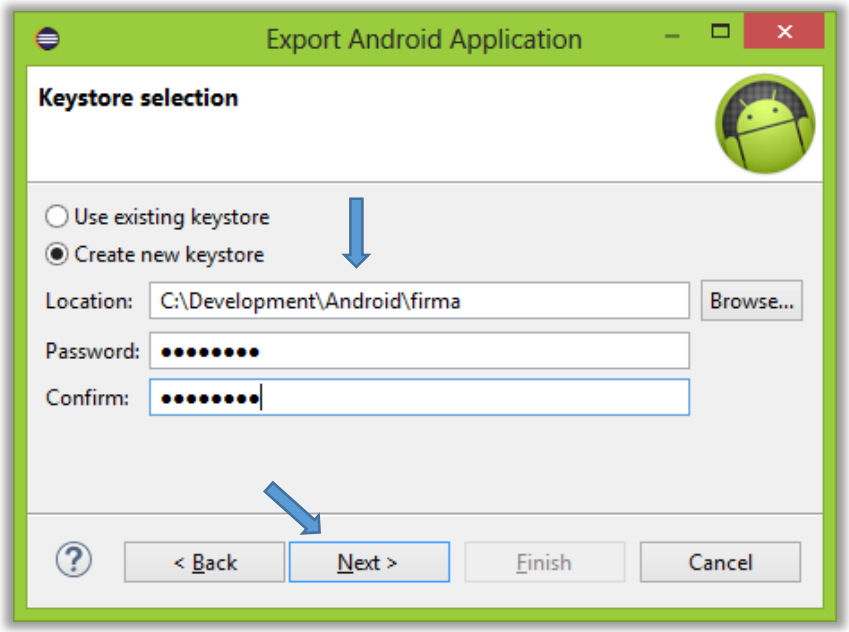

2. Posteriormente en la ventana emergente se selecciona “Export Android Application” y después se da clic al botón “Next”.




3. En la siguiente ventana aparecerá el nombre del proyecto se verifica que sea el correcto, y luego se dar clic en “Next”.

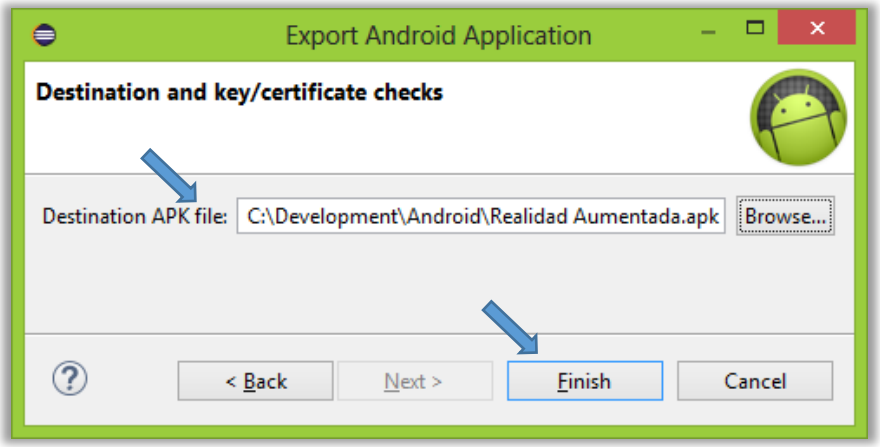



4. A continuación en la siguiente ventana se selecciona el archivo encriptado para firmar la aplicación, en este caso se creara un archivo de firma para lo cual se indica la ruta donde se guardara y se debe asignar una contraseña de al menos 8 caracteres y se procede a dar clic en “Next”.




5. Más adelante en la siguiente ventana se debe llenar los datos de la firma y se procede a dar clic en “Next”. Solo los 5 primeros campos son obligatorios.

6. Por ultimo en la siguiente ventana se selecciona el directorio o ruta donde se guardara el apk de la aplicación y después se selecciona “Finish” y con esto se ha finalizado la exportación de la aplicación a APK.




**Nota:** Una vez creado el archivo encriptado de la firma se puede reutilizar el mismo para firmar otras aplicaciones.

# MANUAL DEL USUARIO

## RA-Desarrollo-Inteligencia

### Descripción breve

El Manual de Usuario comprende la guía de instalación de la aplicación

RA-Desarrollo-Inteligencia, así también como su funcionamiento

Michael Freire
mfffreire@utpl.edu.ec

## TABLA DE CONTENIDOS

## 1. INTRODUCCION.

Para mayor compresión del usuario el presente manual se ha dividido en dos partes, la primera parte tiene la finalidad de guiar al usuario a través de la instalación de la aplicación en el dispositivo móvil, mientras que la segunda parte aborda la descripción de la interfaz gráfica y el funcionamiento de la aplicación.

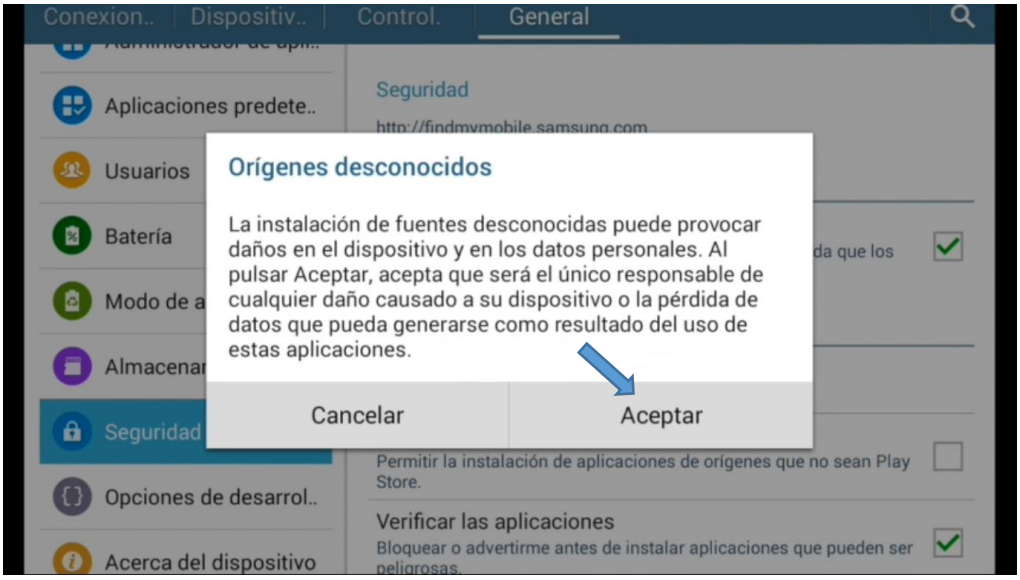
## 2. INSTALACIÓN DE LA APLICACIÓN.

Antes de empezar con la instalación de la aplicación se debe previamente configurar el dispositivo Android para lo cual se debe realizar lo siguiente:

### 2.1. Configuración de Orígenes Desconocidos.

1. Primeramente se debe activar los orígenes desconocidos, para lo cual se debe ir a Ajustes → Seguridad → Orígenes Desconocidos, en este menú se activa la casilla *Permitir la instalación de aplicaciones de orígenes que no sean Play Store.*

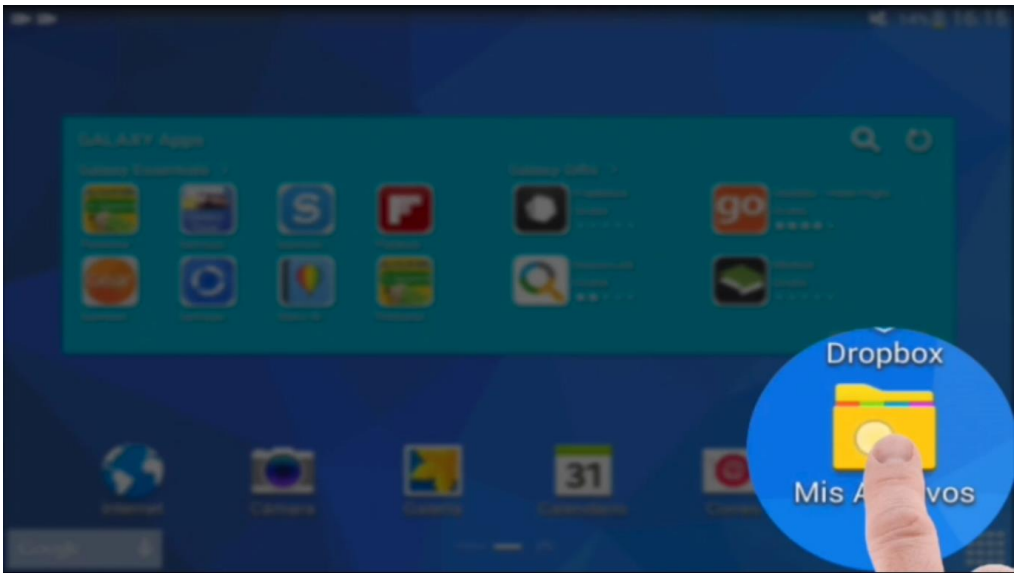

2. Posteriormente el sistema preguntará si desea confirmar esta opción, de clic en la opción aceptar para permitir la instalación de aplicación de orígenes desconocidos



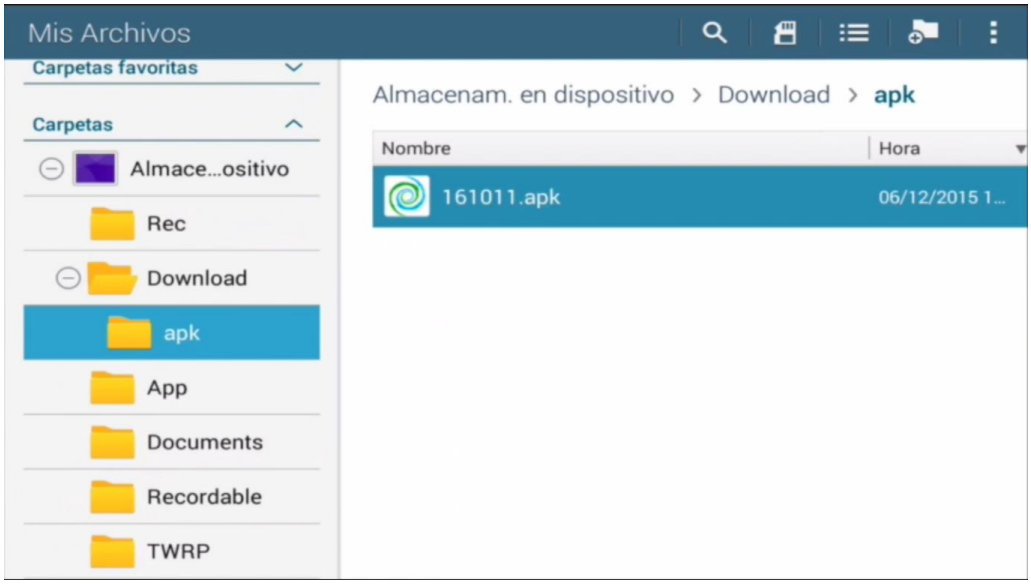

### 2.2. Instalación de RA-Desarrollo-Inteligencia.

A continuación se detallan los pasos necesarios para la instalación de la aplicación RA-Desarrollo-Inteligencia.

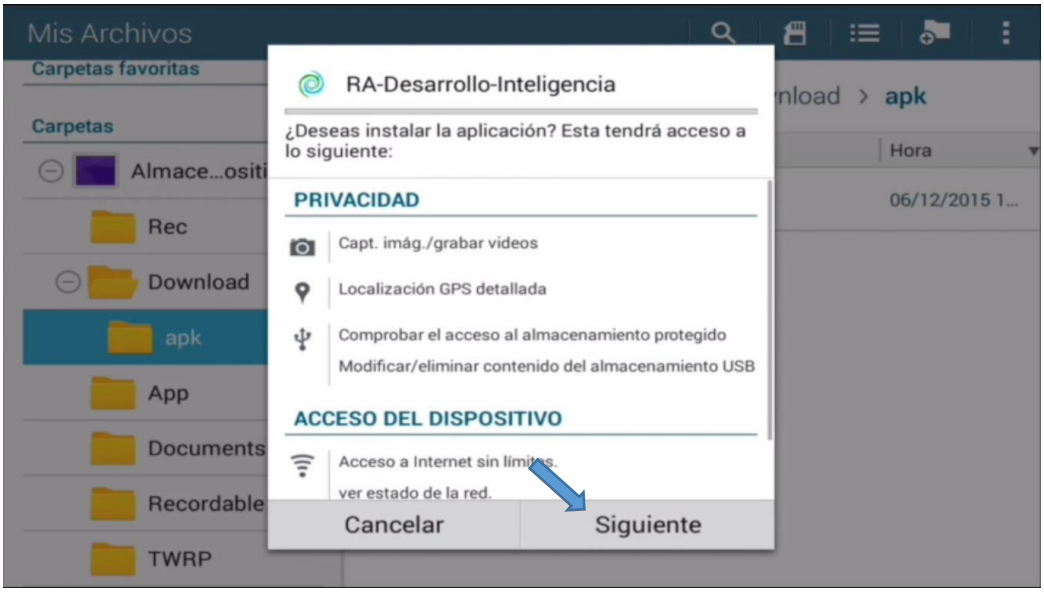
1. Una vez descargada la aplicación se la debe copiar a la Tablet Android para luego proceder a instalarla, en este caso la aplicación ha sido copiada a la ruta //Download/apk.
2. Posteriormente desde el dispositivo se ejecuta el explorador de archivos.



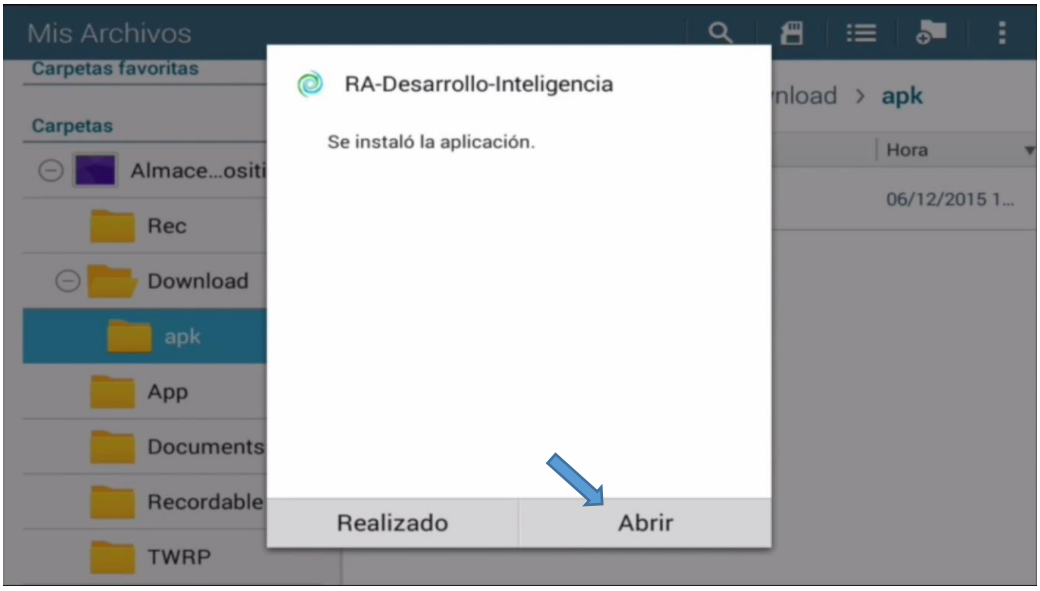
3. Más adelante se procede a buscar la carpeta donde se encuentra almacenada la aplicación de RA.



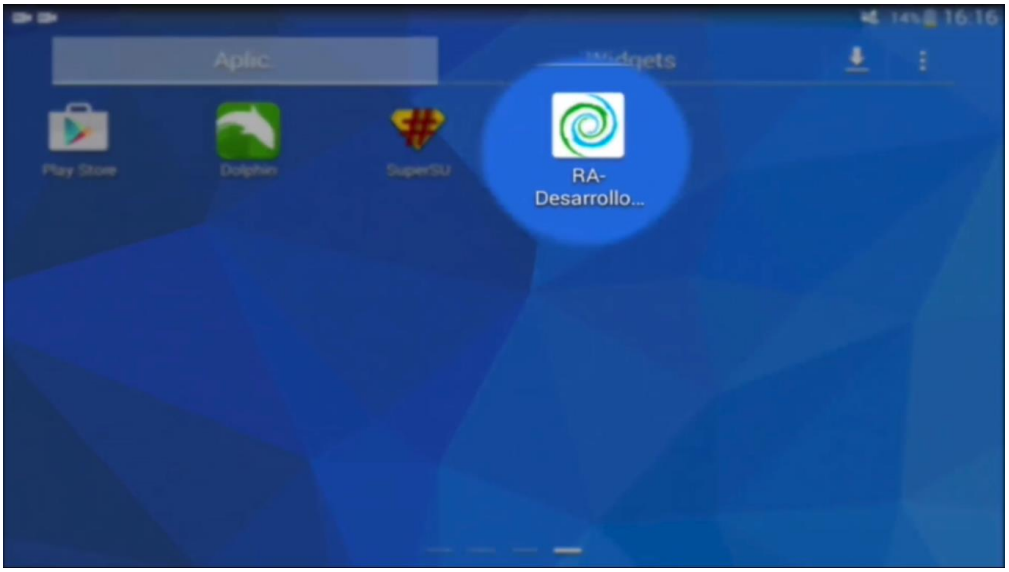

4. Una vez ubicada la aplicación, se debe dar clic sobre esta para iniciar su instalación, en la siguiente ventana se detalla la aplicación y los permisos que requiere para su ejecución, se da clic en aceptar y se espera a que finalice.




5. Una vez finalizada la instalación aparecerá una ventana preguntando si se desea abrir la aplicación, se da clic en abrir, y la aplicación de RA-Desarrollo-Inteligencia se ejecutara automáticamente.



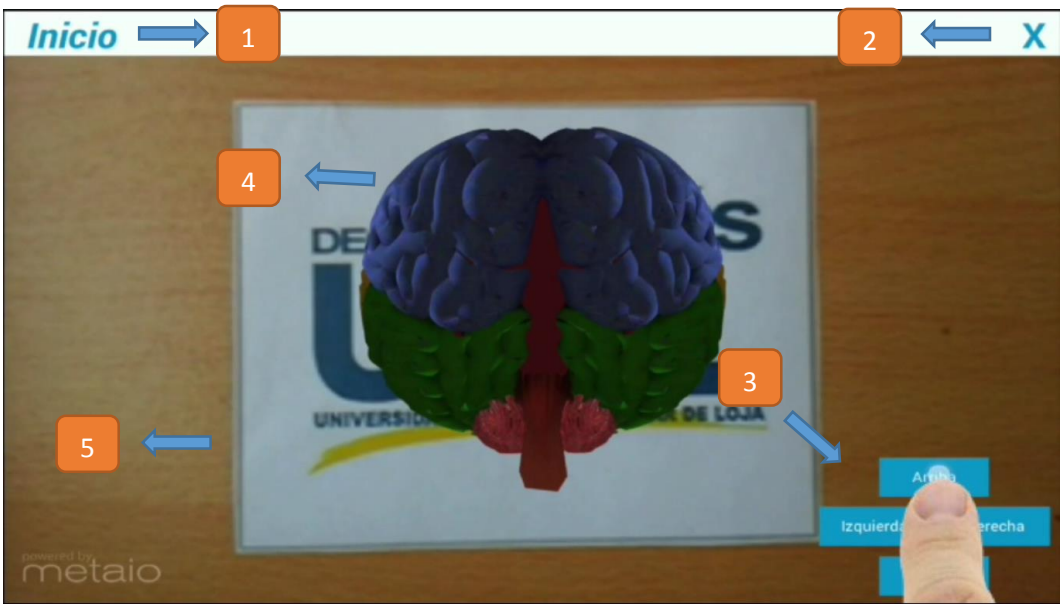

6. Una vez instalada la aplicación se la puede encontrar en la galería de aplicaciones.




## 3. DESCRIPCION DE LA INTERFAZ GRÁFICA DE USUARIO.

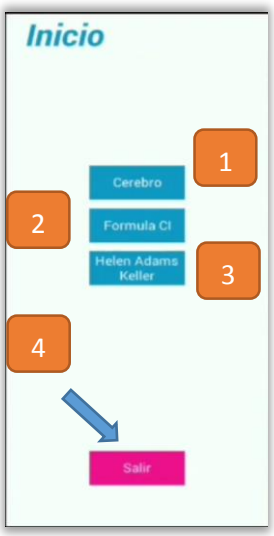
La aplicación de RA-Desarrollo-Inteligencia consta de una interfaz gráfica con una ventana principal, dentro de la cual se encuentran todos los controles necesarios para su ejecución e interacción.

### 3.1. Ventana Principal.




1. Botón de Inicio, al presionarlo se desplegara el conjunto de objetos disponibles para la asignatura Desarrollo de la Inteligencia.
2. Botón X, permite cerrar la aplicación.
3. Botones de rotación, permite rotar el objeto 3D mostrado en pantalla.
4. Objeto 3D de RA.
5. Marcador o tarjeta RA de la aplicación.

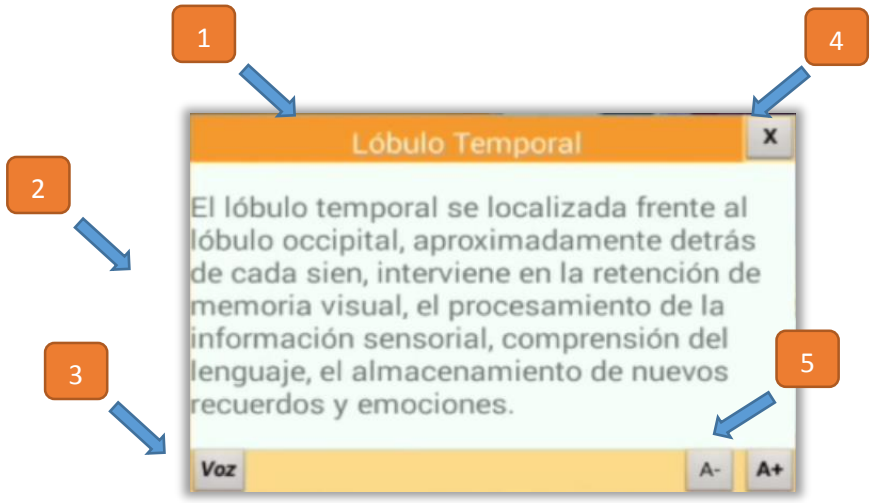
### 3.2. Menú Inicio.




1. Botón cerebro, permite cambiar el objeto 3D actual por el objeto 3D cerebro.
2. Botón Formula CI, permite cambiar el objeto 3D actual por el objeto 3D Formula CI.

3. Botón Helen Adams Keller, permite cambiar el objeto 3D actual por el objeto 3D Helen Adams Keller.
4. Botón salir, permite salir de la aplicación.

### 3.3. Cuadro de Mensaje Desplegable.

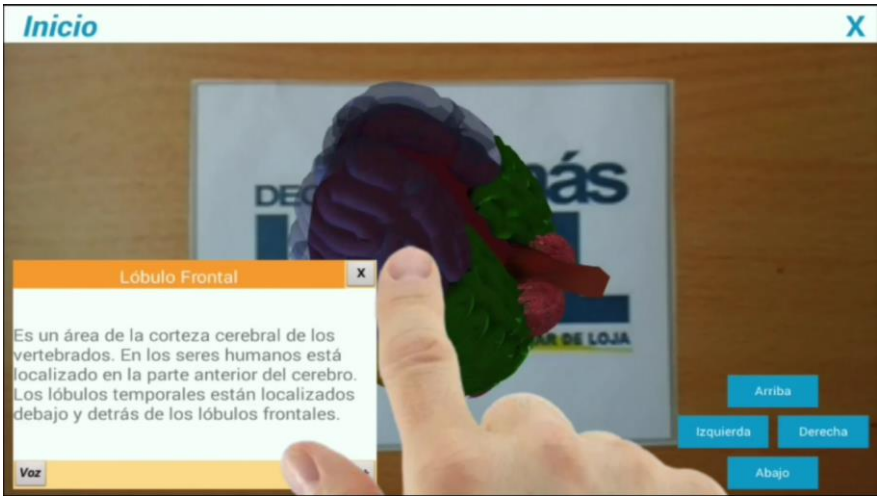



1. Nombre del Objeto seleccionado.
2. Mensaje desplegable del objeto seleccionado.
3. Botón Voz, que permite reproducir el audio correspondiente a dicho objeto.
4. Botón X, permite cerrar el cuadro de mensaje desplegable.
5. Botón A- A+, permiten aumentar o disminuir el tamaño de letra del mensaje desplegable.

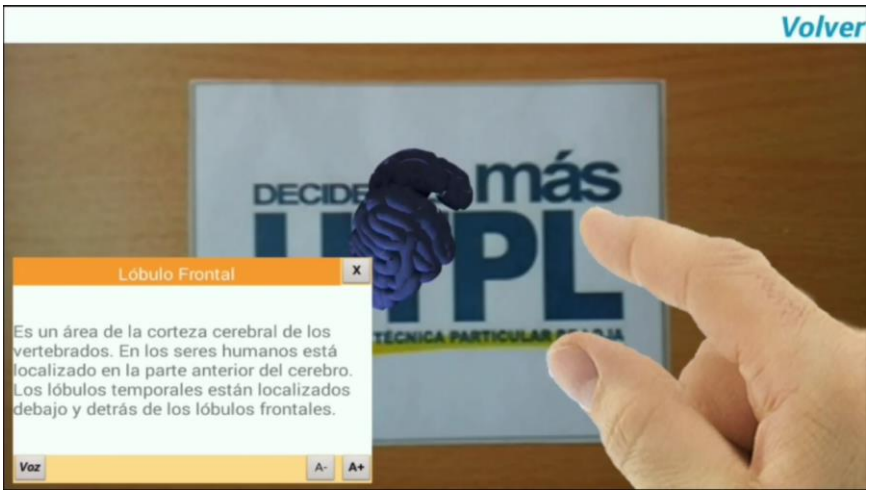
## 4. FUNCIONALIDADES.

Adicionalmente a los botones dispuestos en pantalla para la interacción, la aplicación de Ra-Desarrollo-Inteligencia permite mediante la utilización de gestos táctiles la interacción con los objetos 3D mostrados en pantalla.

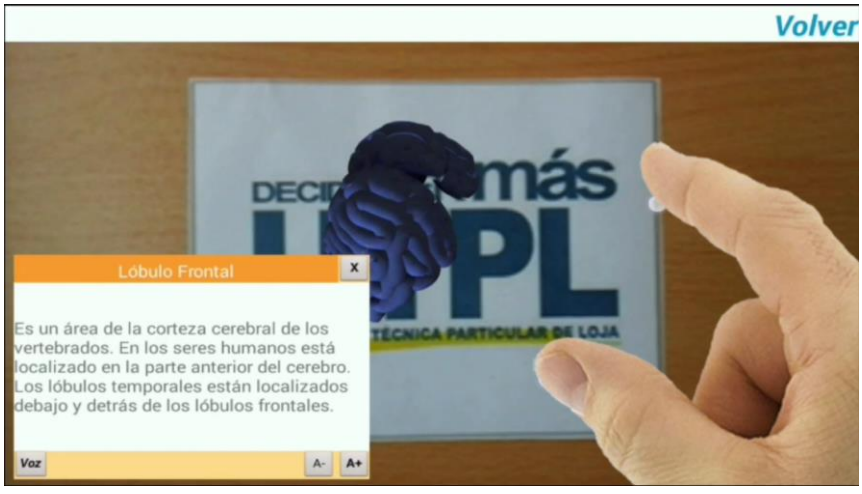
1. Se puede Interactuar con los objetos 3D dando clic sobre estos, lo que provocará que aparezca en pantalla el cuadro de mensaje desplegable con información útil sobre el mismo.




2. Se puede hacer Zoom+ y Zoom- con el gesto de pellizco.




3. Adicionalmente también se puede girar el objeto 3D con el gesto de compás.



# MANUAL DEL PROGRAMADOR

## Administrador AR

El siguiente manual tiene la finalidad de guiar a través de las principales rutinas de código empleadas en el desarrollo del Administrador AR

Michael Freire
mfffreire@utpl.edu.ec

**Introducción.**

El presente Manual de Programador tiene la finalidad de guiar al programador o administrador a través del flujo básico de información dentro de la aplicación de Administrador AR, en el mismo se describe detalladamente las principales clases, funcionalidades y atributos utilizados para la implementación de la aplicación, así mismo se describe como realizar ciertas tareas administrativas dentro de la aplicación.

**Plataforma.**

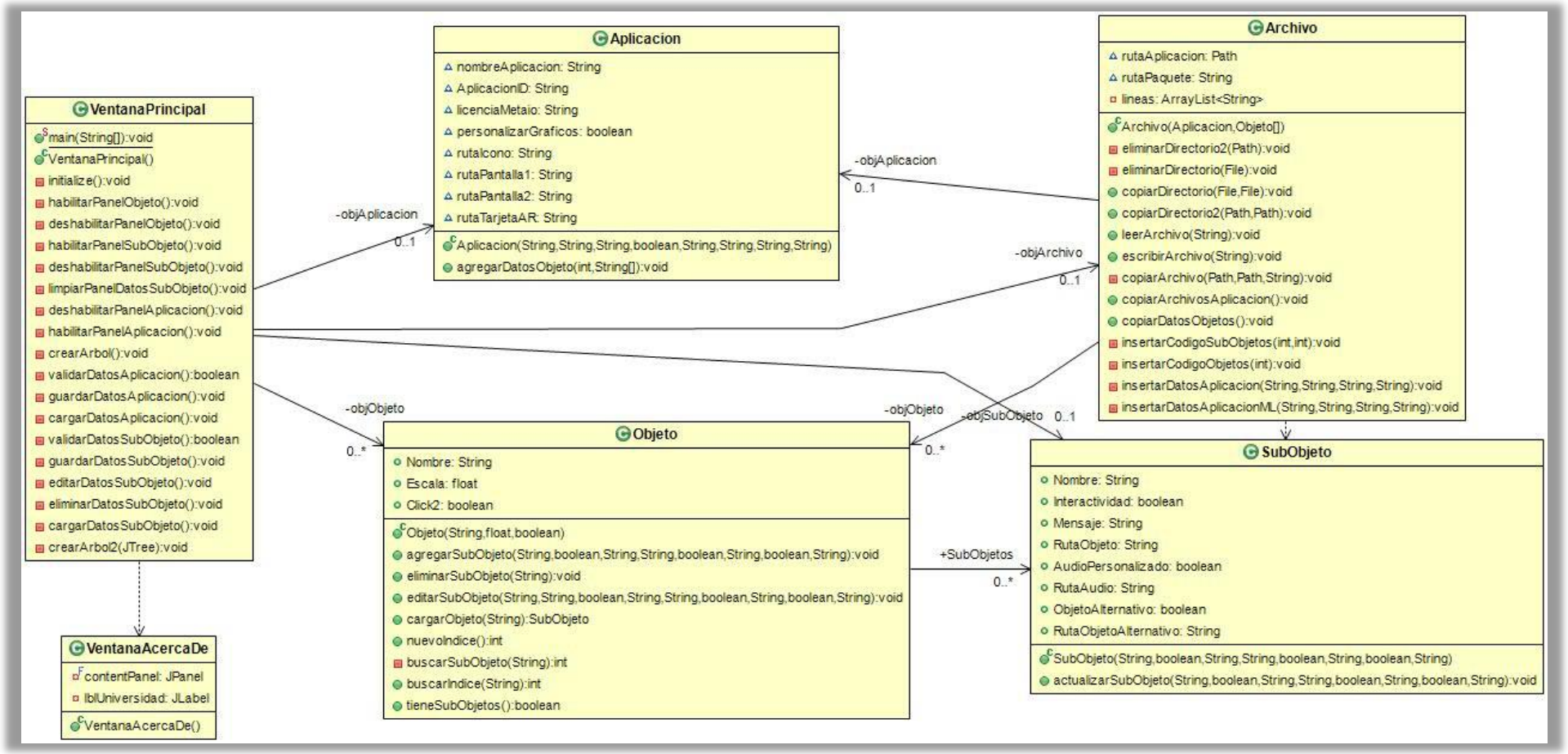
La presente aplicación de Administrador AR ha sido desarrolla para el Sistema Operativo Windows, aunque puede ejecutarse en cualquier otra plataforma a través del JDK, sin embargo la misma solo ha sido testeada bajo dicho Sistema Operativo.

**Tecnologías Utilizadas.**

La presente aplicación ha sido desarrollada con el JDK de Java, el entorno de desarrollo elegido para la realización del presente proyecto de fin de carrera es Eclipse. A continuación se muestra un detalle de las tecnologías utilizadas:

- ✓ Plataforma de Desarrollo: Eclipse Luna versión 4.4.1
- ✓ Java JDK: Java JDK versión 8u20

## Diagrama de Clases



**Descripción de las Clases.**

| Nombre de la Clase | VentanaPrincipal |
|---|---|
| **Detalles** | Clase principal de inicio que contiene la interfaz gráfica de usuario de la aplicación principal, contiene todos los controles necesarios a nivel de interfaz. |
| **Características** | boolean guardarNuevo: Variable de ayuda que permite definir si se guardara un objeto nuevo o se está editando un objeto existente; true: nuevo; false: edición.<br><br>int numObj: Guarda el número del objeto actual para trabajarlo en la aplicación.<br><br>Aplicacion objAplicacion: Guarda una instancia del objeto Aplicación.<br><br>Objeto [] objObjeto: Guarda un arreglo de instancias del objeto Objeto.<br><br>SubObjeto objSubObjeto: Guarda una instancia del objeto SubObjeto.<br><br>Archivo objArchivo: Guarda una instancia del objeto Archivo. |
| **Comportamiento** | main(String[] args): Método principal de inicio de toda aplicación en Java.<br><br>VentanaPrincipal(): Constructor de la clase Ventana Principal<br><br>initialize(): Método que se encarga de cargar toda la interfaz gráfica de usuario.<br><br>habilitarPanelObjeto(): Permite habilitar todos los componentes del panel Objeto.<br><br>habilitarPanelObjeto(): Permite deshabilitar todos los componentes del panel Objeto |

| | |
|---|---|
| | habilitarPanelAplicacion(): Permite habilitar todos los componentes del panel Aplicación.<br><br>deshabilitarPanelAplicacion(): Permite deshabilitar todos los componentes del panel Aplicación.<br><br>habilitarPanelSubObjeto(): Permite habilitar todos los componentes del panel SubObjeto.<br><br>deshabilitarPanelSubObjeto(): Permite deshabilitar todos los componentes del panel Objeto.<br><br>limpiarPanelDatosSubObjeto(): Limpia los datos del panel SubObjeto.<br><br>crearArbol(): Crea el árbol de subobjetos conforme se van ingresando.<br><br>validarDatosAplicacion(): Valida que todos los datos requeridos de la aplicación se encuentren ingresados.<br><br>validarDatosSubObjeto(): Valida que todos los datos requeridos por el subobjeto se encuentren ingresados.<br><br>guardarDatosAplicacion(): Guarda los datos de la aplicación en el objeto objAplicacion.<br><br>cargarDatosAplicacion(): Permite cargar los datos de la aplicación que han sido guardados en el objeto objAplicacion.<br><br>guardarDatosSubObjeto()Guarda los datos de la aplicación en el objeto objSubObjeto.<br><br>editarDatosSubObjeto(): Permite editar un objeto que ha sido previamente ingresado en el objeto objSubObjeto.<br><br>eliminarDatosSubObjeto(): Permite eliminar un subobjeto que ha sido previamente ingresado en el objeto objSubObjeto. |

| | |
|---|---|
| | cargarDatosSubObjeto(): Permite cargar los datos del subobjeto que han sido guardados en el objeto objSubObjeto. |

| Nombre de la Clase | VentanaAcercaDe |
|---|---|
| **Detalles** | Genera una pequeña ventana con los datos de la aplicación Administrador AR. |
| **Características** | No tiene . |
| **Comportamiento** | VentanaAcercaDe() Constructor de la clase VentanaAcercaDe. |

| Nombre de la Clase | Archivo |
|---|---|
| **Detalles** | Clase que contiene todas las rutinas de código necesarias para trabajar con ficheros dentro de la aplicación, es decir permite la edición del código de la aplicación RA-Desarrollo-Inteligencia mediante la lectura y escritura de los archivos de código fuente. |
| **Características** | Aplicacion objAplicacion: Instancia del Objeto Aplicación.<br><br>Objeto [] objObjeto: Arreglo de instancias del objeto Objeto.<br><br>Path rutaAplicacion: Guarda la ruta de instalación de la aplicación.<br><br>String rutaPaquete;ArrayList<String> líneas: ArrayList que guarda las líneas de código conforme se van leyendo para su edición. |
| **Comportamiento** | Archivo (Aplicacion app, Objeto [] obj): Constructor de la clase Archivo, este recibe una instancia del objeto aplicación y otra del objeto Objeto.<br><br>eliminarDirectorio(File f) Permite borrar un directorio o un archivo.<br><br>copiarDirectorio(File origen, File destino): Permite copiar un directorio completo. |

| | |
|---|---|
| | leerArchivo(String strRutaArchivo): Permite leer un archivo.<br><br>escribirArchivo(String strRutaArchivo): Permite escribir un archivo.<br><br>copiarArchivo(Path rutaOrigen, Path rutaDestino, String strNombreArchivo): Permite copiar un archivo.<br><br>copiarArchivosAplicacion( ) Permite copiar toda la nueva aplicación generada.<br><br>copiarDatosObjetos(): Permite copiar todos los archivos multimedia de la aplicación.<br><br>insertarCodigoSubObjetos (int numLinea, int numObjeto): Permite insertar el nuevo código fuente de los subobjetos.<br><br>insertarCodigoObjetos(int numObjeto): Permite insertar el nuevo código fuente de los objetos.<br><br>insertarDatosAplicacionML(String strRutaArchivo,String strBuscar, String strCambiar, String strCambio): Permite insertar el nuevo código fuente de la aplicación. |

| Nombre de la Clase | Aplicación |
|---|---|
| **Detalles** | Clase que guarda todos los datos generales de la aplicación, como nombre, ID, la licencia Metaio, pantallas de carga, marcador, etc. |
| **Características** | String nombreAplicacion: Guarda el nombre de la aplicación<br><br>String AplicacionID: Guarda el ID de la aplicación.<br><br>String licenciaMetaio: Guarda el string de licencia de Metaio<br><br>boolean personalizarGraficos: Determina si se utilizaran los datos por defecto de la aplicación o si estos se pueden personalizar.<br><br>String rutaIcono: Guarda la ruta del icono de la aplicación. |

| | |
|---|---|
| | String rutaPantalla1: Guarda la ruta de la pantalla de carga 1.<br><br>String rutaPantalla2: Guarda la ruta de la pantalla de carga 2.<br><br>String rutaTarjetaAR: Guarda la ruta de la tarjeta AR o marcador. |
| **Comportamiento** | Aplicacion(…..): Constructor de la clase aplicación.<br><br>agregarDatosObjeto(int num, String []objs): Permite agregar datos a la aplicación |

| **Nombre de la Clase** | **Objeto** |
|---|---|
| **Detalles** | Clase que guarda todos los datos generales del Objeto como nombre, interactividad, escala. |
| **Características** | String Nombre: Guarda el nombre del objeto principal.<br><br>float Escala: Guarda la escala del objeto.<br><br>boolean Click2: Determina si la interactividad será por 2 clic o 1 clic; true: 2 clics; false: 1 clic.<br><br>ArrayList<SubObjeto> SubObjetos: ArrayList que guarda todos los subobjeto del objeto principal. |
| **Comportamiento** | Objeto(String n,float e, boolean c): Constructor de la clase Objeto.<br><br>agregarSubObjeto(….): Permite agregar un nuevo subobjeto al objeto principal.<br><br>eliminarSubObjeto(String strSubObjeto): Elimina un subobjeto por su nombre.<br><br>editarSubObjeto(…): Permite editar un subobjeto.<br><br>cargarObjeto(String strSubObjeto): Retorna un objeto Objeto. |

| | |
|---|---|
| | nuevoIndice(): Genera un nuevo índice con el que se guardara un subobjeto.<br><br>buscarSubObjeto(String strSubObjeto): Busca un subobjeto por su nombre.<br><br>buscarIndice(String strSubObjeto) Retorna el índice del objeto buscado por du nombre.<br><br>boolean tieneSubObjetos(): Devuelve, si el objeto principal tiene subobjetos; true: tiene subobjetos; false: no tiene subobjetos. |

| Nombre de la Clase | SubObjetos |
|---|---|
| **Detalles** | Clase en la que se guarda toda la información del subobjeto del objeto principal. |
| **Características** | String Nombre: Guarda el nombre del subobjeto.<br><br>boolean Interactividad: Determina si el objeto es interactivo o no; true: es interactivo, false : no es interactivo.<br><br>String Mensaje: Guarda el mensaje desplegable del subobjeto.<br><br>String RutaObjeto: Guarda la ruta donde se encuentra almacenado el subobjeto .<br><br>boolean AudioPersonalizado: Determina si el objeto tiene audio personalizado; true: tiene audio personalizado, false no tiene audio personalizado.<br><br>String RutaAudio: Guarda la ruta donde se encuentra almacenado el audio personalizado. |

| | |
|---|---|
| | boolean ObjetoAlternativo: Determina si el objeto tiene un objeto alternativo; true: tiene objeto alternativo, false no tiene tiene objeto alternativo.<br><br>String RutaObjetoAlternativo: Guarda la ruta donde se encuentra almacenado el objeto alternativo. |
| **Comportamiento** | No tiene |

**Código Fuente de la aplicación.**

A continuación se detalla el código fuente de las principales clases, métodos y funciones de la aplicación Administrador AR, en las cuales se puede apreciarlas principales rutinas de código para la modificación y generación de la aplicación móvil de Realidad Aumentada.

**Clase: VentanaPrincipal.**

**Variables.**

```
       private boolean guardarNuevo = true;
      private int numObj = 0;//inicializamos en 0 para saber si se han creado
objetos

      private Aplicacion objAplicacion;
      private Objeto [] objObjeto;
      private SubObjeto objSubObjeto;
      private Archivo objArchivo;
```

**Métodos y Funciones.**

```
            private void habilitarPanelObjeto() {
            int aux = cmbNumeroObjetos.getSelectedIndex();
            lblNumeroObjetos.setEnabled(true);
            cmbNumeroObjetos.setEnabled(true);
            cmbNumeroObjetos.addItem("5");
            cmbNumeroObjetos.setSelectedIndex(4);
            cmbNumeroObjetos.setSelectedIndex(aux);
            cmbNumeroObjetos.removeItem("5");

            lblNota1.setEnabled(true);
            lblNota2.setEnabled(true);

            btnEditarObjeto.setEnabled(false);
            btnCancelarObjeto.setEnabled(true);
            btnGuardarObjeto.setEnabled(true);

            ((JTabbedPane)pnlObjeto.getParent()).setEnabledAt(0, false);
            ((JTabbedPane)pnlObjeto.getParent()).setEnabledAt(2, false);

      }

      private void deshabilitarPanelObjeto() {
            for(Component p:pnlDatosObjeto.getComponents())
                  for(Component c:((JPanel)p).getComponents())
                        c.setEnabled(false);
            for(Component c:pnlOpcionesObjeto.getComponents())
                  c.setEnabled(false);
            lblNumeroObjetos.setEnabled(false);
            cmbNumeroObjetos.setEnabled(false);
            lblNota1.setEnabled(false);
            lblNota2.setEnabled(false);
            btnEditarObjeto.setEnabled(true);
            ((JTabbedPane)pnlObjeto.getParent()).setEnabledAt(0, true);
            ((JTabbedPane)pnlObjeto.getParent()).setEnabledAt(2, true);

      }
```

```
	private void habilitarPanelSubObjeto() {
		txtMensaje.setEnabled(true);
		for(Component c:pnlDatosSubObjeto.getComponents())
			c.setEnabled(true);
		for(Component c:pnlOpcionesSubObjeto.getComponents())
			c.setEnabled(false);
		btnGuardar.setEnabled(true);
		btnCancelar.setEnabled(true);
		tree.setEnabled(false);
		((JTabbedPane)pnlAplicacion.getParent()).setEnabledAt(0, false);
		((JTabbedPane)pnlAplicacion.getParent()).setEnabledAt(1, false);
	}

	private void deshabilitarPanelSubObjeto() {
		for(Component c:pnlDatosSubObjeto.getComponents())
			c.setEnabled(false);
		txtMensaje.setEnabled(false);
		for(Component c:pnlOpcionesSubObjeto.getComponents())
			c.setEnabled(false);
		if(guardarNuevo){
		 btnNuevo.setEnabled(true);
		}else{
		   btnEliminar.setEnabled(true);
		   btnEditar.setEnabled(true);
		}

		 tree.setEnabled(true);
		 ((JTabbedPane)pnlAplicacion.getParent()).setEnabledAt(0, true);
		 ((JTabbedPane)pnlAplicacion.getParent()).setEnabledAt(1, true);

		 //para actualizar datos //
		 DefaultMutableTreeNode nodo =
(DefaultMutableTreeNode)tree.getLastSelectedPathComponent();
		 tree.setSelectionPath( new
TreePath(((DefaultMutableTreeNode)nodo.getParent()).getPath()));
		 tree.setSelectionPath( new TreePath(nodo.getPath()));
	}

	private void limpiarPanelDatosSubObjeto() {

	txtNombreObjetoPrincipal.setText(((DefaultMutableTreeNode)tree.getLastSele
ctedPathComponent()).toString());
		 txtNombreSubObjeto.setText("");
		 rdbtnSiInteractivo.setSelected(true);
		 txtMensaje.setText("");
		 txtRutaSubObjeto.setText("");
		 txtRutaObjetoAlternativo.setText("");
		 txtRutaAudio.setText("");
		 rdbtnNOAudioPersonalizado.setSelected(true);
		 rdbtnNOObjetoAlternativo.setSelected(true);
	}

	private void deshabilitarPanelAplicacion() {
		for(Component p:pnlAplicacion.getComponents())
			for(Component c:((JPanel)p).getComponents())
				c.setEnabled(false);
		btnEditarAplicacion.setEnabled(true);
```

```
            ((JTabbedPane)pnlAplicacion.getParent()).setEnabledAt(1, true);

            if(numObj != 0){
                  ((JTabbedPane)pnlAplicacion.getParent()).setEnabledAt(2,
true);
            }else{
                  //sirve para deshabilitar los paneles de objetos:
obj1,obj2,obj3,obj4
                  cmbNumeroObjetos.setSelectedIndex(1);
                  cmbNumeroObjetos.setSelectedIndex(0);
            }
      }


      private void habilitarPanelAplicacion() {
            for(Component p:pnlAplicacion.getComponents())
                  for(Component c:((JPanel)p).getComponents())
                        c.setEnabled(true);
            if(rdbtnNOPersonalizarGraficos.isSelected())
                  for(Component c:pnlGraficos.getComponents())
                        c.setEnabled(false);
            btnEditarAplicacion.setEnabled(false);
            ((JTabbedPane)pnlAplicacion.getParent()).setEnabledAt(1, false);
            ((JTabbedPane)pnlAplicacion.getParent()).setEnabledAt(2, false);
      }

      private void crearArbol() {
            //Construimos los nodos del arbol que seran ramas u hojas
            //Definimos cual será el directorio principal o la raiz de nuestro
arbol

              DefaultMutableTreeNode raiz = new
DefaultMutableTreeNode("Objetos");

             //Definimos el modelo donde se agregaran los nodos
              DefaultTreeModel modelo = new DefaultTreeModel(raiz);

            //Definimos mas nodos del arbol
             DefaultMutableTreeNode [] Objetos = new
DefaultMutableTreeNode[numObj];

             for(int i = 0; i < numObj; i++){
                   Objetos[i] = new
DefaultMutableTreeNode(objObjeto[i].Nombre);
             }
            //Definimos donde se agrega los nodos, dentro de que rama y que
posicion
             for(int i = 0; i < numObj;i++)
                   modelo.insertNodeInto(Objetos[i] , raiz, i);

             //agregamos el modelo al arbol, donde previamente establecimos la
raiz y los nodos
             tree.setModel(modelo);

             // expandimos al arbol para su vizualizacion
             for (int i = 0; i < tree.getRowCount(); i++) {
               tree.expandRow(i);
           }
             tree.setSelectionRow(1);
```

```
	}

	private boolean validarDatosAplicacion() {

	if((!txtNombreAplicacion.getText().trim().isEmpty())&&(!txtAplicacionID.ge
tText().trim().isEmpty())&&(!txtStringLicencia.getText().trim().isEmpty())){
			return true;
		}

		JOptionPane.showMessageDialog(pnlAplicacion, "Error al guardar\nPor
favor ingrese todos los datos");
		return false;
	}

	private void guardarDatosAplicacion() {
		String n = txtNombreAplicacion.getText().trim();
		String id = txtAplicacionID.getText().trim();
		String lm = txtStringLicencia.getText().trim();

		boolean pa = rdbtnSiPersonalizarGraficos.isSelected();
		String ri = txtRutaIcono.getText().trim();
		String rp1 = txtRutaPantalla1.getText().trim();
		String rp2 = txtRutaPantalla2.getText().trim();
		String rt = txtRutaTarjetaAR.getText().trim();

		if(!pa){
			ri = "C:\\AdministradorAR\\Recursos\\ic_launcher.png";
			rp1 = "C:\\AdministradorAR\\Recursos\\splash_utpl.jpg";
			rp2 = "C:\\AdministradorAR\\Recursos\\splash_asignatura.jpg";
			rt = "C:\\AdministradorAR\\Recursos\\imageTarget.jpg";
		}

		objAplicacion = new Aplicacion(n, id, lm, pa, ri, rp1, rp2,rt);

	}

	private void cargarDatosAplicacion() {
		txtNombreAplicacion.setText(objAplicacion.nombreAplicacion);
		txtAplicacionID.setText(objAplicacion.AplicacionID);
		txtStringLicencia.setText(objAplicacion.licenciaMetaio);
		if(objAplicacion.personalizarGraficos)
			rdbtnSiPersonalizarGraficos.setSelected(true);
		else
			rdbtnNOPersonalizarGraficos.setSelected(true);

		txtRutaIcono.setText(objAplicacion.rutaIcono);
		txtRutaPantalla1.setText(objAplicacion.rutaPantalla1);
		txtRutaPantalla2.setText(objAplicacion.rutaPantalla2);
		txtRutaTarjetaAR.setText(objAplicacion.rutaTarjetaAR);

	}

	private boolean validarDatosSubObjeto() {

		String mensaje = "Error al guardar\nPor favor ingrese todos los
datos";
```

```
      if((!txtNombreSubObjeto.getText().trim().isEmpty())&&(!txtMensaje.getText(
).trim().isEmpty())&&(!txtRutaSubObjeto.getText().trim().isEmpty())){
                  if(rdbtnSIAudioPersonalizado.isSelected()){
                        if(txtRutaAudio.getText().trim().isEmpty()){
                              JOptionPane.showMessageDialog(pnlSubObjeto,
mensaje);
                              return false;
                        }
                  }
                  if(rdbtnSIObjetoAlternativo.isSelected()){

      if(txtRutaObjetoAlternativo.getText().trim().isEmpty()){
                              JOptionPane.showMessageDialog(pnlSubObjeto,
mensaje);
                              return false;
                        }
                  }

                  return true;
            }

            JOptionPane.showMessageDialog(pnlSubObjeto, mensaje);
            return false;

      }
      private void guardarDatosSubObjeto() {

            int indice = 0;

            DefaultMutableTreeNode nodopadre =
(DefaultMutableTreeNode)tree.getLastSelectedPathComponent();
            String strObjeto = nodopadre.getUserObject().toString();

            String n = txtNombreSubObjeto.getText().trim();
            boolean in = rdbtnSiInteractivo.isSelected();
            String m = txtMensaje.getText().trim();
            String ro = txtRutaSubObjeto.getText().trim();
            boolean ap = rdbtnSIAudioPersonalizado.isSelected();
            String ra = txtRutaAudio.getText().trim();
            boolean oa = rdbtnSIObjetoAlternativo.isSelected();
            String roa = txtRutaObjetoAlternativo.getText().trim();

            if(!ap)
                  ra = "";

            for (int i = 0; i < numObj; i++){
                  if(strObjeto.compareTo(objObjeto[i].Nombre) == 0){

                        indice = objObjeto[i].nuevoIndice();
                        objObjeto[i].agregarSubObjeto(n,in, m, ro, ap, ra, oa,
roa);

                  }
            }

            DefaultTreeModel modelo = (DefaultTreeModel)tree.getModel();
            DefaultMutableTreeNode nodonuevo = new DefaultMutableTreeNode(n);
```

```
            modelo.insertNodeInto(nodonuevo, nodopadre, indice);

            tree.setSelectionPath(new TreePath(nodonuevo.getPath()));
            tree.setSelectionPath(new TreePath(nodopadre.getPath()));
      }

      private void editarDatosSubObjeto() {

            int indice = 0;
            DefaultMutableTreeNode nodo =
(DefaultMutableTreeNode)tree.getLastSelectedPathComponent();
            DefaultMutableTreeNode nodopadre =
(DefaultMutableTreeNode)nodo.getParent();
            String strObjeto =  nodopadre.getUserObject().toString();
            String strSubObjeto = nodo.getUserObject().toString();

            String n = txtNombreSubObjeto.getText().trim();
            boolean in = rdbtnSiInteractivo.isSelected();
            String m = txtMensaje.getText().trim();
            String ro = txtRutaSubObjeto.getText().trim();
            boolean ap = rdbtnSIAudioPersonalizado.isSelected();
            String ra = txtRutaAudio.getText().trim();
            boolean oa = rdbtnSIObjetoAlternativo.isSelected();
            String roa = txtRutaObjetoAlternativo.getText().trim();

            if(!ap)
                  ra = "";

            for (int i = 0; i < numObj; i++){
                  if(strObjeto.compareTo(objObjeto[i].Nombre) == 0){

                        indice = objObjeto[i].buscarIndice(strSubObjeto);
                        objObjeto[i].editarSubObjeto(strSubObjeto, n, in, m,
ro, ap, ra, oa, roa);
                  }
            }
            DefaultTreeModel modelo = (DefaultTreeModel)tree.getModel();
            modelo.removeNodeFromParent(nodo);

            DefaultMutableTreeNode nodonuevo = new DefaultMutableTreeNode(n);
            modelo.insertNodeInto(nodonuevo, nodopadre,indice );

            tree.setSelectionPath(new TreePath(nodonuevo.getPath()));
      }
      private void eliminarDatosSubObjeto() {

            DefaultMutableTreeNode nodo =
(DefaultMutableTreeNode)tree.getLastSelectedPathComponent();
            DefaultMutableTreeNode nodopadre =
(DefaultMutableTreeNode)nodo.getParent();
            String strObjeto =  nodopadre.getUserObject().toString();
            String strSubObjeto = nodo.getUserObject().toString();

            for (int i = 0; i < numObj; i++){
                  if(strObjeto.compareTo(objObjeto[i].Nombre) == 0){
                        objObjeto[i].eliminarSubObjeto(strSubObjeto);
                  }
            }
            DefaultTreeModel modelo = (DefaultTreeModel)tree.getModel();
```

```
            modelo.removeNodeFromParent(nodo);
            tree.setSelectionPath( new TreePath(nodopadre.getPath()))
      }

      private void cargarDatosSubObjeto() {

            DefaultMutableTreeNode nodo =
(DefaultMutableTreeNode)tree.getLastSelectedPathComponent();
            DefaultMutableTreeNode nodopadre =
(DefaultMutableTreeNode)nodo.getParent();
            String strObjeto =  nodopadre.getUserObject().toString();
            String strSubObjeto = nodo.getUserObject().toString();

            for (int i = 0; i < numObj; i++){
                  if(strObjeto.compareTo(objObjeto[i].Nombre) == 0){
                        objSubObjeto =
objObjeto[i].cargarObjeto(strSubObjeto);
                  }
            }
            txtNombreObjetoPrincipal.setText(nodopadre.toString());

            if(objSubObjeto.Interactividad)
                  rdbtnSiInteractivo.setSelected(true);
            else
                  rdbtnNoInteractivo.setSelected(true);

            txtNombreSubObjeto.setText(objSubObjeto.Nombre);
            txtMensaje.setText(objSubObjeto.Mensaje);
            txtRutaSubObjeto.setText(objSubObjeto.RutaObjeto);

            if(objSubObjeto.AudioPersonalizado)
                  rdbtnSIAudioPersonalizado.setSelected(true);
            else
                  rdbtnNOAudioPersonalizado.setSelected(true);
            txtRutaAudio.setText(objSubObjeto.RutaAudio);

            if(objSubObjeto.ObjetoAlternativo)
                  rdbtnSIObjetoAlternativo.setSelected(true);
            else
                  rdbtnNOObjetoAlternativo.setSelected(true);
      txtRutaObjetoAlternativo.setText(objSubObjeto.RutaObjetoAlternativo);

      }
```

**Clase: VentanaAcercaDe.**

```
public class VentanaAcercaDe extends JDialog {

      private final JPanel contentPanel = new JPanel();
      private JLabel lblUniversidad;
      /**
```

```
	 * Create the dialog.
	 */
	public VentanaAcercaDe() {
		setBounds(100, 100, 450, 300);
		getContentPane().setLayout(new BorderLayout());
		contentPanel.setBorder(new EmptyBorder(5, 5, 5, 5));
		getContentPane().add(contentPanel, BorderLayout.CENTER);
		contentPanel.setLayout(null);

		lblUniversidad = new JLabel("Universidad Tecnica Particula de
Loja");
		lblUniversidad.setFont(new Font("Tahoma", Font.BOLD, 18));
		lblUniversidad.setBounds(44, 39, 346, 14);
		contentPanel.add(lblUniversidad);
		{
			JLabel lblAplicacion = new JLabel("AdministradorAR  -
version: 1.5");
			lblAplicacion.setFont(new Font("Tahoma", Font.BOLD, 12));
			lblAplicacion.setBounds(109, 188, 206, 14);
			contentPanel.add(lblAplicacion);
		}
		{
			JLabel lblEscuela = new JLabel("Escuela de Ciencias de la
Computacion");
			lblEscuela.setFont(new Font("Tahoma", Font.BOLD, 14));
			lblEscuela.setBounds(83, 75, 265, 14);
			contentPanel.add(lblEscuela);
		}{
			JLabel lblTema = new JLabel("Realidad Aumentada en el
componente academico Desarrollo de la Inteligencia");
			lblTema.setFont(new Font("Tahoma", Font.PLAIN, 11));
			lblTema.setBounds(27, 122, 375, 14);
			contentPanel.add(lblTema);
		}{
			JLabel lblTema_1 = new JLabel("de la Modalidad Abierta y a
Distancia de la UTPL");
			lblTema_1.setFont(new Font("Tahoma", Font.PLAIN, 11));
			lblTema_1.setBounds(97, 141, 229, 14);
			contentPanel.add(lblTema_1);
		}{
			JPanel buttonPane = new JPanel();
			buttonPane.setLayout(new FlowLayout(FlowLayout.RIGHT));
			getContentPane().add(buttonPane, BorderLayout.SOUTH);
			{
				JButton okButton = new JButton("OK");
				okButton.addActionListener(new ActionListener() {
					public void actionPerformed(ActionEvent e){
					}
				});
				okButton.setActionCommand("OK");
				buttonPane.add(okButton);
				getRootPane().setDefaultButton(okButton);
			}
		}
		this.setModal(true);
	}
}
```

**Clase: Archivo.**

```
public class Archivo {

	private Aplicacion objAplicacion;
	private Objeto [] objObjeto;
	Path rutaAplicacion;
	String rutaPaquete;

	private ArrayList<String> lineas = new ArrayList<String>();

	public Archivo(Aplicacion app, Objeto [] obj) {
		objAplicacion = app;
		objObjeto = obj;
		}

	public void copiarDirectorio(File origen, File destino) throws IOException
{
		  if(origen.isDirectory()){
		    Files.createDirectories(destino.toPath());

		    for (File c : origen.listFiles()){
		      Path nrd = destino.toPath().resolve(c.getName());
		      copiarDirectorio(c,  nrd.toFile());
		    }
		  } else {
		    // copiamos el archivo
		      CopyOption[] co = new
CopyOption[]{StandardCopyOption.REPLACE_EXISTING,StandardCopyOption.COPY_ATTRIBU
TES};
		      Files.copy(origen.toPath(), destino.toPath(),co);
		  }
		}

	public void leerArchivo(String strRutaArchivo)throws
FileNotFoundException, IOException{

		  lineas.clear();

		    FileReader fr = new FileReader(strRutaArchivo);
		BufferedReader br = new BufferedReader(fr);

		for(String l = br.readLine(); l!=null; l = br.readLine())
		  lineas.add(l);
		fr.close();

	}
	public void escribirArchivo(String strRutaArchivo)throws
FileNotFoundException, IOException{

		  FileWriter fw = new FileWriter(strRutaArchivo);
	  PrintWriter pw = new PrintWriter(fw);

	  for(int i = 0; i < lineas.size();i++)
	      pw.println(lineas.get(i));
	  fw.close();
	}
```

```
      private void copiarArchivo(Path rutaOrigen, Path rutaDestino,String
strNombreArchivo)throws FileNotFoundException, IOException{

            Files.createDirectories(rutaDestino);
            Path nrd = rutaDestino;
            if(strNombreArchivo != null)
                  nrd = Paths.get(rutaDestino+"\\"+strNombreArchivo);
            else
                  nrd = rutaDestino.resolve(rutaOrigen.getFileName());

        CopyOption[] co = new
CopyOption[]{StandardCopyOption.REPLACE_EXISTING,StandardCopyOption.COPY_ATTRIBU
TES};
        Files.copy(rutaOrigen, nrd, co);
      }

      public void copiarArchivosAplicacion( )throws FileNotFoundException,
IOException {

            rutaAplicacion = Paths.get("C:\\AdministradorAR\\" +
objAplicacion.nombreAplicacion);

            Path rutaOrigen = Paths.get("C:\\AdministradorAR\\UTPL");
            Path rutaDestino = rutaAplicacion;

            copiarDirectorio(rutaOrigen.toFile(),rutaDestino.toFile());

            rutaOrigen = Paths.get(rutaAplicacion+"\\src\\temp\\metaio\\UTPL");
            rutaDestino =
Paths.get(rutaAplicacion+"\\src\\"+objAplicacion.AplicacionID.replaceAll("\\.",
"\\\\"));

            copiarDirectorio(rutaOrigen.toFile(),rutaDestino.toFile());
      eliminarDirectorio(Paths.get(rutaAplicacion+"\\src\\temp\\").toFile());

            rutaOrigen =
Paths.get(rutaAplicacion+"\\gen\\temp\\metaio\\UTPL\\R.java");
            rutaDestino =
Paths.get(rutaAplicacion+"\\gen\\"+objAplicacion.AplicacionID.replaceAll("\\.",
"\\\\"));
            copiarArchivo(rutaOrigen,rutaDestino,null);
      eliminarDirectorio(Paths.get(rutaAplicacion+"\\gen\\temp\\").toFile());

            rutaDestino = Paths.get(rutaAplicacion+"\\res\\drawable-hdpi\\");

            rutaOrigen = Paths.get(objAplicacion.rutaIcono);
            copiarArchivo(rutaOrigen, rutaDestino,"ic_launcher.png");

            rutaOrigen = Paths.get(objAplicacion.rutaPantalla1);
            copiarArchivo(rutaOrigen, rutaDestino,"splash_utpl.jpg");

            rutaOrigen = Paths.get(objAplicacion.rutaPantalla2);
            copiarArchivo(rutaOrigen, rutaDestino,"splash_asignatura.jpg");

            rutaPaquete = objAplicacion.AplicacionID.replaceAll("\\.", "\\\\");

            //cambia nombre de aplicacion
```

```
      insertarDatosAplicacionML(rutaAplicacion+"\\.project","UTPL","UTPL",
objAplicacion.nombreAplicacion);

      insertarDatosAplicacionML(rutaAplicacion+"\\res\\values\\strings.xml","UTP
L","UTPL", objAplicacion.nombreAplicacion);

            //cambia string licencia

      insertarDatosAplicacionML(rutaAplicacion+"\\res\\values\\signature.xml","s
jbDtCqB1mA/BFBG9qEXy72VHiSng7ZVAKF7kfbJ6Uc=","sjbDtCqB1mA/BFBG9qEXy72VHiSng7ZVAK
F7kfbJ6Uc=", objAplicacion.licenciaMetaio);

            //cambia import and package al nuevo nombre o ID

      insertarDatosAplicacionML(rutaAplicacion+"\\gen\\"+rutaPaquete+"\\R.java",
"com.metaio.UTPL","com.metaio.UTPL", objAplicacion.AplicacionID);

      insertarDatosAplicacionML(rutaAplicacion+"\\src\\"+rutaPaquete+"\\ARELView
Activity.java","com.metaio.UTPL","com.metaio.UTPL", objAplicacion.AplicacionID);

      insertarDatosAplicacionML(rutaAplicacion+"\\src\\"+rutaPaquete+"\\MainActi
vity.java","com.metaio.UTPL","com.metaio.UTPL", objAplicacion.AplicacionID);

      insertarDatosAplicacionML(rutaAplicacion+"\\src\\"+rutaPaquete+"\\SplashAc
tivity.java","com.metaio.UTPL","com.metaio.UTPL", objAplicacion.AplicacionID);

      insertarDatosAplicacionML(rutaAplicacion+"\\src\\"+rutaPaquete+"\\UTPL.jav
a","com.metaio.UTPL","com.metaio.UTPL", objAplicacion.AplicacionID);

      insertarDatosAplicacionML(rutaAplicacion+"\\AndroidManifest.xml","com.meta
io.UTPL","com.metaio.UTPL", objAplicacion.AplicacionID);

      }

      public void copiarDatosObjetos()throws FileNotFoundException, IOException
{
            Path rutaOrigen;
            Path rutaDestinoArchivos;
        rutaDestinoArchivos = Paths.get(rutaAplicacion+"\\assets\\");

        for(int i = 0; i< objObjeto.length;i++){
            for(int j = 0; j < objObjeto[i].SubObjetos.size();j++){
                  rutaOrigen =
Paths.get(objObjeto[i].SubObjetos.get(j).RutaObjeto);

      copiarArchivo(rutaOrigen,rutaDestinoArchivos,rutaOrigen.getFileName().toSt
ring().replaceAll(" ", "_"));
            }
        }

        for(int i = 0; i< objObjeto.length;i++){
            for(int j = 0; j < objObjeto[i].SubObjetos.size();j++){
                  if(objObjeto[i].SubObjetos.get(j).ObjetoAlternativo){
                        rutaOrigen =
Paths.get(objObjeto[i].SubObjetos.get(j).RutaObjetoAlternativo);

      copiarArchivo(rutaOrigen,rutaDestinoArchivos,rutaOrigen.getFileName().toSt
ring().replaceAll(" ", "_"));
```

```
                    }
               }
          }

          rutaDestinoArchivos = Paths.get(rutaAplicacion+"\\res\\raw\\");

          for(int i = 0; i< objObjeto.length;i++){
               for(int j = 0; j < objObjeto[i].SubObjetos.size();j++){
                    if(objObjeto[i].SubObjetos.get(j).AudioPersonalizado){
                         rutaOrigen =
Paths.get(objObjeto[i].SubObjetos.get(j).RutaAudio);

      copiarArchivo(rutaOrigen,rutaDestinoArchivos,rutaOrigen.getFileName().toSt
ring().replaceAll(" ", "_"));
                    }
               }
          }
          System.out.println( "copiado terminado");

          for(int i = 0; i< objObjeto.length;i++){
               insertarCodigoObjetos(i);
          }
       }

       private void insertarCodigoSubObjetos(int numLinea, int numObjeto)throws
FileNotFoundException, IOException {

               //inicializamos todas las variables
               String lineaNombre= "\tprivate String []
strNomPartesObj"+(numObjeto+1)+" = {";
               String lineaMensaje = "\tprivate String []
strMensajeObj"+(numObjeto+1)+" = {\n\t\t";
               String lineaRutaObj = "\tprivate String []
strRutasObj"+(numObjeto+1)+" = {";
               String lineaRutaAltObj = "\tprivate String []
strRutasAltObj"+(numObjeto+1)+" = {";
               String lineaRutaAudio = "\tprivate Integer []
rutasAudioObj"+(numObjeto+1)+" = {";
               String lineaInteractividad = "\tprivate boolean []
interactividadObj"+(numObjeto+1)+" = {";

               String lineaActivarObj = "\tprivate boolean
activarObj"+(numObjeto+1)+" = true;";// asegurarse de setar los obj con false
               String lineaEscala = "\tprivate float escalaObj"+(numObjeto+1)+"
" + objObjeto[numObjeto].Escala+"f;";
               String lineaClick = "\tprivate boolean click2Obj"+(numObjeto+1)+"
" + objObjeto[numObjeto].Click2 + ";";

               Path p;

               for(int i = 0; i < objObjeto[numObjeto].SubObjetos.size();i++){
                    //agrega los nobre de subObjeto, su mensaje y si son o no
interactivos
                    lineaNombre = lineaNombre + "\""+
objObjeto[numObjeto].SubObjetos.get(i).Nombre+"\", ";
                    lineaMensaje = lineaMensaje +
"\""+objObjeto[numObjeto].SubObjetos.get(i).Mensaje+"\",\n\t\t";
                    lineaInteractividad = lineaInteractividad +
objObjeto[numObjeto].SubObjetos.get(i).Interactividad +", ";
```

```
                    //para agregar los objetos
                    p =
Paths.get(objObjeto[numObjeto].SubObjetos.get(i).RutaObjeto);
                    lineaRutaObj = lineaRutaObj +
"\""+p.getFileName().toString().replace(" ", "_")+"\", ";

                    //para agregar objeto alternativo
             if(objObjeto[numObjeto].SubObjetos.get(i).ObjetoAlternativo){
                           p =
Paths.get(objObjeto[numObjeto].SubObjetos.get(i).RutaObjetoAlternativo);
                           lineaRutaAltObj = lineaRutaAltObj +
"\""+p.getFileName().toString().replace(" ", "_")+"\", ";
                    }else
                           lineaRutaAltObj = lineaRutaAltObj +  "null, ";

                    //para agregar el audio
             if(objObjeto[numObjeto].SubObjetos.get(i).AudioPersonalizado){
                           p =
Paths.get(objObjeto[numObjeto].SubObjetos.get(i).RutaAudio);
                           lineaRutaAudio = lineaRutaAudio + "R.raw."+
p.getFileName().toString().replace(" ", "_").split("\\.")[0]+", ";
                    }else
                           lineaRutaAudio = lineaRutaAudio + "null, ";
             }
             //realizamos la correccion de la ultima coma y cerramos la linea de
codigo
             lineaNombre= lineaNombre.substring(0, lineaNombre.length()-2)+"};";
             lineaMensaje = lineaMensaje.substring(0, lineaMensaje.length()-
4)+"};";
             lineaRutaObj = lineaRutaObj.substring(0, lineaRutaObj.length()-
2)+"};";
             lineaRutaAltObj = lineaRutaAltObj.substring(0,
lineaRutaAltObj.length()-2)+"};";
             lineaRutaAudio= lineaRutaAudio.substring(0,
lineaRutaAudio.length()-2)+"};";
             lineaInteractividad = lineaInteractividad.substring(0,
lineaInteractividad.length()-2)+"};";

             //agregamos todas las lineas realizadas
             lineas.add(numLinea, lineaClick);
             lineas.add(numLinea, lineaEscala);
             lineas.add(numLinea, lineaActivarObj);
             lineas.add(numLinea, lineaInteractividad);
             lineas.add(numLinea, lineaRutaAudio);
             lineas.add(numLinea, lineaRutaAltObj);
             lineas.add(numLinea, lineaRutaObj);
             lineas.add(numLinea, lineaMensaje);
             lineas.add(numLinea, lineaNombre);

             //imprimimos en consola las lineas de codigo generadas
             System.out.println( lineaNombre);
             System.out.println( lineaMensaje);
             System.out.println( lineaRutaObj);
             System.out.println( lineaRutaAltObj);
             System.out.println( lineaRutaAudio);
             System.out.println( lineaInteractividad);
             System.out.println( lineaActivarObj);
             System.out.println( lineaEscala);
```

```
            System.out.println( lineaClick);
      }

      private void insertarCodigoObjetos(int numObjeto)throws
FileNotFoundException, IOException {
            String strRutaArchivo =
rutaAplicacion+"\\src\\"+rutaPaquete+"\\UTPL.java";
            leerArchivo(strRutaArchivo);
            for (int i = 0; i < lineas.size();i++){
      if(lineas.get(i).trim().contains("ObjetoX"+(numObjeto+1))){

            lineas.add(i,lineas.get(i).replaceAll("ObjetoX"+(numObjeto+1),
objObjeto[numObjeto].Nombre));
                                 //antes de empezar a editar el codigo
                                //se remueve el codigo que sera cambiado
                                for(int j = 0; j < 10; j++)
                                      lineas.remove(i+1);
                                System.out.println( lineas.get(i));
                                insertarCodigoSubObjetos(i+1,numObjeto)
                  }
            }
            escribirArchivo(strRutaArchivo);
      }
private void insertarDatosAplicacionML(String strRutaArchivo,String strBuscar,
String strCambiar, String strCambio)throws FileNotFoundException, IOException {

            leerArchivo(strRutaArchivo);
            for (int i = 0; i < lineas.size();i++){
                        if(lineas.get(i).trim().contains(strBuscar)){

      lineas.add(i,lineas.get(i).replaceAll(strCambiar, strCambio));
                              lineas.remove(i+1);
                              System.out.println( lineas.get(i));
                        }
            }
            escribirArchivo(strRutaArchivo);
      }
}
```

**Clase: Aplicación.**

```
public class Aplicacion {
      String nombreAplicacion;
      String AplicacionID;
      String licenciaMetaio;
      boolean personalizarGraficos;
      String rutaIcono;
      String rutaPantalla1;
      String rutaPantalla2;
      String rutaTarjetaAR;

      public Aplicacion(String n, String id,String lm, boolean pa, String ri,
String rp1, String rp2, String rt) {
            nombreAplicacion = n;
            AplicacionID = id;
            licenciaMetaio = lm;
```

```
            personalizarGraficos = pa;
            rutaIcono = ri;
            rutaPantalla1 = rp1;
            rutaPantalla2 = rp2;
            rutaTarjetaAR = rt;
      }
      public void agregarDatosObjeto(int num, String []objs){

      }

}
```

**Clase: Objeto.**

```
public class Objeto {

      public String Nombre;
      public float Escala;
      public boolean Click2;
      public ArrayList<SubObjeto> SubObjetos = new ArrayList<SubObjeto>();

      public Objeto(String n,float e, boolean c) {
            Nombre = n;
            Escala = e;
            Click2 = c;
      }

      public void agregarSubObjeto(String n, boolean in, String m, String ro,
boolean ap, String ra, boolean oa, String roa) {
            SubObjetos.add(new SubObjeto(n, in, m, ro, ap, ra, oa, roa));
      }

      public void eliminarSubObjeto(String strSubObjeto) {
            SubObjetos.remove(buscarSubObjeto(strSubObjeto));
      }

      public void editarSubObjeto(String strSubObjeto, String n, boolean in,
String m,String ro, boolean ap, String ra, boolean oa, String roa) {

      SubObjetos.get(buscarSubObjeto(strSubObjeto)).actualizarSubObjeto(n, in,
m, ro, ap, ra, oa, roa);
      }

      public SubObjeto cargarObjeto(String strSubObjeto) {
            return SubObjetos.get(buscarSubObjeto(strSubObjeto));
      }
      public int nuevoIndice() {
            return SubObjetos.size();
      }

      private int buscarSubObjeto(String strSubObjeto) {
            for(int i = 0; i < SubObjetos.size();i++){
                if (SubObjetos.get(i).Nombre.compareTo(strSubObjeto) == 0)
                        return i;
            }
            return -1;
```

```
	}

	public int buscarIndice(String strSubObjeto) {
		return buscarSubObjeto(strSubObjeto);
	}

	public boolean tieneSubObjetos() {
		if(SubObjetos.size() > 0)
			return true;
		else
			return false;
	}
}
```

**Clase: SubObjeto.**

```
public class SubObjeto {

	public String Nombre;
	public boolean Interactividad;
	public String Mensaje;
	public String RutaObjeto;
	public boolean AudioPersonalizado;
	public String RutaAudio;
	public boolean ObjetoAlternativo;
	public String RutaObjetoAlternativo;

	public SubObjeto(String n ,boolean in, String m, String ro, boolean ap,
String ra, boolean oa, String roa){
		Nombre = n;
		Interactividad = in;
		Mensaje = m;
		RutaObjeto = ro;
		AudioPersonalizado = ap;
		RutaAudio = ra;
		ObjetoAlternativo = oa;
		RutaObjetoAlternativo = roa;
	}
	public void actualizarSubObjeto(String n, boolean in, String m,  String
ro, boolean ap, String ra, boolean oa, String roa) {
		Nombre = n;
		Interactividad = in;
		Mensaje = m;
		RutaObjeto = ro;
		AudioPersonalizado = ap;
		RutaAudio = ra;
		ObjetoAlternativo = oa;
		RutaObjetoAlternativo = roa;
	}
}
```

# MANUAL DEL ADMINISTRADOR

## Administrador AR

### Descripción breve

Manual de administración de la aplicación de Realidad Aumentada del componente académico Desarrollo de la Inteligencia de la modalidad abierta y a distancia de la UTPL

Michael Freire
mfffreire@utpl.edu.ec

## TABLA DE CONTENIDOS

## 1. INTRODUCCIÓN.

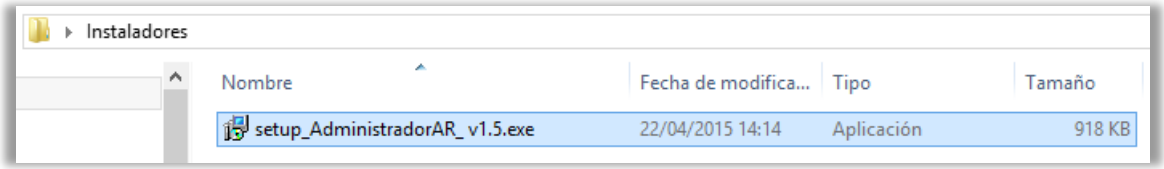
La aplicación de “Administrador AR” fue desarrollada con el fin de facilitar las tareas administrativas de la aplicación de Realidad Aumentada (RA) desarrollada en el actual proyecto de fin de carrera, la misma permite de forma rápida e intuitiva personalizar la aplicación de RA del componente académico “Desarrollo de la Inteligencia” con el objetivo de poder reutilizar dicha aplicación de RA en otros componentes académicos de la UTPL.

## 2. INSTALACION DE LA APLICACION DE ADMINISTRACION.

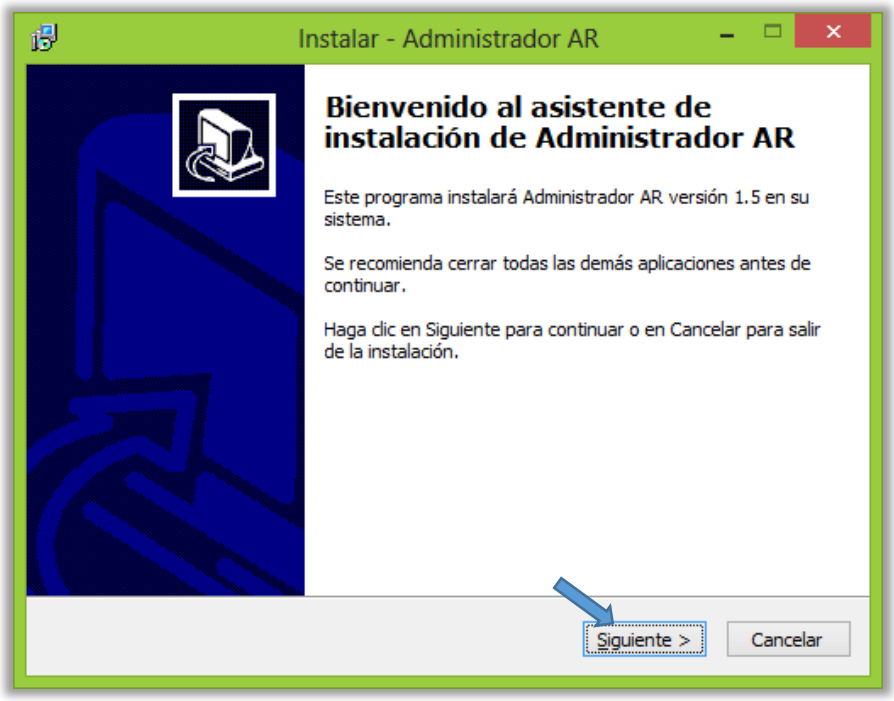
Para la ejecución de “Administrador AR” en necesario tener previamente instalado el JDK de java; si ya se ha instalado el entorno de desarrollo Android de Eclipse tal y como se explica en el Manual del Programador, se debería tener ya instalado el JDK de Java; caso contrario, remitirse a dicho manual para realizar la respectiva instalación del JDK de Java.

A continuación se detalla los paso a seguir para la instalación del Administrador AR.

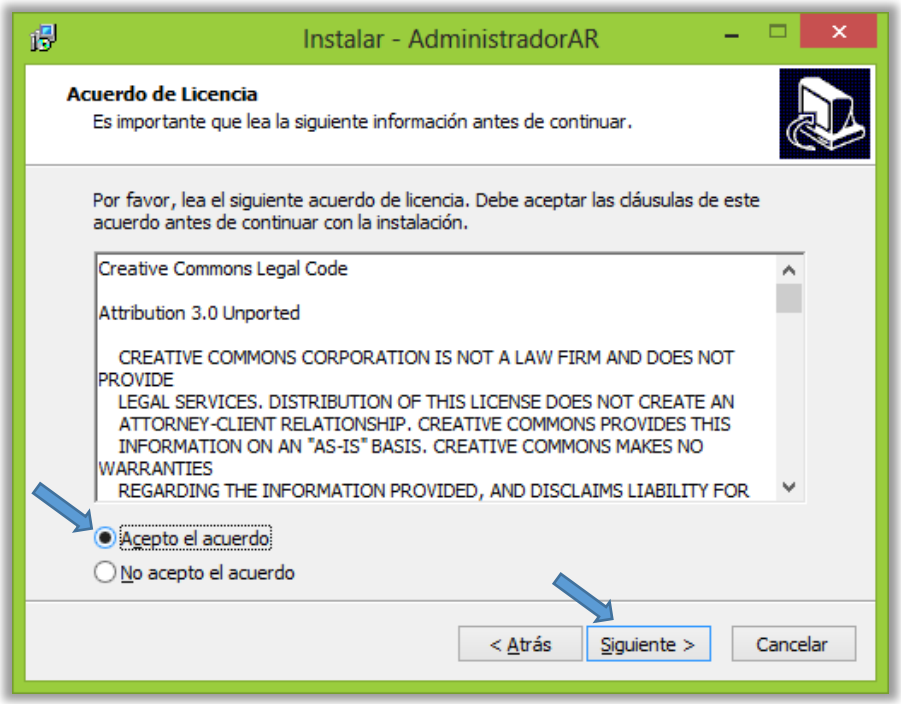
1. Se ejecuta el archivo de instalador de Administrador AR llamado “setup_AdministradorAR_ v1.5.exe”.



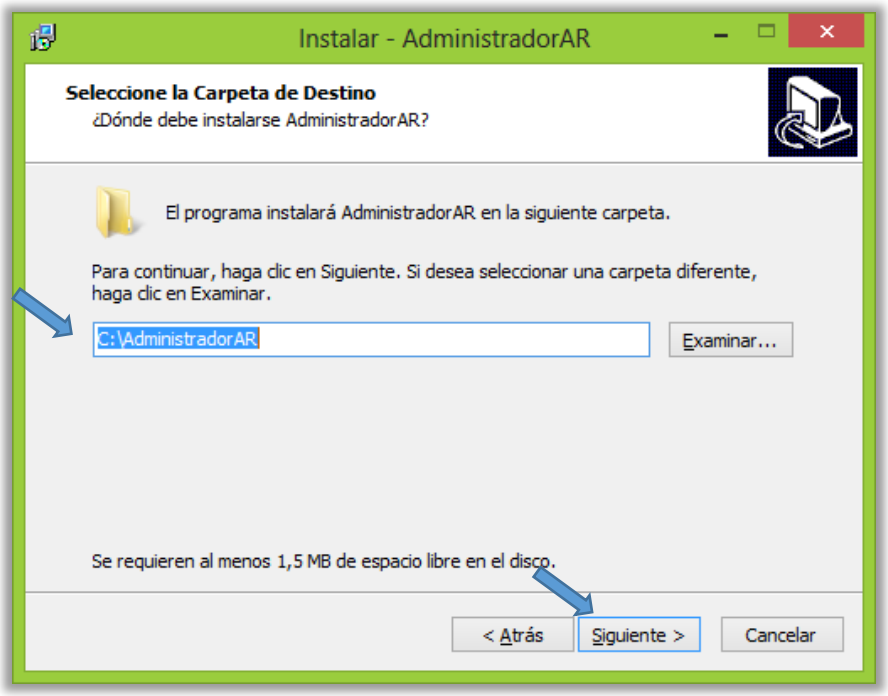

2. Posteriormente se cargara el instalador y se muestra la siguiente ventana de bienvenida en la que se debe dar clic en “Siguiente”.



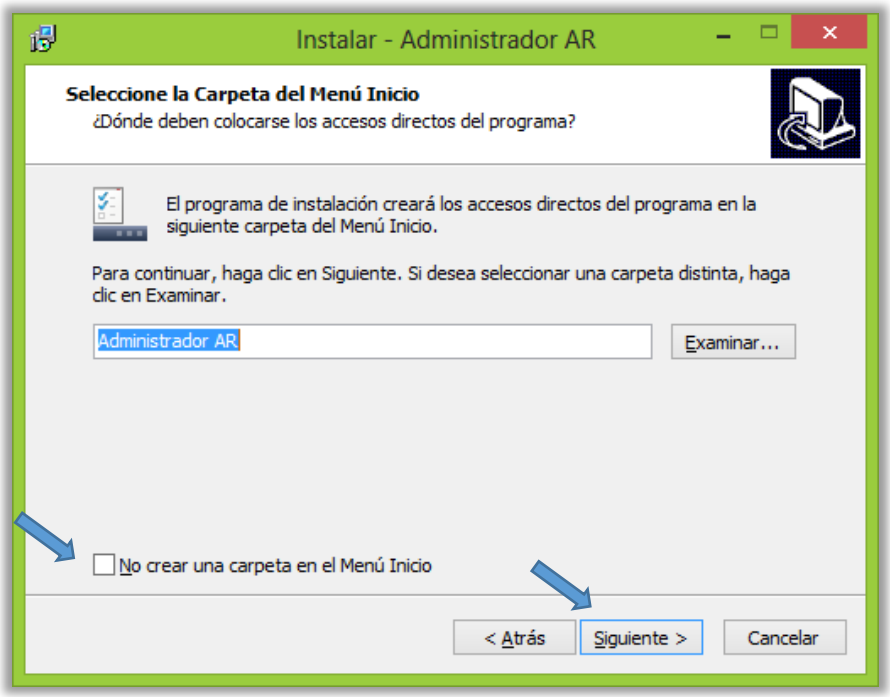

3. Posteriormente se acepta la licencia de la aplicación y se presiona en el botón “Siguiente” para continuar con la instalación.



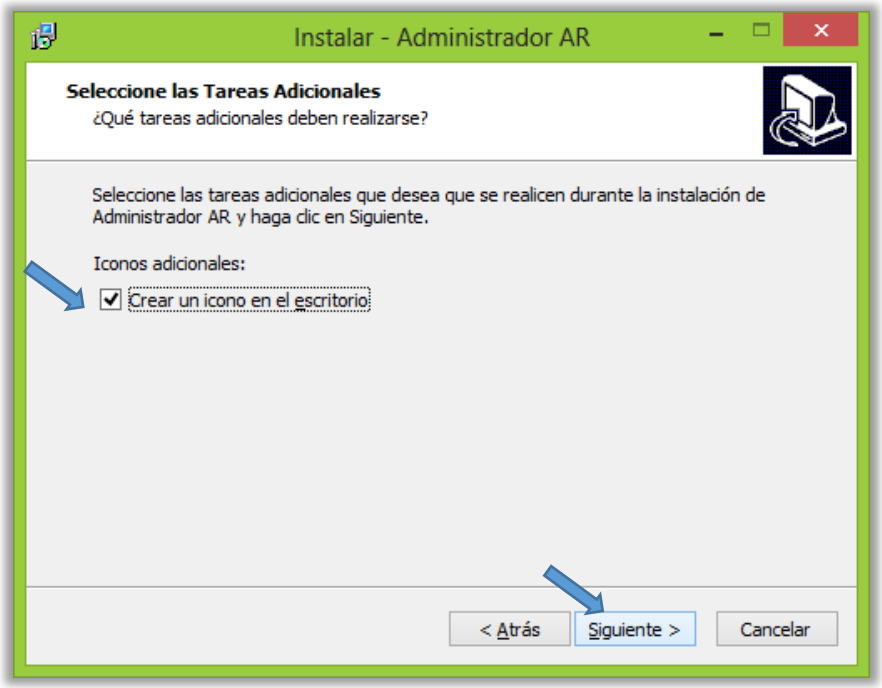

4. Posteriormente se cargara la siguiente ventana donde se debe seleccionar el directorio de instalación (No cambiar dicha ruta), y se da clic en “Siguiente”.



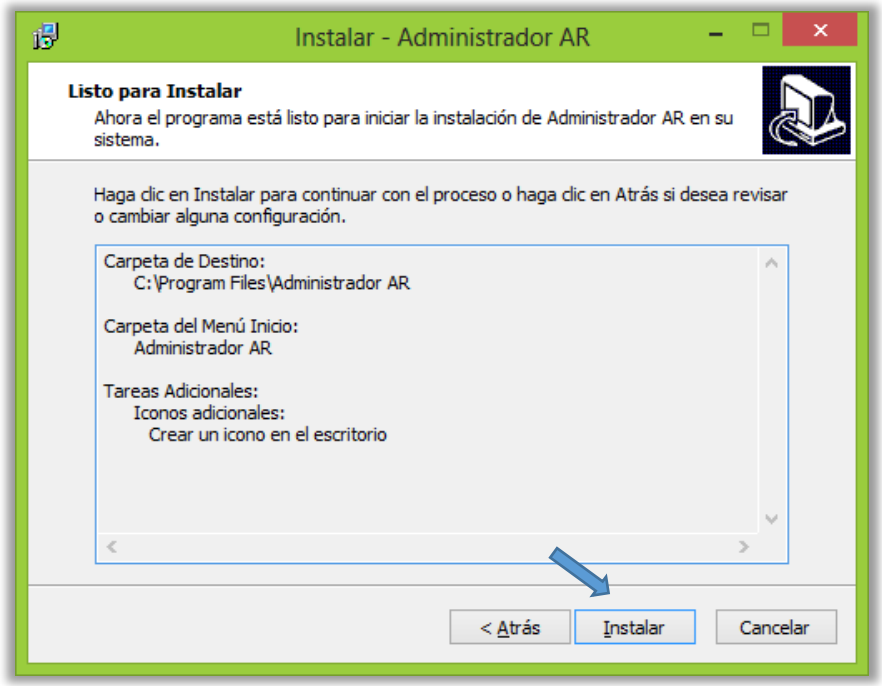

5. Posteriormente se puede seleccionar la carpeta de instalación del menú inicio o también se puede optar por no crear la carpeta de menú de inicio.




6. A continuación en la siguiente ventana se puede seleccionar si se desea crear un icono de acceso directo en el escritorio, y se debe dar clic en “Siguiente”.




7. Posteriormente se muestra un resumen con las tareas que realizara el instalador de Administrador AR, si todo esta correcto se procede a instalar dando clic en el botón “Instalar”.




8. Se espera hasta que termine la instalación y se da clic en Finalizar y automáticamente se ejecutara la aplicación “Administrador AR”.

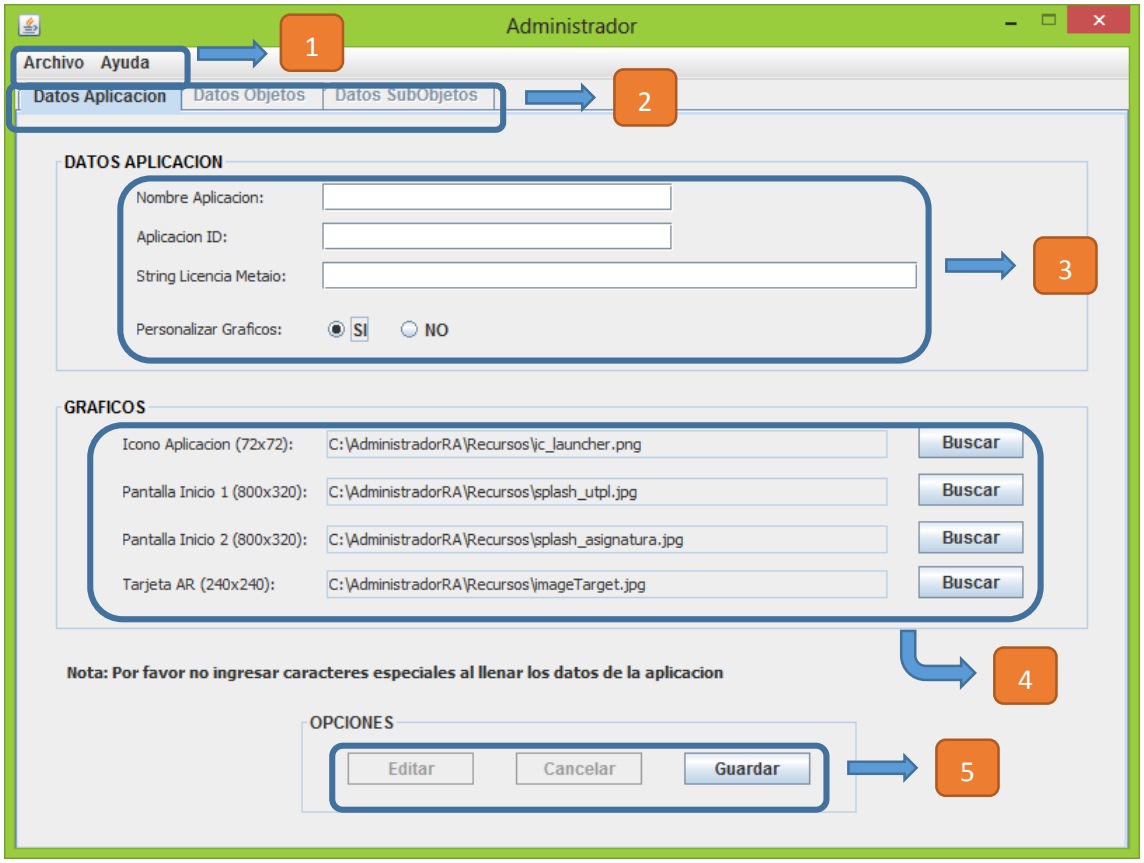
**Nota:** El instalador y todas las herramientas utilizadas en el presente manual serán proporcionadas en un cd adjunto al presente manual del programador.

## 3. DESCRIPCION DE LA INTERFAZ GRÁFICA DE USUARIO.

La aplicación de administración AR consta de una interfaz gráfica con una ventana principal, y dentro de esta un selector de paneles disponibles para el ingreso de la información requerida para la creación la aplicación de RA para Android.

### 3.1. Ventana Principal y Panel Datos Aplicación.

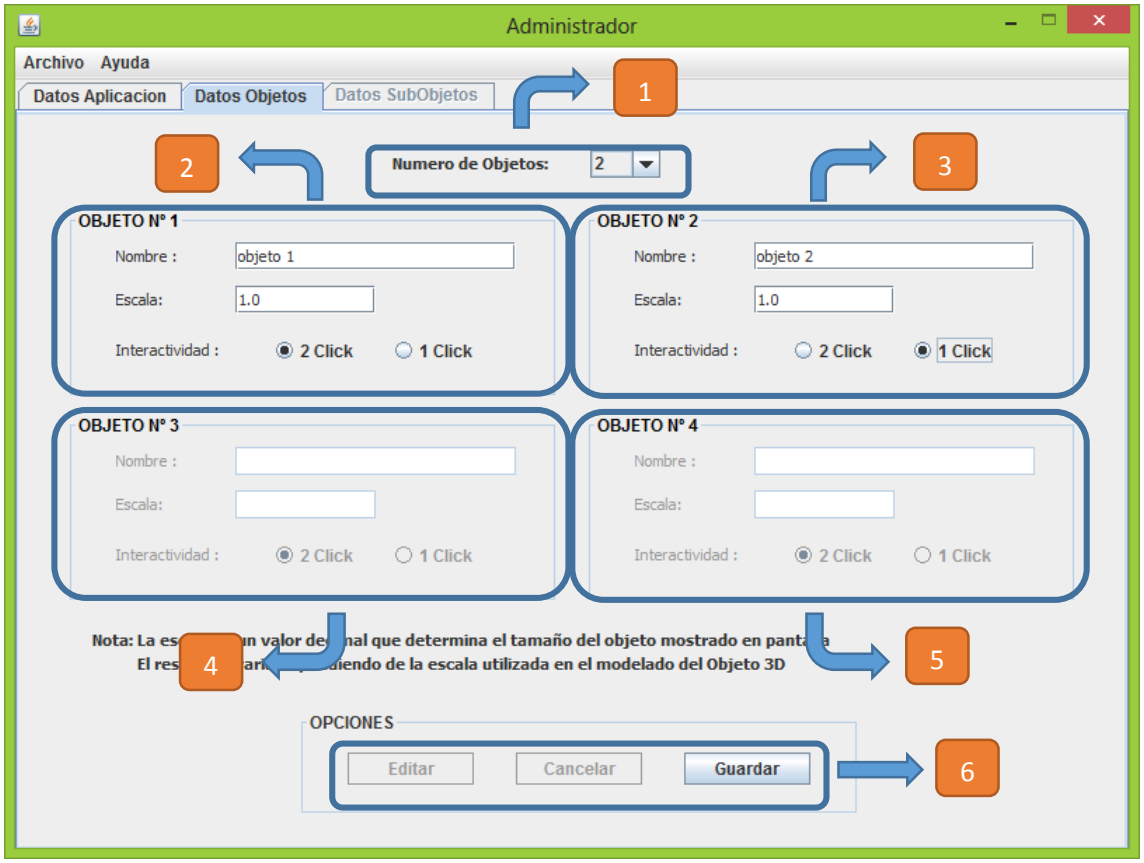
A continuación se detalla la ventana principal de la aplicación y el panel datos de la aplicación:



1. Barra de Menú (Archivo y Ayuda).
2. Selector de Paneles con 3 paneles (Datos Aplicación, Datos Objetos y Datos SubObjetos).
3. Datos principales de la aplicación (Nombre, Aplicación ID, String de Licencia Metaio).
4. Datos de Personalización (Icono, Tarjeta RA y Pantallas de carga 1 y 2).
5. Opciones de Aplicación (Editar, Cancelar y Guardar).

### 3.2. Panel Datos Objetos.

A continuación se detalla el panel datos de objetos:

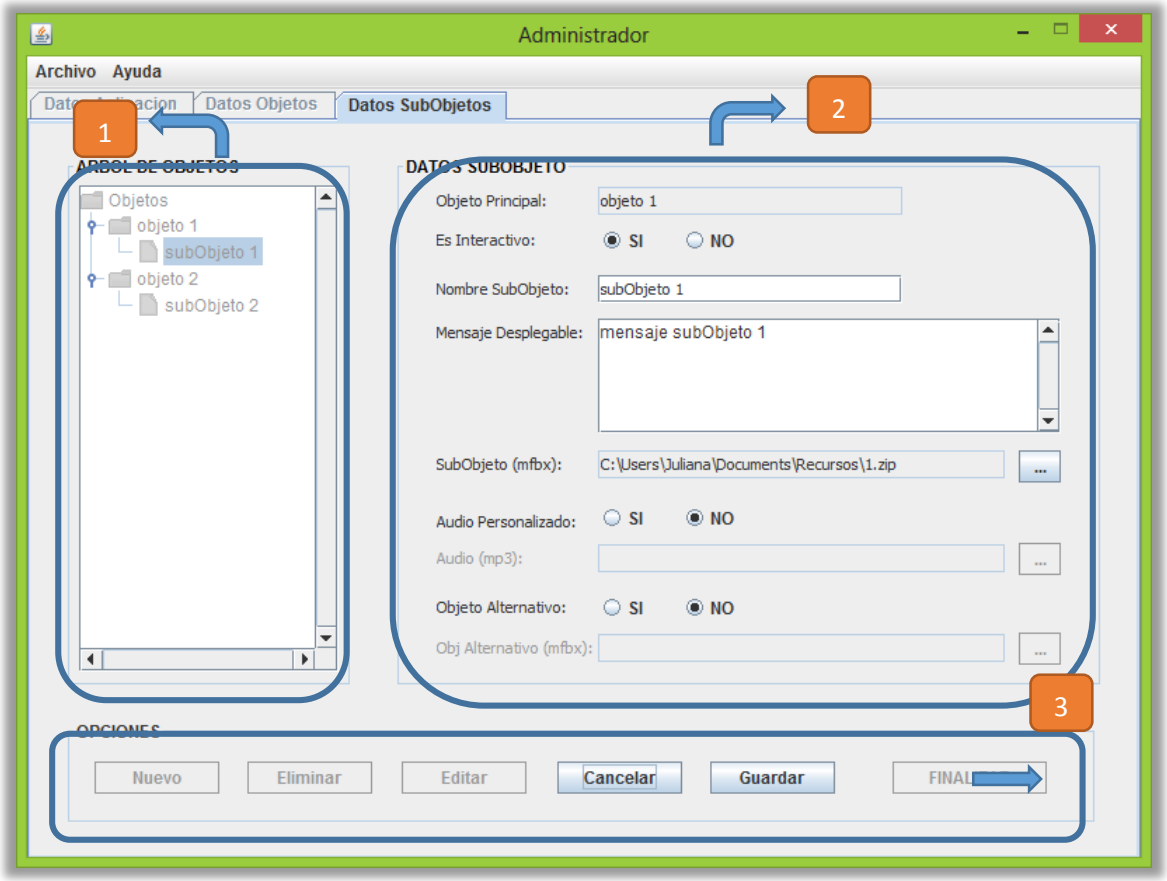



1. Selector de número de objetos principales (1 a 4).
2. Datos de Objeto 1 (Nombre, Escala, Interactividad).
3. Datos de Objeto 2 (Nombre, Escala, Interactividad).
4. Datos de Objeto 3 (Nombre, Escala, Interactividad).
5. Datos de Objeto 4 (Nombre, Escala, Interactividad).

6. Opciones de Objeto (Editar, Cancelar y Guardar).

### 3.3. Panel Datos SubObjetos.

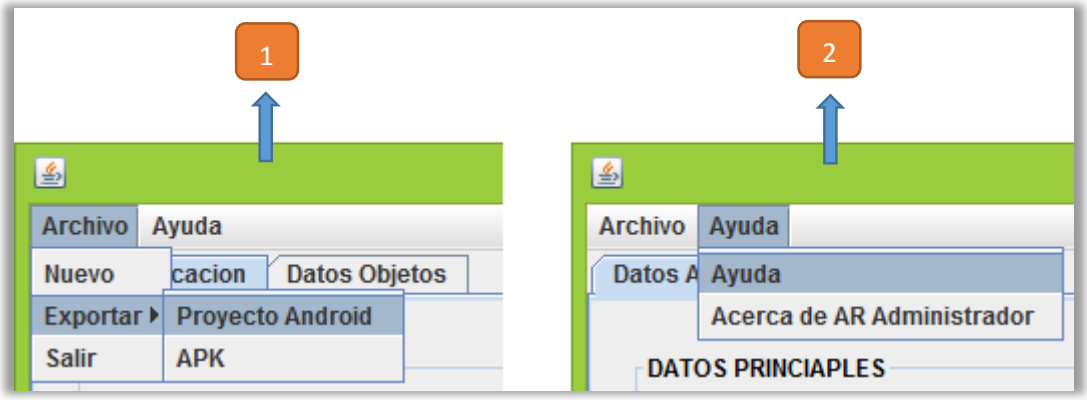
A continuación se detalla el panel datos de sub-objetos:




1. Estructura jerárquica en forma de árbol que facilita la navegación y administración objetos y sub-objetos.
2. Datos del sub-objeto (Interactividad, Nombre SubObjeto, Mensaje Desplegable, SubObjeto(mfbx), Audio(mp3), Obj Alternativo(mfbx)).
3. Opciones de los sub-objetos (Nuevo, Eliminar, Editar, Cancelar, Guardar y Finalizar).

### 3.4. Barra de Menú Principal.

A continuación se detalla la barra de menú principal:




1. Menú Archivo con opciones Nuevo, Salir y Exportar (Proyecto Android, APK (no implementada)).
2. Menú Ayuda que contiene el presente manual en el literal Ayuda y la opción “acerca de” la cual contiene datos informativos de la aplicación.

## 4. FUNCIONALIDADES.

Administrador AR permite crear una aplicación Android completamente personalizada tomando como base la aplicación de RA “RA-Desarrollo-Inteligencia”. A continuación se detalla el conjunto de funcionalidades presentes en la aplicación de administración:

- ✓ Cambiar Nombre, ID de aplicación y String de Licencia Metaio de la aplicación Andorid.
- ✓ Cambiar icono, pantallas de carga 1 y 2, y la tarjeta RA.
- ✓ Crear de 1 a 4 objetos principales, cada uno con su respectivo Nombre, Escala y su Interactividad (2 Clics o 1 Clic).
- ✓ Crear n SubObjetos de un Objeto Principal, cada uno con su respectiva Interactividad, Nombre SubObjeto, Mensaje Desplegable, SubObjeto(mfbx), Audio(mp3), Obj Alternativo(mfbx).
- ✓ Personalizar la interactividad, audio, objeto alternativo de cada SubObjeto dependiendo de las opciones seleccionadas por el usuario.
- ✓ Visualización de Objetos y Sub-Objetos mediante una estructura jerárquica en forma de árbol que facilita la administración de los mismos.
- ✓ Editar o Eliminar SubObjetos de cada objeto principal.

- ✓ Editar los datos de la aplicación y datos de objetos (si se editan los objetos principales se borraran todos los SubObjetos creados anteriormente).
- ✓ Permite exportar la aplicación personalizada como una carpeta de proyecto Android de Eclipse.

## 5. PREPARAR LOS CONTENIDOS PARA LA APLICACIÓN DE RA.

Antes de empezar con la creación de una aplicación de RA personalizada se debe tener todos los recursos necesarios para poder trabajar con el administrador. A continuación se detalla el conjunto de actividades a realizar antes de empezar a utilizar la aplicación:

- ✓ Características de objeto 3D para RA en Metaio.
- ✓ Creación de la escena de RA.
- ✓ Exportación de objetos 3D pre configurados.
- ✓ Conversión de Objetos al formato mfbx.
- ✓ Generar el String de licencia Metaio.

### 5.1. Características de objeto 3D para RA en Metaio.

Es importante recordar que los modelos de los objetos 3D deben cumplir una serie de requisitos o características, tal y como se menciona en el actual proyecto de fin de carrera, resumiendo dichas características se tiene:

- ✓ El modelo del objeto 3D debe estar en formato m2d, obj, dae o fbx.
- ✓ La textura del objeto debe ser trabajada en un archivo por separado en jpg o png y tener una resolución recomendada de 512x512 pixeles.
- ✓ Se recomienda que el modelo no sea excesivamente detallado y que no supere el tamaño de 5 mb.
- ✓ Que cada parte interactiva del objeto debe ser un objeto 3D independiente a nivel de modelado.

Una vez se obtiene los modelos de los objetos 3D y se comprueba que cumplen con los requisitos previos, es importante identificar si los subobjetos o partes que conforman un objeto principal se encuentran almacenadas en un solo archivo de modelado 3D, o cada parte, es un archivo independiente. Si todas las partes están almacenada en un solo archivo de modelado 3D con sus correspondientes texturas, se tiene una escena de RA preliminar; caso contrario,

si cada parte o subobjeto es un archivo independiente se debe crear una escena de RA con dichos subobjetos.

## 5.2. Creación de la Escena de RA.

Cuando cada parte de un objeto principal es un archivo de modelo 3D independiente (m2d, obj, dae o fbx y su correspondiente textura) es necesario hacer un pre-procesado de dichos objetos con el fin de asegurar que todos los sub-objetos que en conjunto forman un objeto principal estén correctamente ubicados con una escala y orientación adecuada, es decir, se creara una escena de RA. Este paso es de vital importancia, y repercute en la correcta visualización de los objetos 3D dentro de la aplicación de RA.

### 5.2.1. Instalación de 3DMAX.

Para esta tarea utilizaremos el software de renderización, animación y modelado 3D llamado 3DMAX el cual se lo puede descargar gratuitamente con una licencia académica o de estudio desde el siguiente enlace:

- ✓ http://www.autodesk.com/education/free-software/3ds-max?_ga=1.58011235.838361461.1429675015

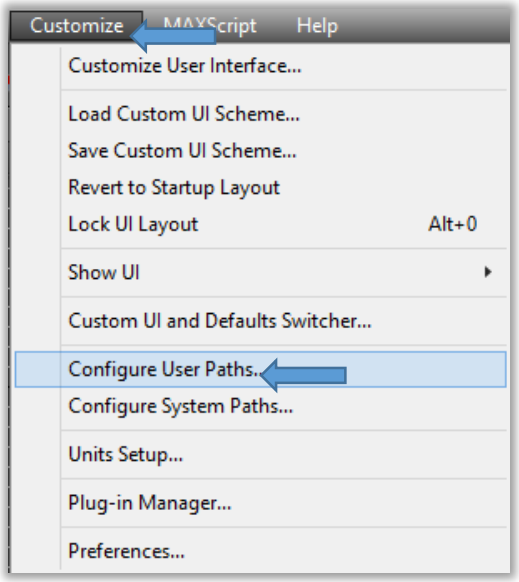
Y se puede instalar fácilmente mediante el siguiente tutorial proporcionado por la misma empresa desde el siguiente enlace:

- ✓ http://knowledge.autodesk.com/customer-service/installation-activation-licensing/install-configure/single-computer-installation/stand-alone-installation

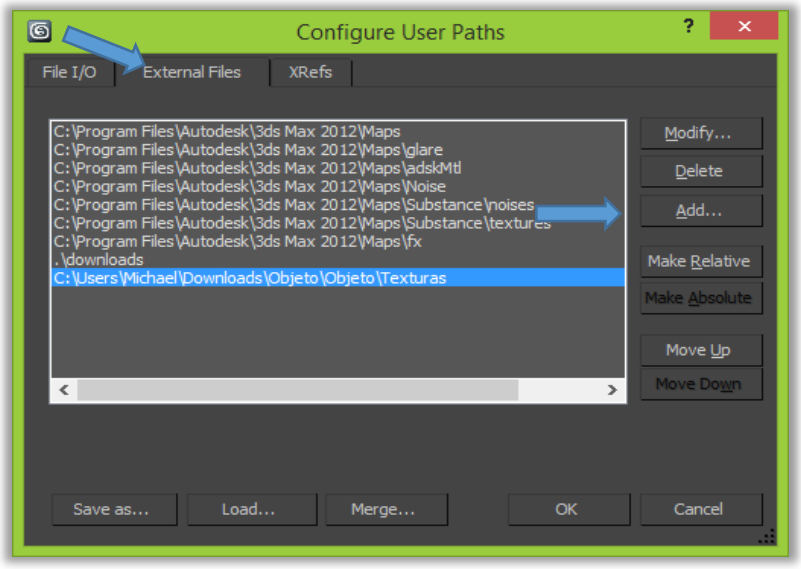
### 5.2.2. Establecer la ruta de las texturas de los modelos 3D.

Abrimos la aplicación de 3DMAX, y una vez cargado el programa se debe indicar la ruta o directorio donde se encuentran las texturas (jpg o png) de los objetos a cargar en la aplicación, para lo cual se realiza lo siguiente:

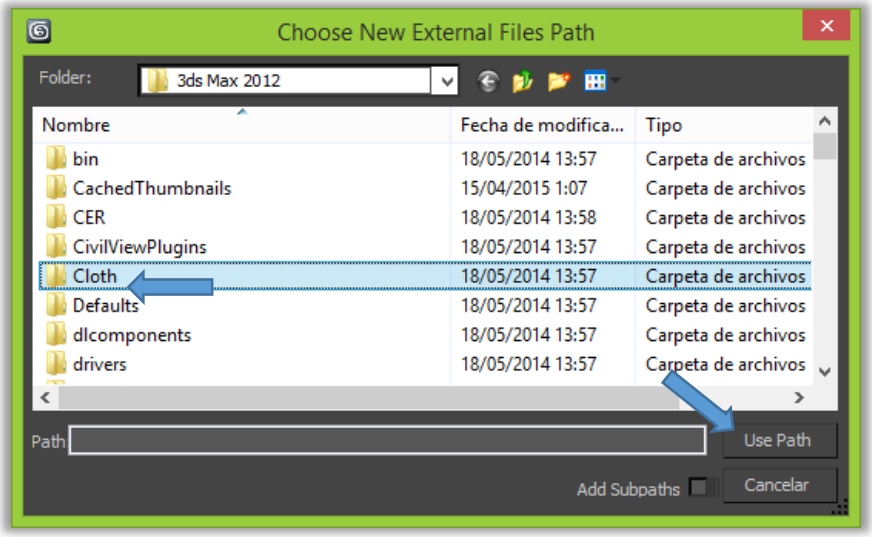
1. Primeramente en el menú principal se selecciona la opción “Customize” y posteriormente se debe seleccionar “Configure User Paths..”, de la siguiente forma:




2. Posteriormente aparecerá una ventana emergente donde se selecciona la pestaña “External Files” y después se da clic al botón Add.




3. Más tarde aparecerá una ventana donde se debe indicar la ruta de las texturas que utilizaran los objetos que se desea cargar en 3DMAX, una vez seleccionada la carpeta que contiene las texturas se da clic al botón “Use Path” tal y como se muestra en la siguiente imagen.

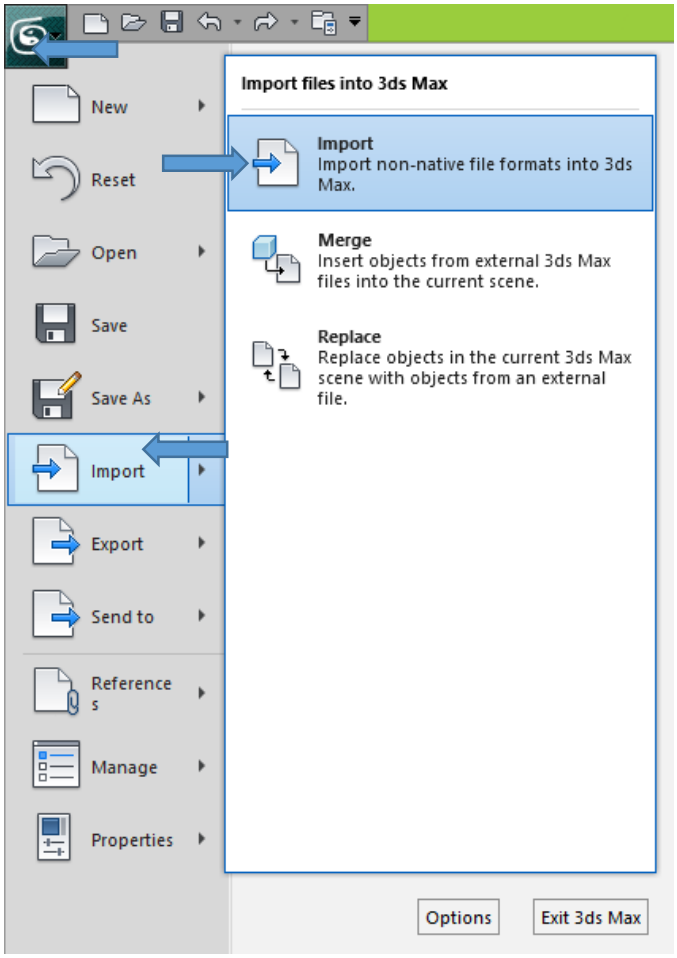



4. Por último se debe dar clic en “OK” en la ventana posterior de configuración de rutas de usuario y listo. Ahora 3DMAX ya sabe dónde buscar las texturas cuando se vaya a cargar el modelo de un objeto 3D.

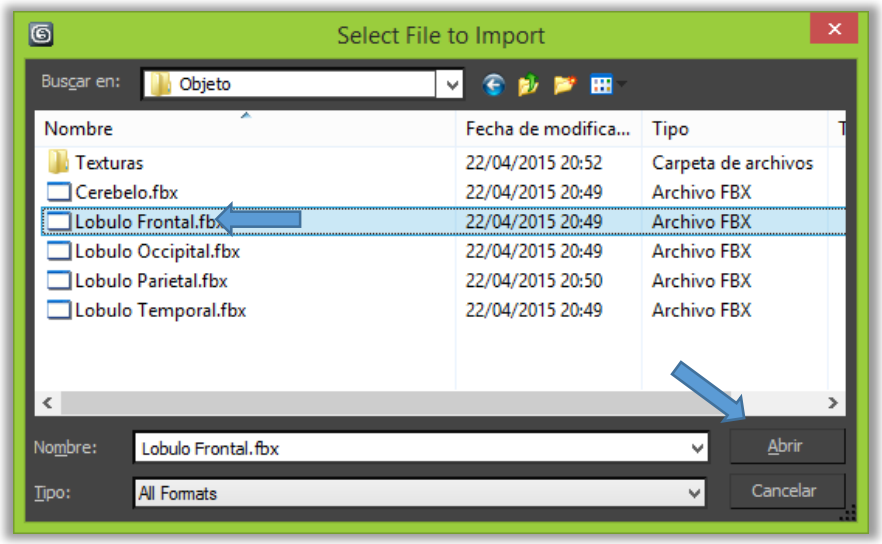
### 5.2.3. Cargar los Objetos en 3D MAX.

Independientemente de si los modelos de objetos están en un solo archivo o se encuentran en archivos por separado se procede a cargar el modelo o los modelos de los objetos 3D en 3DMAX de la siguiente forma.

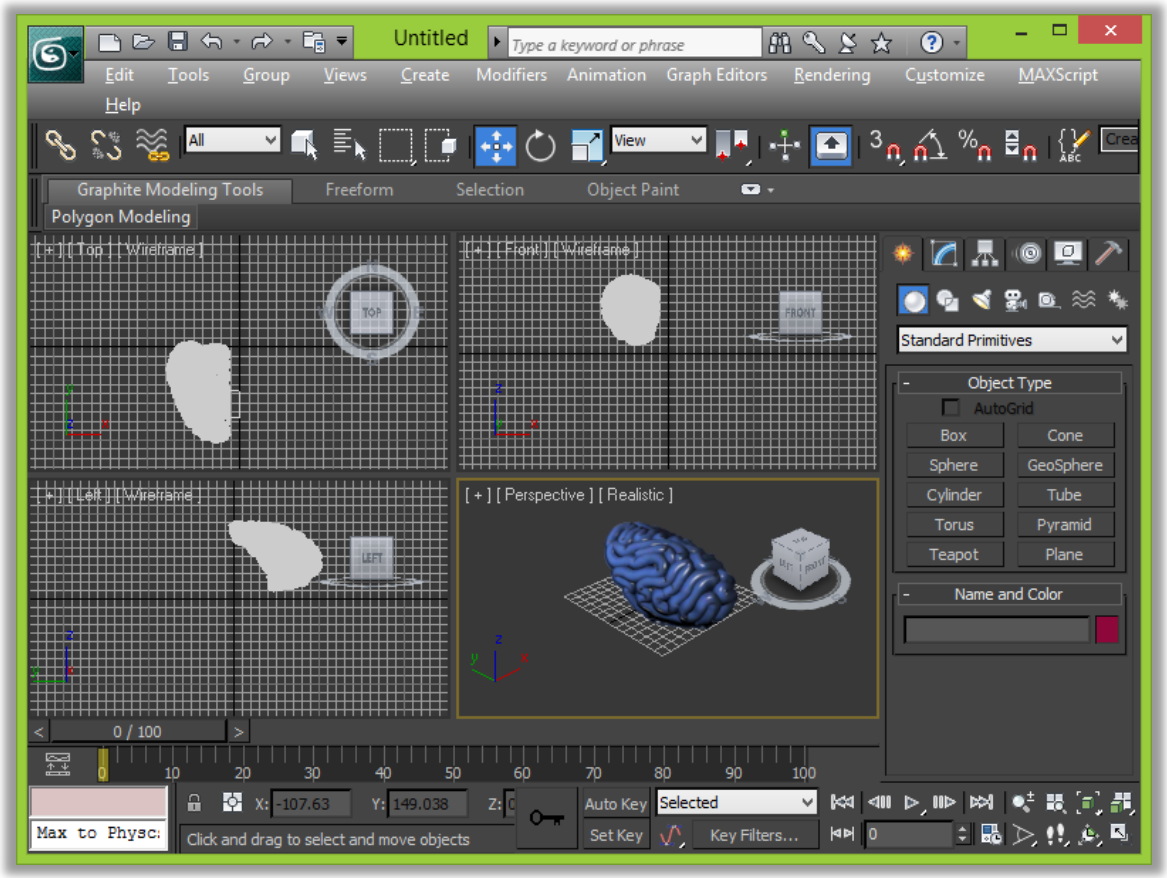
1. Se selecciona el menú a “Inicio” y luego a “Import” y posteriormente a “Import non-native file formats into 3dMax”, como se muestra a continuación:




2. Posteriormente aparecerá una ventana donde se debe buscar el modelo del objeto 3D guardado en el computador.



3. Por último, se da clic en abrir y 3DMAX procederá a cargar el modelo del Objeto 3D como se muestra a continuación.




4. Se repite los pasos del 1 al 3 para cargar todos los modelos de objetos 3D que se utilizaran para generar la escena de RA en caso de que cada subobjeto sea un archivo independiente.

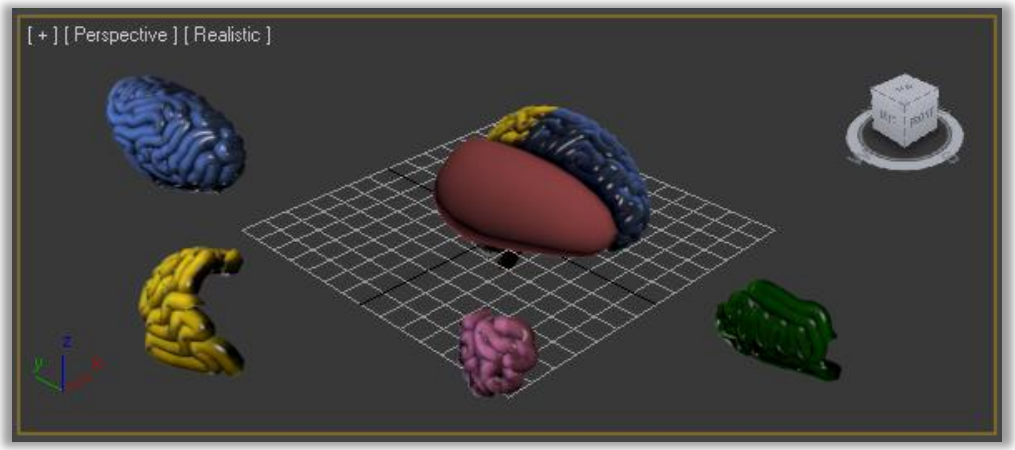
### 5.2.4. Configuración de objetos 3D en la escena de RA.

Una vez cargados todos los modelos que conforman un objeto principal se procede a ubicarlos, rotarlos, o redimensionarlos acorde a las necesidades de la aplicación. Cabe recalcar que si todos los objetos se encontraban guardados en un único archivo de modelado 3D, es posible que estos ya se encuentren correctamente posicionados dentro de la escena de RA, y no sea necesario realizar ajustes dentro de la misma.

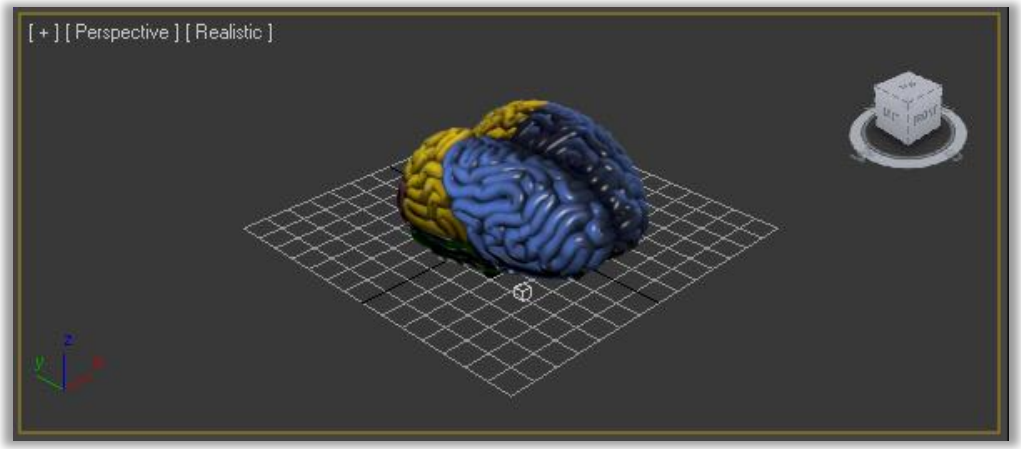
A continuación se detalla los controles de 3DMAX que se utilizaran para seleccionar, ubicar, rotar y redimensionar los objetos 3D, los mismos que se encuentran ubicados en la barra de opciones de objetos en la parte superior de la ventana justo debajo de la barra de menú.


- ✓ Para seleccionar los objetos se utiliza el siguiente botón 
- ✓ Para mover los objetos se utiliza el siguiente botón 
- ✓ Para rotar los objetos se utiliza el siguiente botón 
- ✓ Para redimensionar los objetos se utiliza el siguiente botón 

Ahora para utilizar las funciones simplemente se debe dar clic en el botón con la opción que se desee realizar y posteriormente se selecciona el objeto al que se desee mover, rotar, o redimensionar respectivamente. A través de estos controles es posible configurar la escena de RA a conveniencia, con lo cual se pasaría de una escena de RA como esta:

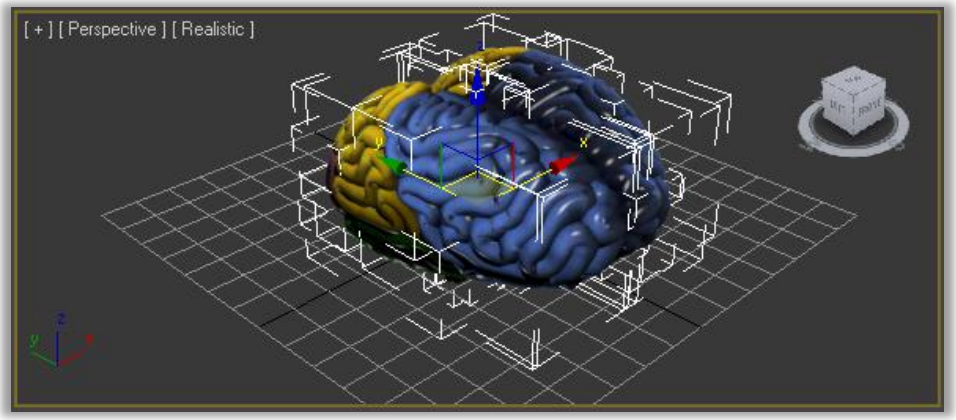



A una escena de RA donde todos los objetos se encuentran en su respectiva posición con un tamaño y orientación adecuado como la que se muestra a continuación.




Una vez que ya se tenga la escena de RA desarrollada es necesario mover la misma al punto central del eje de coordenadas xyz (0,0,0) con el fin de que al cargar la escena en la aplicación de RA esta sea mostrada justo en el centro de la tarjeta AR, para lo cual se debe realizar lo siguiente:

1. Primeramente se da clic en el botón mover  y luego seleccionamos toda la escena de RA, es decir, seleccionamos todos los objetos 3D.




2. Posteriormente en la barra inferior de opciones ponemos manualmente el punto x,y,z (0,0,0) y listo, con esto toda la escena tendrá como centro este punto.

**Nota:** Toda escena de RA debe ser generada tomando como referencia el punto central xyz (0,0,0) salvo que su interacción no lo requiera de esta forma. Recuerde que si la escena no tiene como centro la ubicación (0,0,0) al momento de generar el objeto 3D sobre la tarjeta RA esta no se mostrara en el centro de la misma.

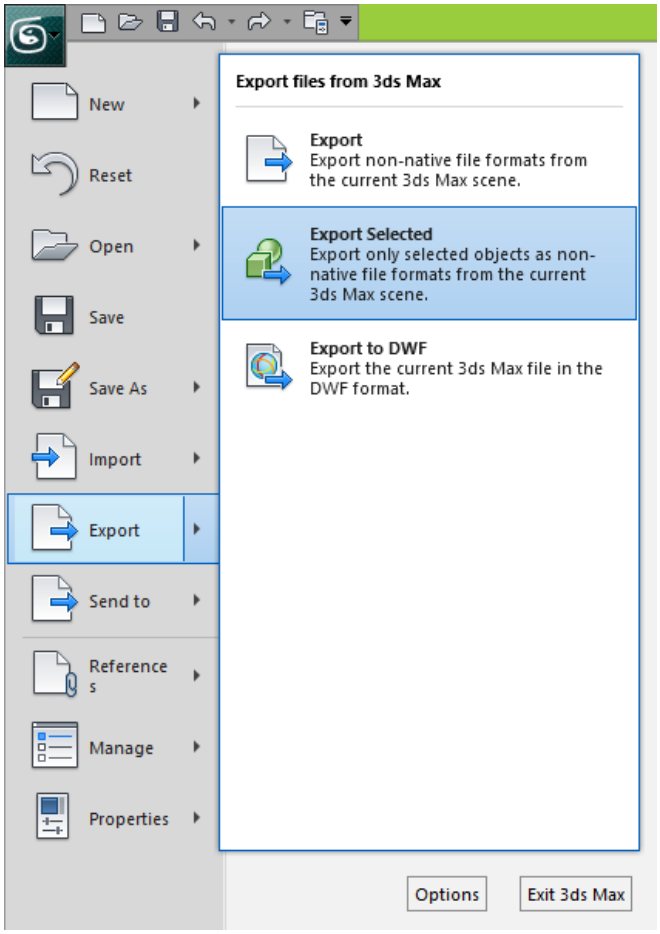
### 5.3 Exportación de objetos 3D pre configurados.

Por último, cuando la escena este completamente terminada y ubicada en el punto xyz (0,0,0) con todos los objetos en su respectivo lugar, procedemos a guardar esta propiedades dentro de cada uno de los subobjetos  para posteriormente exportarlos.

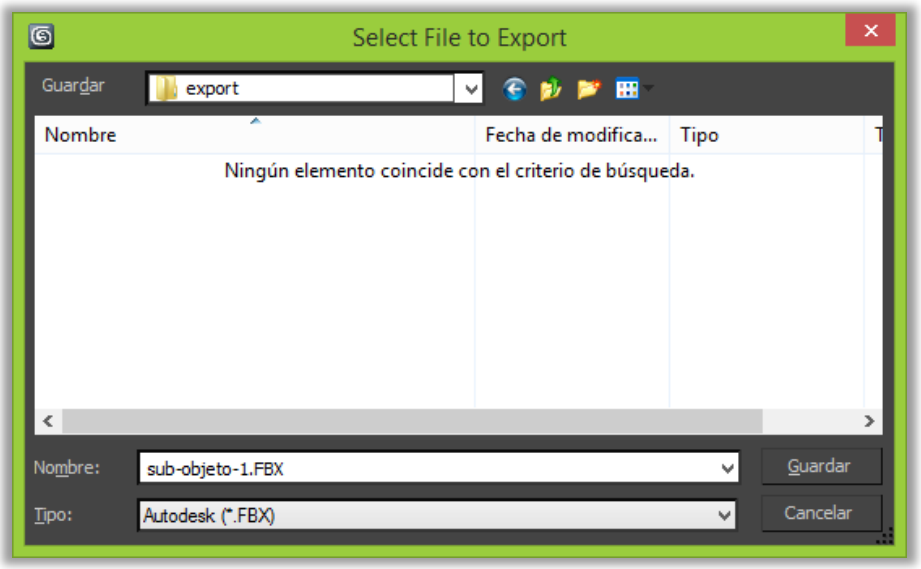
1. Primeramente se da clic en el botón de selección  y luego se debe seleccionar un subobjeto, que conforman el objeto principal o escena de realidad aumentada.

2. Posteriormente con el objeto seleccionado, se utiliza el botón  que se encuentra en la barra inferior de opciones y listo, el subobjeto guardara dentro de sus propiedades la ubicación, orientación y tamaño cuando sea exportado.

3. Posteriormente se procede a exportar el subobjeto a un archivo independiente, para lo cual, con el subobjeto seleccionado vamos al menú “Inicio”, luego a “Export” y por último a “Export Selected”.




4. Por ultimo en la siguiente ventana se selecciona donde se guardara el subobjeto 3D y posteriormente escribimos el nombre del archivo y seleccionamos el formato (en este caso se selecciona fbx), tal y como se muestra a continuación.



5. Por último se realiza el mismo procedimiento con cada uno de los subobjetos de la escena de RA y los se los guarda a todos en una misma carpeta con sus respectivas texturas.

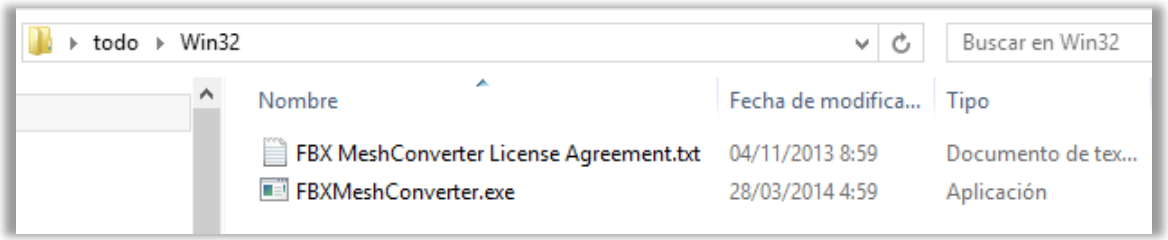
### 5.4. Conversión de Objetos al formato mfbx.

A pesar que Metaio SDK provee soporte para diversos formatos de objetos 3D como: m2d, obj, dae o fbx, es recomendable transformar dichos objetos al formato mfbx, el cual, brinda un mayor rendimiento por ser un formato propio y optimizado de Metaio. Para esta tarea se realiza lo siguiente:

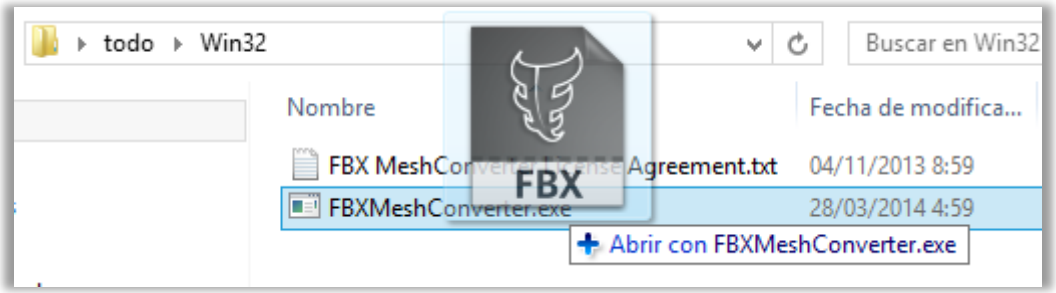
1. Primeramente se descarga la aplicación FBXMeshConverter del siguiente enlace:

   http://dev.metaio.com/content-creation/3d-animation/format/fbx/


2. Una vez descargada la aplicación se procede a descomprimir dicho archivo quedando de la siguiente forma.

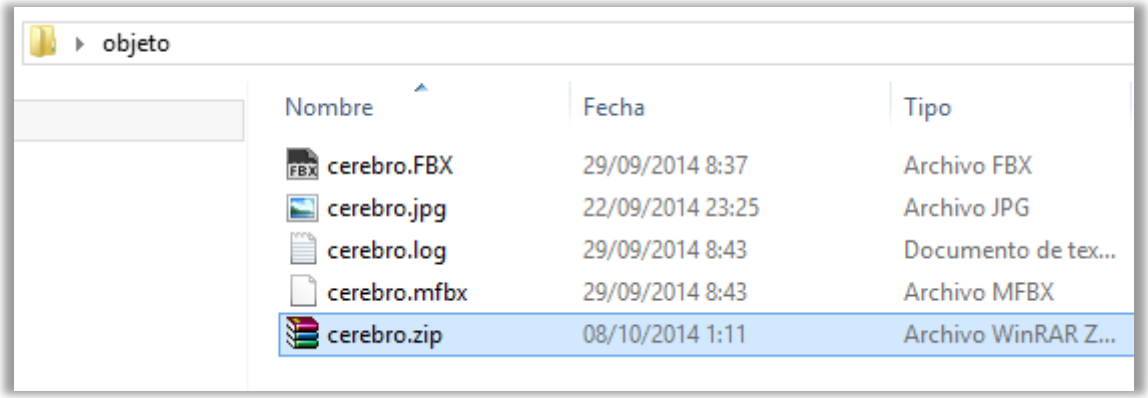


3. A continuación solo se debe arrastrar el objeto que se desea transformar sobre el archivo FBXMeshConverter.exe como se muestra a continuación:




**Nota:** dentro de la carpeta que contiene el archivo del objete 3D debe también encontrarse su respectiva textura en formato jpg o png, caso contrario el objeto se transformara sin su textura correspondiente.

4. Esperamos que el proceso termine y al finalizar se generaran los siguientes archivos dentro de la carpeta de origen del archivo fbx de la siguiente forma:




5. Como se observa se obtiene un archivo .zip con el mismo nombre del objeto de origen en este caso “cerebro.zip”, el cual contiene el modelo en formato mfbx y su correspondiente textura empaquetada en un único archivo comprimido. Este es el archivo de objeto que utilizaremos dentro de la aplicación de RA.

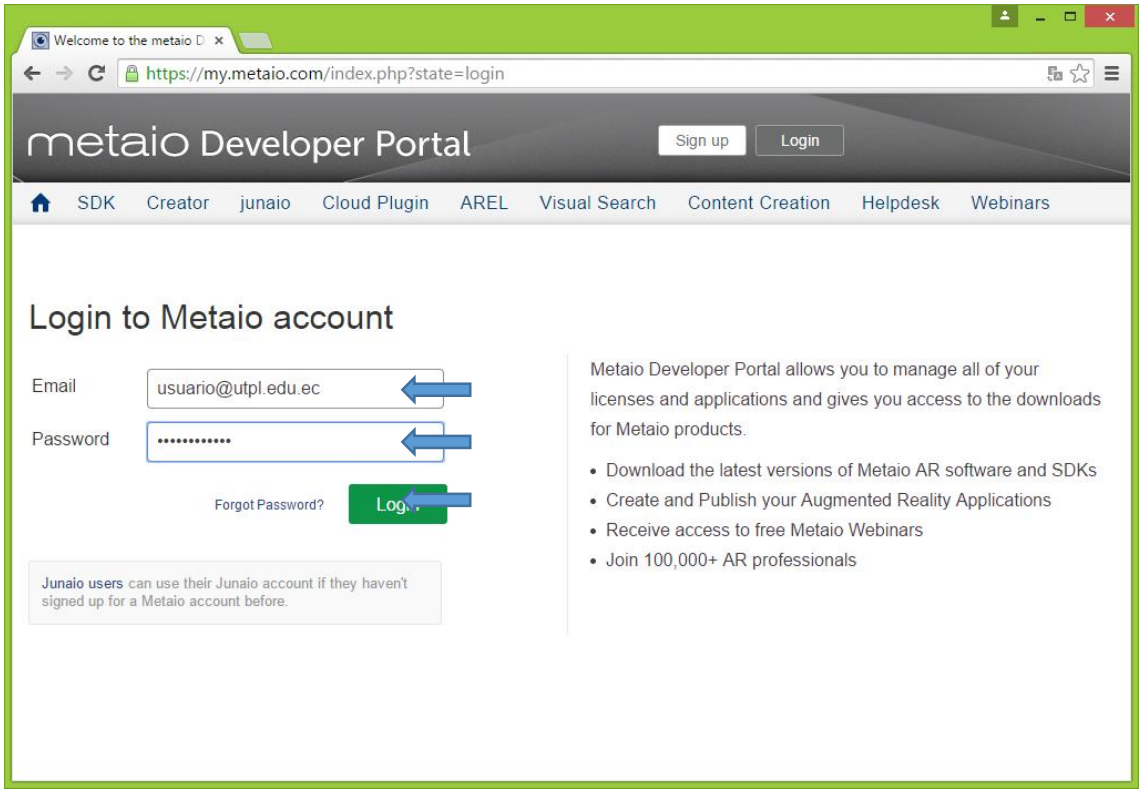
Si se desea comprobar que los objetos han sido correctamente transformados a mfbx se lo puede realizar con la ayuda de Metaio Creator, la cual se puede descargar e instalar desde los siguientes enlaces:

- ✓ **Instalador:** http://ar.metaio.com/download_creator
- ✓ **Manual de Instalación:** https://dev.metaio.com/creator/getting-started/system-requirements-and-installation/windows/
- ✓ **Manual de Funcionamiento:** https://dev.metaio.com/creator/tutorials/create-your-first-ar-scenario/

### 5.5. Generar el String de Licencia Metaio.

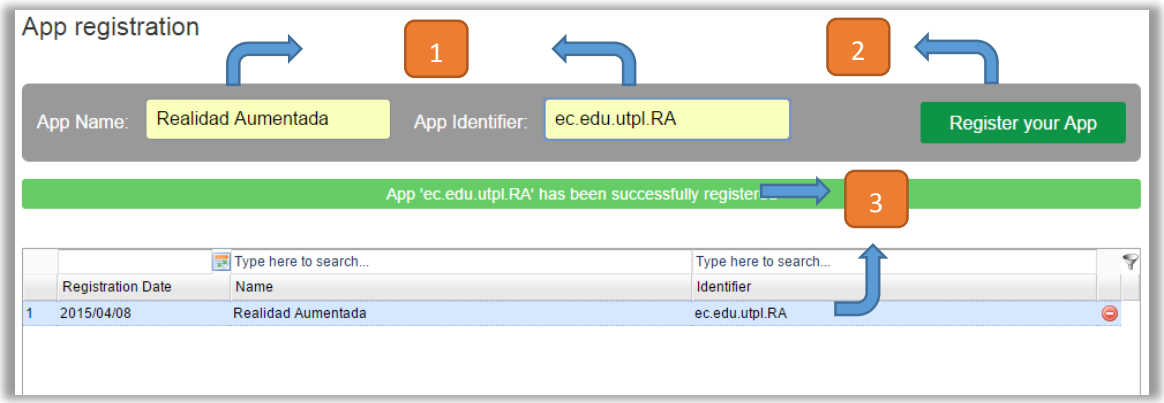
Por último se debe registrar el Nombre de la Aplicación y el ID de aplicación en Metaio Developer Portal para generar el String de Licencia de Metió para dicha Aplicación, a continuación se procede de la siguiente forma:

1. Se debe abrir el siguiente enlace y se procede a ingresar las credenciales obtenidas al comprar la Licencia del SDK de Metaio.
   https://my.metaio.com/index.php?state=login



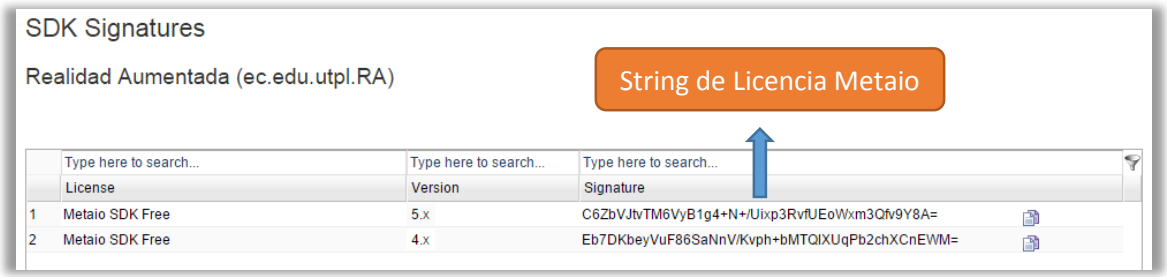
2. Una vez que se ha ingresado con nuestras credenciales seleccionamos en el menú principal la opción “My Apps”.


3. Se ingresa el App Name y el App Identifier y luego se da clic en Register your App y todo ha salido correctamente debe aparecer el siguiente mensaje “App 'App Name' has been successfully registered” y agregarse la aplicación en la tabla.




   a. **App Name:** Se Ingresa el Nombre de a aplicación (Ejm: Realidad Aumentada)
   b. **App Identifier:** Se ingresa el ID o identifico de la aplicación, el cual debe ser único e irrepetible para cada aplicación. Normalmente se utiliza la nomenclatura inversa de la página de la empresa más el nombre de la aplicación o sus siglas (Ejm: ec.edu.utpl.RA).

4. Una vez registrada la aplicación en Metaio copiamos la licencia de acuerdo a la versión de Metaio que se está trabajando (en nuestro caso es la versión 5) y la guardamos para su posterior uso en la aplicación de administración.




## 6. CREACIÓN DE UNA NUEVA APLICACIÓN DE RA PERSONALIZADA.

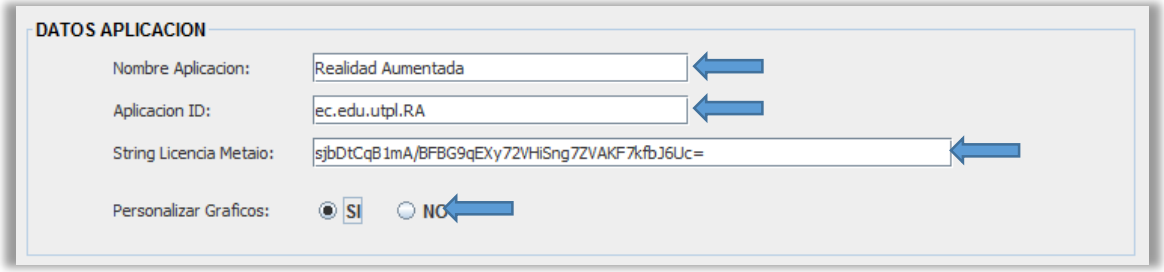
Como se mencionó anteriormente la aplicación de administración permite generar aplicaciones de RA personalizada tomando como base la aplicación de RA para el componente académico “Desarrollo de la Inteligencia” de la UTPL, dicha aplicación de administración permite facilitar las tareas administrativas a través de una amigable interfaz gráfica cuya principal ventaja es que no requiere ningún conocimiento de programación por parte del administrador, exceptuando la compilación de la aplicación para generar el archivo de aplicación android APK, el cual se debe realizar una vez generado el proyecto con la aplicación de administración.

A continuación se detalla el proceso de creación de una nueva aplicación de RA personalizada paso a paso.

### 6.1. Ingreso Datos de Aplicación.

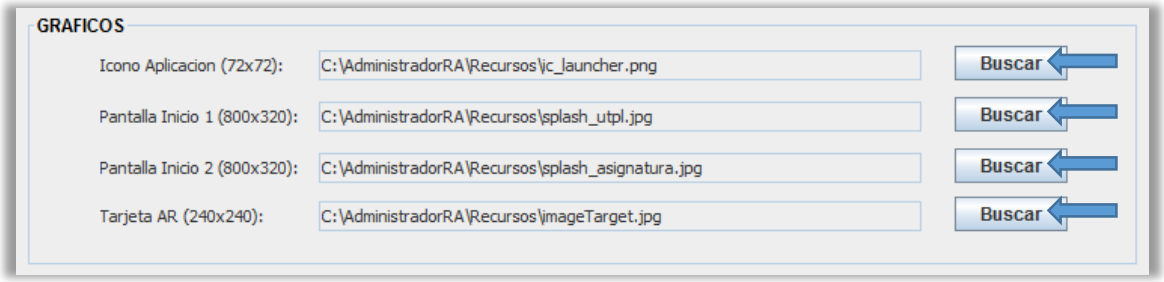
Se debe ejecutar la aplicación de Administrador llamada “Administrador AR” y a continuación se debe dirigir al menú principal, donde se selecciona el menu “Archivo” y luego la opción “Nuevo” y de esta manera se habilitara el panel de ingreso de datos de la aplicación.

1. A continuación en el panel “Datos Aplicación” se procede a ingresa los datos principales de la aplicación:




   a. **Nombre Aplicación:** Se ingresa el nombre de a aplicación utilizado al registrar la aplicación en Metaio (Realidad Aumentada).
   b. **Aplicación ID:** Se ingresa el ID de la aplicación utilizado al registrar la aplicación en Metaio Developer Portal (Ejm: ec.edu.utpl.RA).
   c. **String de Licencia de Metaio:** Se ingresa el String de Licencia de Metaio generada al registrar la aplicación en Metaio.
   d. **Personalización de Grafico (SI/NO):** Seleccionamos “SI”, si se desea personalizar los gráficos de la aplicación y de esta manera se habilitara el panel destinado para este propósito, caso contrario se selecciona “NO” y la aplicación utilizara los gráficos por defecto.

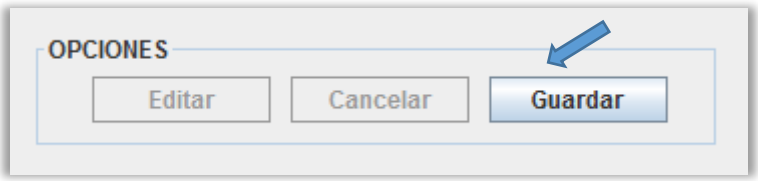
2. A continuación si se ha seleccionado la opción de personalizar gráficos se habilita el panel donde se podrá cargar los archivos de personalización como se muestra a continuación.




   a. **Icono de Aplicación (72x72):** Es el icono de la aplicación que se mostrara en el dispositivo móvil, se debe seleccionar un icono que este en formato PNG o JPG con una resolución mínima de 72x72 pixeles.

b. **Pantalla de Inicio 1(800x320):** Es la primera pantalla de carga de la aplicación la cual se muestra al iniciar la aplicación, se debe seleccionar un icono que este en formato PNG o JPG con una resolución mínima de 800x320 pixeles.
c. **Pantalla de Inicio 2(800x320):** Es la segunda pantalla de carga de la aplicación la cual se muestra después de la primera pantalla de carga, se debe seleccionar un icono que este en formato PNG o JPG con una resolución mínima de 800x320 pixeles.
d. **Tarjeta AR (240x240):** Es la tarjeta de RA el cual se utilizara dentro de la aplicación como marcador para generar la escena de RA, se debe seleccionar un icono que este en formato PNG o JPG con una resolución mínima de 240x240 pixeles.

3. Por último se da clic a la opción guardar con lo cual se guardara los datos de la aplicación y se habilitara el panel para el ingreso de datos de los objetos principales.

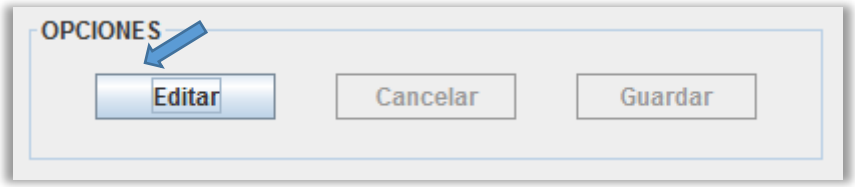



**Nota:** Al llenar los datos de la aplicación evite usar caracteres especiales en los nombres tales como (“,” - ”;” - “ñ” - “}” - “+” - “*” - “/” , etc.) para evitar errores en la generación de la aplicación, así mismo, evite usar espacios.

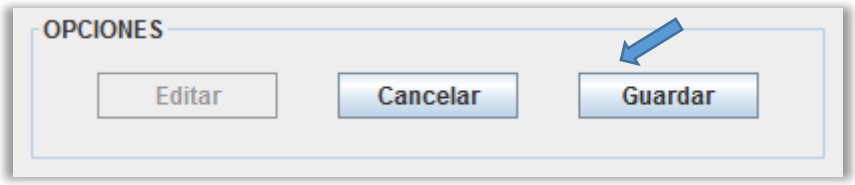
### 6.2. Edición Datos de Aplicación:

Después que se guardan los datos principales de la aplicación es posible editarlos en cualquier momento para lo cual nos situamos en el panel “Datos Aplicación” y se realiza los siguientes pasos.

1. Se da clic en el botón “Editar” con el cual se habilitara el panel de datos de aplicación permitiendo editar los datos ingresados.



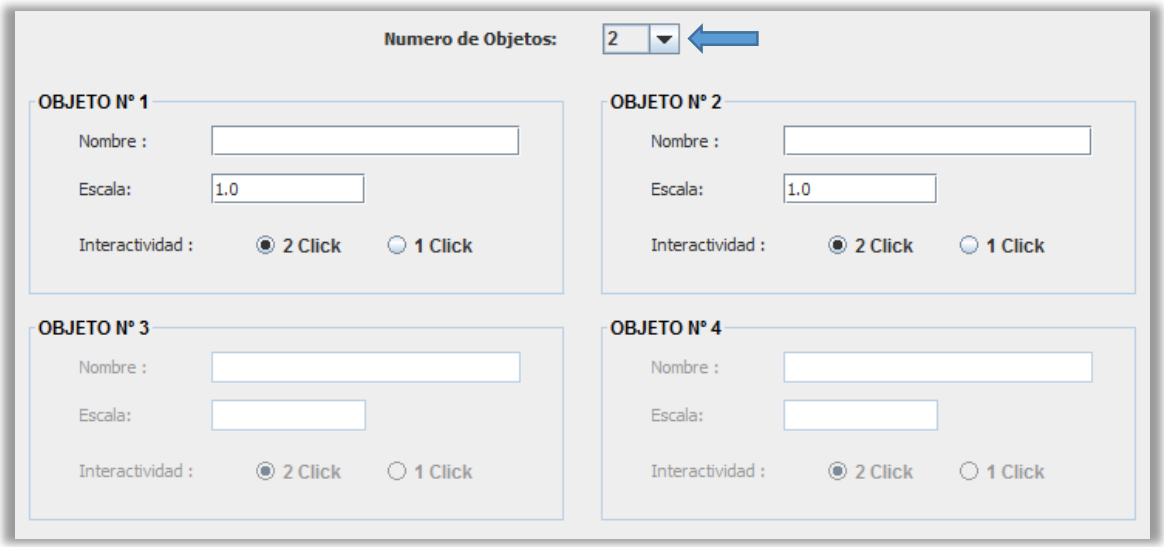

2. Una vez editados los datos de la aplicación el sistema nos brinda dos opciones: guardar los cambios o cancelar, si se desea guardar los cambios realizados se da clic en Guardar, caso contrario Cancelar.



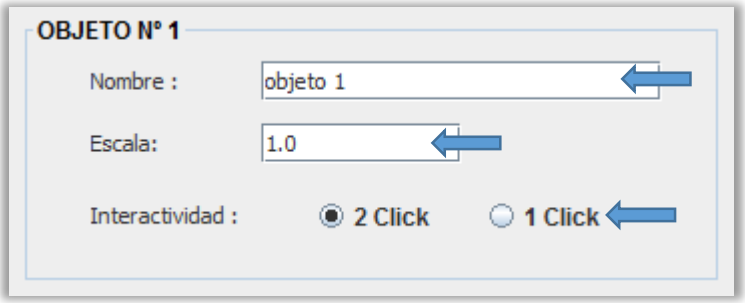

### 6.3. Ingreso Datos de Objetos.

Para poder ingresar los datos de los objetos, previamente se debe haber llenado los datos de la aplicación en el panel “Datos Aplicación” una vez guardados dichos datos se habilitara el panel “Datos Objetos”. Para el ingreso de los datos de los objetos se realiza lo siguiente:

1. Se selecciona el Panel “Datos Objetos” y posteriormente se selecciona el número de objetos que se desea en la aplicación, para lo cual se debe seleccionar en el cuadro de texto desplegable el número de objetos que tendrá la aplicación, el cual puede ser de 1 a 4, dependiendo del número seleccionado se habilitara las regiones para el ingreso de datos de cada Objeto Principal.

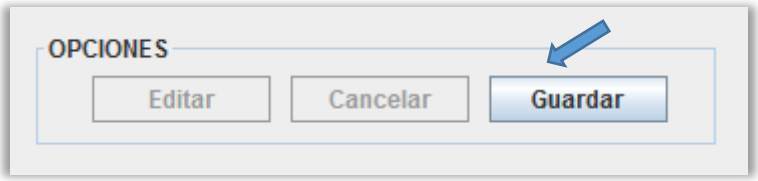



2. Seguidamente de acuerdo al número de objetos seleccionado se procede a llenar los datos correspondientes para cada objeto. Recuerde que cada objeto principal consta de n subobjetos.




   a. **Nombre:** Se ingresa el nombre de cada objeto principal, este nombre es el que se visualizara en lo botones de cambio de objeto (máximo 20 caracteres).
   b. **Escala:** se debe ingresar un valor decimal el cual representa el tamaño del objeto en pantalla, el resultado puede variar dependiendo de la escala utilizada en el modelado del objeto 3D (normalmente dejar en 1.0).
   c. **Interactividad:** Puede ser de 1 Clic o 2 Clics, define cuantos clics se necesitan para separar el subobjeto del objeto principal.

3. Por último se da clic a la opción guardar con lo cual se guardara los datos de los objetos y se habilitara el panel para el ingreso de subobjetos.

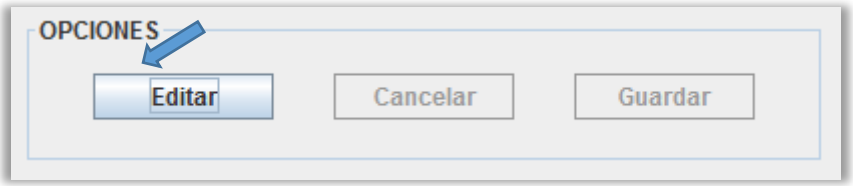



**Nota:** Al ingresar el valor decimal de la escala evite utilizar valores demasiado pequeños o sumamente altos.

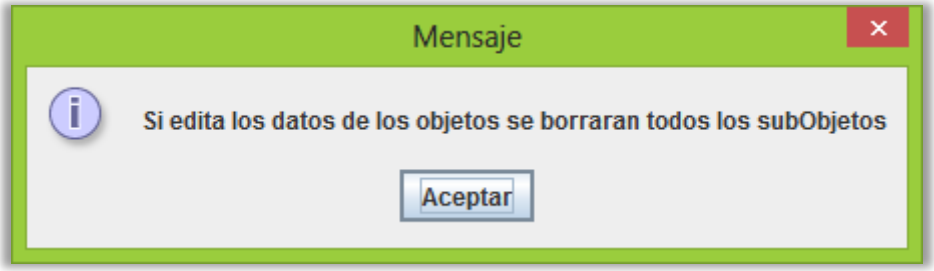
### 6.4. Editar Datos de Objetos.

Después que se han guardado los datos principales de los objetos es posible editarlos en cualquier momento, sin embargo se debe tener especial cuidado, pues esta opción borrara todos los datos de subobjetos que se hayan ingresado hasta el momento. Para editar los objetos, nos situamos en el panel “Datos Objetos” y se realiza los siguientes pasos:

1. Se da clic en el botón “Editar” con el cual se puede editar los datos de la aplicación ingresados

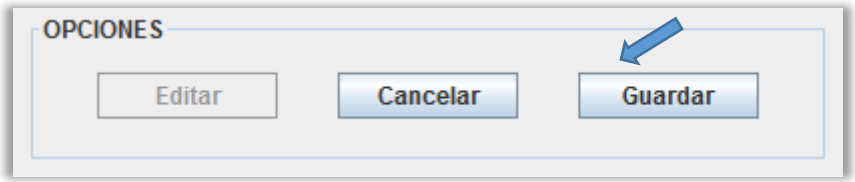



2. Aparecerá un mensaje de advertencia indicando que, si se editan lo datos de los objetos se borraran todos los subobjetos, se acepta el mensaje y se continua.




3. Seguidamente se procede a editar los datos de los objetos y el sistema nos brinda dos opciones: guardar los cambios o cancelar, si se desea guardar los cambios

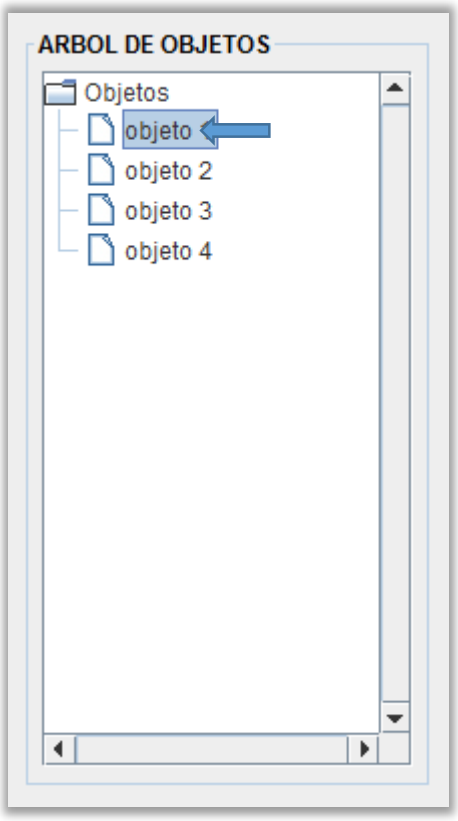
realizados se da clic en Guardar (se borraran todos los subobjetos creados hasta el momento), caso contrario Cancelar (los subobjetos no tendrán ningún cambio).



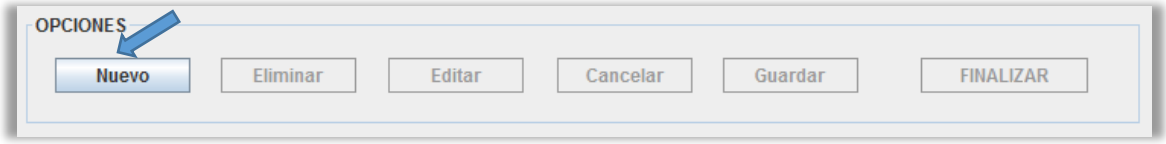

### 6.5. Ingreso Datos de Sub-Objetos.

Para poder ingresar los datos de los subobjetos, previamente se debe haber llenado los datos de los objetos en el panel “Datos Objetos” una vez guardados dichos datos se habilitara el panel “Datos SubObjetos”. Para el ingreso de los datos de los subobjetos se realiza lo siguiente:

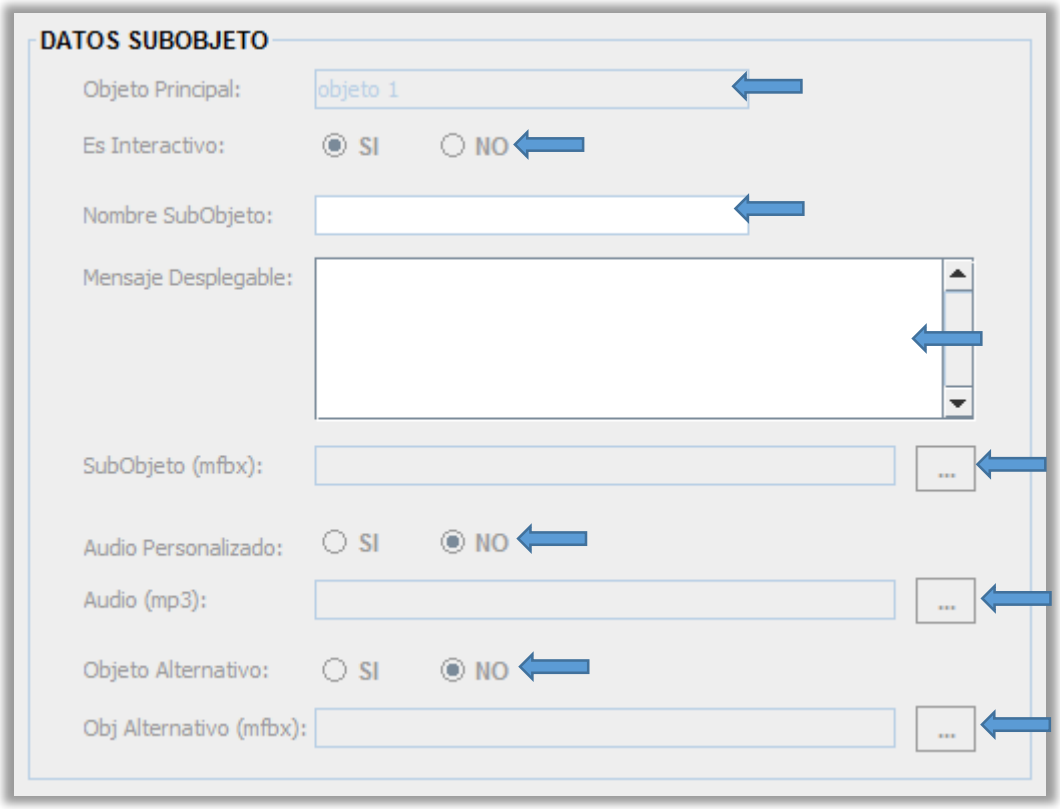
1. Se selecciona el Panel “Datos SubObjetos” y posteriormente se procede a seleccionar el objeto principal al cual se le desea agregar subobjetos en la estructura del árbol de objetos, de la siguiente forma:



2. Una vez seleccionado el objeto principal se habilitara en el panel de opciones el botón “Nuevo”, con el cual se puede ingresar un nuevo subobjeto para el objeto principal seleccionado.




3. Seguidamente se da clic en el botón “Nuevo” con lo cual se habilitara el panel para el ingreso del nuevo subobjeto.




a. **Objeto Principal:** Muestra el Objeto principal al cual se le agregara el nuevo subobjeto.
b. **Es Interactivo:** Se selecciona “SI”, si el objeto es interactivo, es decir podemos darle clic e interactuar con él, caso contrario se selecciona “NO”, si el objeto es estático y únicamente se podrá ingresar el nombre del subobjeto y el archivo que contiene el modelo de dicho subobjeto.
c. **Nombre SubObjeto:** Se ingresa el nombre que tendrá el subobjeto, este nombre aparecerá en el cuadro de información desplegable que aparece al dar clic al subobjeto actual en la aplicación de RA.

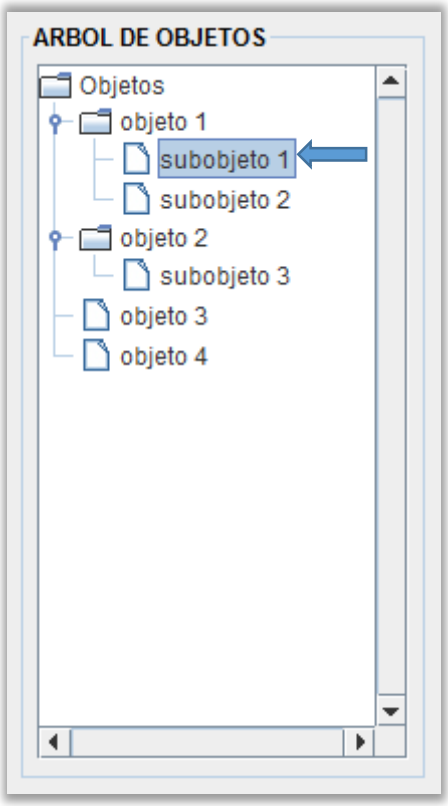
d. **Mensaje Desplegable:** Se ingresa el mensaje que aparecerá en el cuadro de información desplegable que aparece al dar clic al subobjeto actual en la aplicación de RA.
e. **SubObjeto (mfbx):** Se debe ingresar el modelo del objeto 3D en formato zip, es decir el formato de archivo comprimido del modelo del subobjeto que se generó al transformar el objeto de fbx a mfbx con el FBXMeshConverte.
f. **Audio Personalizado:** Si se selecciona “SI” se habilitara la opción de ingresar el audio que se reproducirá en la aplicación de RA para el subobjeto actual, caso contrario se selecciona “NO” y la aplicación de RA leerá el texto del Mensaje Desplegable.
g. **Audio (mp3):** Si se ha selecciona la opción “SI” de Audio Personalizado se debe ingresar el audio en formato mp3 que se reproducirá en la aplicación de RA al interactuar con el subobjeto actual.
h. **Objeto Alternativo:** Si se selecciona “SI” se habilitara la opción de ingresar el objeto alternativo que se mostrara en la aplicación de RA al interactuar con el subobjeto actual, caso contrario se selecciona “NO” y la aplicación mostrara el mismo sub-objeto al interactuar con él.
i. **Obj Alternativo (mfbx):** Si se ha selecciona la opción “SI” de Objeto Alternativo se debe ingresar el modelo del objeto 3D en formato zip.

**Nota:** Evite usar archivos de audio de larga duración y con bitrate mayor a 128 kbps con el fin de evitar la sobrecarga de la aplicación.

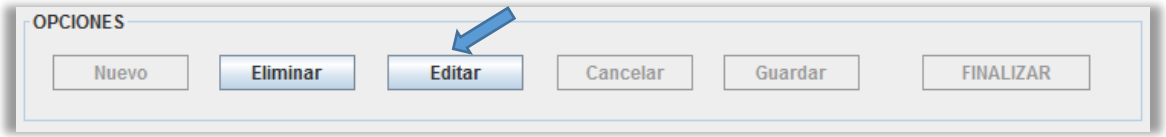
### 6.7. Edición de SubObjetos.

Una vez creados los subobjetos es posible editarlos o eliminarlos en cualquier momento, Para editar los subobjetos la cual se realiza lo siguiente:

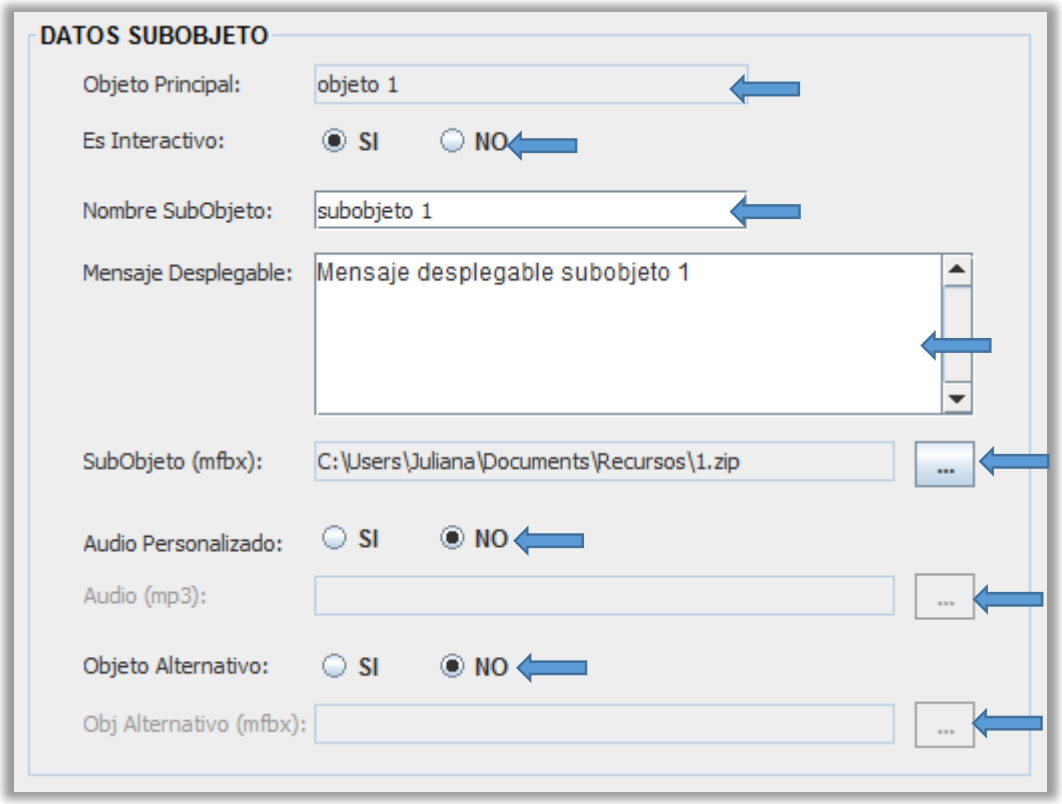
1. Primeramente se procede a seleccionar el subobjeto al cual se desea editar en la estructura del árbol de objetos, de la siguiente forma:



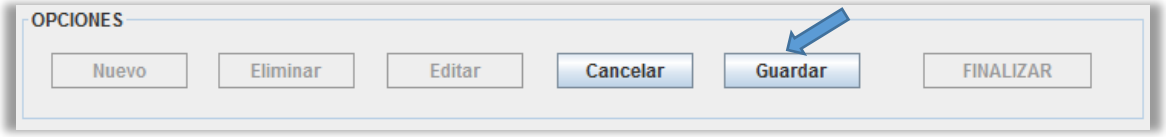

2. Una vez seleccionado el sub-objeto se habilitara en el panel de opciones el botón “Editar” y “Eliminar”, con los cuales se puede editar o eliminar el subobjeto seleccionado. Se selecciona la opción “Editar” en el panel de opciones de subobjetos.




3. Inmediatamente se habilitara y se cargara la información del subobjeto seleccionado el panel de “Datos SubObjeto” para su edición.



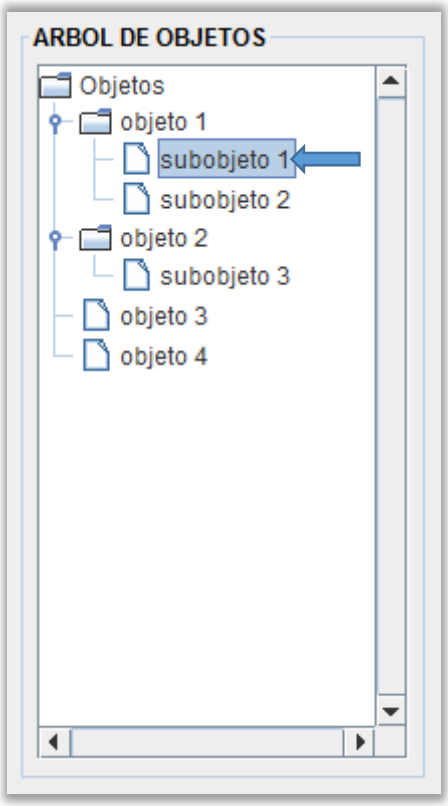

4. Una vez editados los datos del subobjetos el sistema brinda dos opciones: guardar los cambios o cancelar, si se desea guardar los cambios realizados se da clic en Guardar, caso contrario Cancelar.



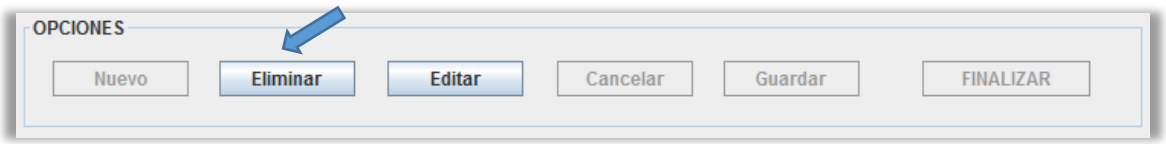

### 6.7. Eliminación de un SubObjetos.

Para eliminar un sub-objeto se realiza los siguientes pasos:

1. Primeramente se procede a seleccionar el subobjeto al cual se desea eliminar en la estructura del árbol de objetos, de la siguiente forma:




2. Una vez seleccionado el sub-objeto se habilitara en el panel de opciones el botón “Editar” y “Eliminar”, con los cuales se puede editar o eliminar el subobjeto seleccionado. Se selecciona la opción “Eliminar” en el panel de opciones de subobjetos.



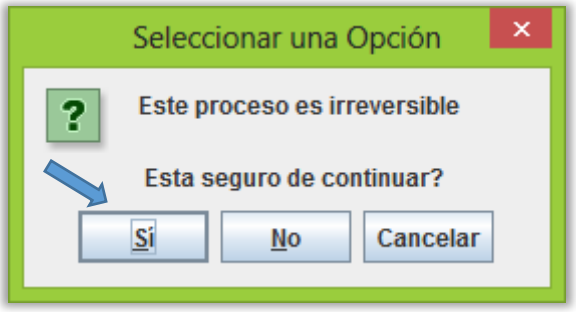

**3.** Inmediatamente se eliminara el subobjeto seleccionado de la aplicación y desaparecerá del “Árbol SubObjetos”.

## 7. GENERAR PROYECTO ANDROID.

Una vez que se ha ingresado los datos de la aplicación, objetos y subobjetos; el sistema de Administrador AR validara que exista por lo menos un subobjeto por cada objeto principal y de esta manera habilitara la opción en el menú Archivo “Exportar -- > Proyecto Android” y el botón “FINALIZAR” en el panel de opciones de objetos, caso contrario si existe objetos principales sin subobjetos asignado,s el sistema no permitirá la exportación de la aplicación.

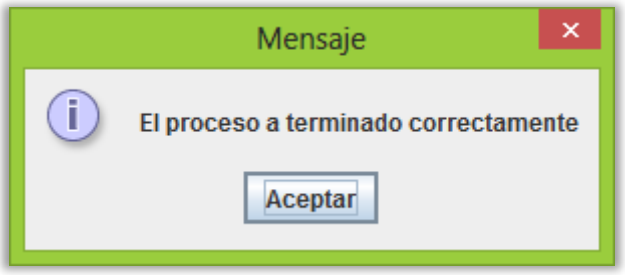
1. Para exportar el proyecto Android simplemente se selecciona la opción “Exportar -- > Proyecto Android” en el menú Archivo o dando clic en el botón “FINALIZAR”


2. Posteriormente el sistema advertirá que este es un proceso irreversible, es decir una vez generado el proyecto no se podrá editar los datos ingresados, y se tendrá que realizar todo el trabajo nuevamente.

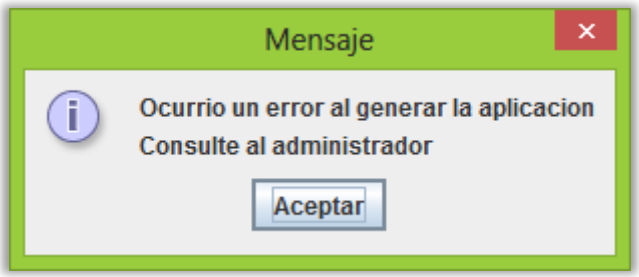



3. Se debe Seleccionar “Si”, si se desea continuar con la creación del proyecto Android, caso contrario, seleccionamos “No” o “Cancelar”. Una vez seleccionado “Si” el sistema de Administrador AR empezara la exportación del proyecto Android.

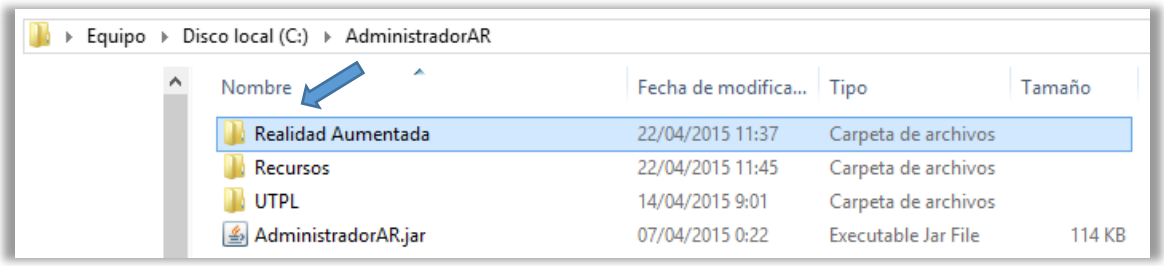
    **a.** Si todo ha salido correctamente mostrara el siguiente mensaje.




**b.** Caso contrario aparecerá el siguiente mensaje de error. Y se debe poner en contacto con el administrador de la aplicación para resolver los problemas.




4. Una vez exportado el proyecto Android exitosamente lo puede encontrar en una carpeta con el nombre de la aplicación en el directorio de instalación de la aplicación (C:\AdministradorAR\nombre_de_aplicacion) el cual contendrá todos los archivos de proyecto Andorid listos para ser cargados en Eclipse y poder exportarlos posteriormente como apk.




**Nota:** Una vez generado la carpeta con el proyecto Android es recomendable copiarla a otra ubicación y de esta manera evitar problemas si se genera dos proyectos con el mismo nombre de aplicación.

## 8. CREACIÓN DEL APK DE LA APLICACIÓN.

Con la carpeta del proyecto creada solo se debe importar dicho proyecto en el entorno de desarrollo de eclipse y desde este se procede a exportar el proyecto a una aplicación de android (instalador APK). Los pasos a seguir para importar la carpeta del proyecto a Eclipse y exportar el mismo como aplicación APK se encuentran debidamente documentados en el Manual del Administrador de RA-Desarrollo-Inteligencia.

## 9. CARACTERISTICAS PARA FUTURAS VERSIONES.

- ✓ Validar si existen Objetos y SubObjetos con el mismo nombre, actualmente no se está validando y puede ocasionar problemas al editar o eliminar sub-objetos.
- ✓ Verificar el uso de caracteres especiales como acentos o ñ o demás signos o caracteres especiales y el uso de mayúsculas en los archivos de audio.
- ✓ Agregar temas para los colores de la aplicación de RA.
- ✓ Seleccionar la ruta de guardado del proyecto exportado.
- ✓ Crear directamente el apk de la aplicación.