**Universidad Nacional de Jujuy**

**Facultad de Ingeniería**

**año 2014**

**Proyecto Final:**

**PROTOTIPO INTERACTIVO DE REALIDAD AUMENTADA PARA SUPERAR LAS BARRERAS DE LA COMUNICACIÓN EN PERSONAS SORDAS E HIPOACÚSICAS.**



Autores:

**Carrizo, Fernando Martin**

**Vale, Walter Jesús**

**Dirigidos por:**

***TUTOR: Ing. Elizabeth Reinoso***

***ASESOR: Lic. Graciela del Valle Espinosa***

Agradecimientos

A la Universidad Nacional de Jujuy.

A Pablo Chávez, Dueño del Restaurante Charlotte, por su tiempo y dedicación al proyecto.

A los que nos guiaron en cada etapa del desarrollo de este trabajo final, nuestro tutor Ingeniera *Elizabeth Reinoso* y nuestra asesor Licenciada *Graciela del Valle Espinosa*.

A Cruz, Verónica Viviana, por prestarnos un lugar para el desarrollo del proyecto.

A nuestros amigos y compañeros de Ikaro por el tiempo brindado en el diseño de interfaces de Usuario.

A nuestros padres, hermanos y familiares por su apoyo incondicional.

A nuestros amigos y compañeros que nos acompañaron a lo largo de la carrera que hicieron ameno este largo trayecto

**Estructura de la tesis(cambiar)**

La estructura del proyecto cuenta con 8 capítulos en los cuales se describe los documentos necesarios para desarrollar dicho proyecto.

En el Capitulo 1 "Introducción" se realiza una introducción a la realidad aumentada en dispositivos móviles, además se definen los objetivos general y secundarios. Finalmente se describe como se encuentra estructurada la tesis.

El Capitulo 2 "Marco teórico" se presenta las definiciones y abreviaturas que se encuentran en la documentación del proyecto, además de los conceptos teóricos necesarios para el desarrollo del proyecto relacionadas con la realidad aumentada, su arquitectura y las herramientas en la actualidad.

En el Capitulo 3 "Análisis y evaluación de tecnologías de realidad aumentada" se centra en la investigación y el estudio de las distintas tecnologías de realidad aumentada.

En el Capitulo 4 "Metodología de desarrollo" se describe la metodología que se utilizo durante el desarrollo del proyecto.

En el Capitulo 5 "Preparación del entorno de trabajo" se describe la preparación de los programas que se utilizaran en el trabajo y sus requerimientos técnicos.

En el Capitulo 6 "Desarrollo del prototipo de realidad aumentada" contiene la descripción, documentación, código y prueba del "Prototipo Interactivo de Realidad Aumentada" que se planteó realizar.

En el Capitulo 7 "Conclusión" se plasman las conclusiones obtenidas durante el desarrollo del proyecto, además, también se definen las posibles líneas futuras de investigación y de desarrollo.

En el capítulo 8 "Definiciones y abreviaturas", contiene las definiciones de las palabras que se pueden encontrar en el proyecto.

En el Capitulo 9 "Bibliografía" se presenta la lista de las referencias bibliográficas utilizadas como complemento teórico para el desarrollo de este proyecto.

Índice General

**INTRODUCCION**

**Introducción**

Las personas sordas e hipoacúsicas graves están incluidas en una comunidad mayoritaria de oyentes y se enfrentan a barreras comunicativas que les dificultan poder acceder a la información y a la comunicación con el contexto social, situando a los mismos en una encrucijada que les obstaculiza con frecuencia el desarrollo personal y social. [Martínez J. L. A y otros, 2010]. Por ello las personas con capacidades diferentes, disminuida auditivamente presentan dificultades en sus vidas cotidianas, que surge como consecuencia de los problemas para percibir los sonidos de su ambiente. Podemos citar un ejemplo que se presenta en los restaurantes cuando los clientes con capacidades diferentes tratan de interactuar con un mozo que no está preparado para brindar servicios a personas sordas e hipoacúsicas graves. Por las dificultades de comunicación, las mismas requieren dependencia de familiares y amigos que cumplan el rol de intérpretes para poder evitar el problema de comunicación.

En la actualidad se vive en un mundo gobernado por las tecnologías, que facilitan las tareas realizadas diariamente. Una de las tecnologías más consumidas en los últimos años son los dispositivos móviles, como las tablets y los celulares, que gracias a su fácil uso y acceso a internet abren un mundo de oportunidades a las personas en general y en particular a las personas con discapacidad. Estos dispositivos permitieron el paso a la utilización de realidad aumentada que combinan elementos de la realidad con información visual para crear una realidad mixta en tiempo real[Redondo, 2012] . Véase la Figura 1. Esta tecnología utiliza técnicas de reconocimiento de formas y visualización 3D para añadir virtualidad a una imagen real de forma coherente y en función de la localización de la escena real [Mamolar, 2012].



Figura 1.1: La tablet combina elementos de la realidad con información visual para crear una realidad mixta en tiempo real [Paradox, 2013]

Haciendo uso de las nuevas tecnologías en dispositivos móviles y de la realidad aumentada, se propone en este proyecto realizar un prototipo para ayudar en la comunicación de personas sordas e hipoacúsicas con capacidades diferentes en los centros gastronómicos de nuestra localidad sin necesidad de un intermediario y aprovechando el uso de íconos y pictogramas que forman parte de la metodología para personas sordas e hipoacúsicas graves.

**Objetivos**

**General**

Este proyecto tiene como objetivo el análisis, diseño y implementación de un prototipo móvil de realidad aumentada para disminuir la desventajas de la comunicación en personas sordas e hipoacúsicas incluidas en una comunidad mayoritaria de oyentes.

**Secundario**

* Realizar una investigación de las tecnologías en RA móviles que se aplican en la actualidad
* Diseñar la arquitectura de software para el prototipo de realidad aumentada en teléfonos móviles.

**CAPITULO 2-MARCO TEORICO.**

**Lenguaje de Señas:**

Uno de los primeros estudios de los sistemas de signos como medio de comunicación para personas sordas fueron realizadas por Juan de Pablo Bonet ,(1573-1633) pedagogo español que escribió “Reducción de las letras y Arte para enseñar a hablar los Mudos “, en este libro se mostraba un método de comunicación mediante alfabeto de signos para las personas sordas.

A partir del alfabeto publicado por J. Bonet, Charles-Michel de l'Épée (1712-1789) publica un alfabeto en el que se basan los alfabetos de signos que se usan en la actualidad. Dicho alfabeto fue publicado por su sucesor el abate Sicard en la forma de “Diccionario general de Signos”.

En 1817 Thomas Gallaudet abrió una escuela de sordos en Connecticut (Estados Unidos) donde sentó las bases para la ASL, tomando como referencia la lengua de signos francesa y algunos signos utilizados por las tribus indias de Norteamérica. En 1965 William Stokoe, describió una gramática precisa para el lenguaje de signos americano en el libro “A Dictionary of American Sign Language on Linguistic Principles” [López, 2009 ].

La lengua de signos es un lenguaje no oral, no auditivo, sino gestual y visual que construyo un nuevo campo de investigación para la ciencia lingüística, centrada, hasta entonces, en los lenguajes hablados [[Rodríguez González](http://www.cervantesvirtual.com/obras/autor/4364/Rodr%C3%ADguez%20Gonz%C3%A1lez,%20Mar%C3%ADa%20%C3%81ngeles), 2003].

En la actualidad existen estudios de los lenguajes de signos, los más conocidos son el ASL(Lenguaje de signo Americano) de Norte América y el LSE (Lenguaje de signos Español) realizado en España ,además existen trabajos sobre los lenguajes de signos gestuales utilizados en otros países de Europa como Gran Bretaña, Francia, Suecia, Dinamarca y Holanda.

A pesar de estos estudios el lenguaje de signos no es universal y puede variar dependiendo la región en la que nos encontremos. Por ello el sistema deberá presentar los símbolos del alfabeto LSE que son utilizados por la comunidad sorda de Jujuy.

**Que es realidad aumentada?**

De una forma vulgar se puede decir que la Realidad Aumentada es un sistema que potencia las capacidades de nuestros sentidos. Similar al trabajo que realiza una lupa, un microscopio, etc. pero, en este caso ampliando la percepción que el usuario tiene de la realidad real mediante la inclusión de elementos virtuales en la misma.

De una forma más técnica, según Ronald Azuma, la Realidad Aumentada (RA) es un entorno que incluye elementos de Realidad Virtual y elementos del mundo real. Por ejemplo, un usuario de RA puede llevar un dispositivo móvil, a través de las cuáles puede ver el mundo, así como imágenes generadas por computadora que se proyectan encima de ese mundo. Siguiendo esta definición, un sistema de RA es aquel que:

* Mejora el mundo real(agregando nueva información)
* Es interactivo en tiempo real
* Se registra en 3 dimensiones

En definitiva, podemos decir que un sistema de RA necesita un dispositivo que se encargue de recoger información sobre la realidad, un ordenador capaz de crear imágenes sintéticas y procesar la imagen real añadiendo esta información (procesador + software) y un medio que permita proyectar la imagen final (pantalla) [Rolando, 2012 ].

**Historia de la Realidad Aumentada**

A principios de los años 60’s varias personas comenzaron a pensar en algún tipo de sistema que pudiese crear un nuevo mundo diferente al real, un mundo que llamarían “realidad virtual”. En 1962, un director de fotografía, Morton Heilig crea un simulador de moto llamado ‘Sensorama’ con imágenes, sonido, vibración y olfato. Varios años más tarde, en 1966, Ivan Sutherland inventa el HMD, display de cabeza, lo que sugiere una ventana a un mundo virtual.

En el año 1975, Myron Krueger, crea “Videoplace”, un laboratorio de realidad virtual, que diseña un sistema que permite a los usuarios interactuar con objetos virtuales. Jaron Lanier fundador de “VPL Research” popularizó a mediados de los 80’s el término “Realidad Virtual”.

En el año 1992 Steven Feiner, Blair MacIntyre y doree Seligmann diseñan KARMA, un prototipo de un sistema de Realidad Aumentada. En el año 2000, los sistemas informáticos llega el “boom” de la realidad aumentada, y se presenta ARQuake, el primer juego al aire libre con dispositivos móviles de Realidad Aumentada. A finales del 2008 sale a la venta AR WIkitude Guía, una aplicación para viajes y turismo desarrollada para la plataforma Android.

En 2009 se crea el logo oficial de la Realidad Aumentada con el fin de estandarizar la identificación de la tecnología aplicada en cualquier soporte o medio por parte del público general [Rolando, 2012 ].



Figura 2.3 Logo oficial de Realidad Aumentada [Total Inmersion, 2011]

**Antecedentes Actuales de Realidad Aumentada**

Hasta ahora, la mayoría de los proyectos o productos de realidad aumentada que se han dado a conocer en ferias o que han llegado al mercado están enfocados a los videojuegos, al turismo o a la publicidad. Son pocas las empresas que desarrollaron aplicaciones destinadas a ayudar a personas con capacidades diferentes, se puede mencionar:

**“accentac” es una nueva aplicación móvil creada por Pixtorm para mejorar la movilidad en Madrid.** La aplicación accentac, de carácter gratuito, describe y sitúa los servicios necesarios para facilitar la accesibilidad y movilidad de los mismos en rampas, ascensores, baños, accesos, salidas de emergencia, puntos de información. Esta aplicación es muy útil para los ciudadanos con discapacidad visual, auditiva o con dificultades de movilidad que necesiten un apoyo al estar en un nuevo entorno[TICbeat, 2012].

En el ámbito del cine, hay que destacar que en el Instituto de Desarrollo Tecnológico y Promoción de la Innovación de la Universidad Carlos III de Madrid se ha desarrollado la aplicación ‘Whatscine’; un sistema de accesibilidad de bajo coste para cine digital que muestra audio descripciones, subtitulado y traducción al lenguaje de signos de las películas para ayudar a las personas con discapacidad auditiva y visual[Bernat, 2013].

Las compañías estadounidenses, Fundación Vodafone y Fundación Aprocor han desarrollado una serie de aplicaciones de realidad aumentada para ayudar a la integración de personas con discapacidad intelectual en el entorno laboral. El proyecto de realidad aumentada viene a ser un tutorial pensado para discapacitados intelectuales, esto se logra gracias a una cámara, una conexión 3G y un dispositivo como una tablet. Concretamente, esta aplicación ha sido desarrollada para solucionar tres tareas: cómo llegar a un sitio, identificar lugares o personas clave en el trabajo y aprender a utilizar una máquina [*Europa Pres*, 2012].

Android

Android es un sistema operativo móvil basado en Linux, que junto a un conjunto de aplicaciones y herramientas middleware trata de proporcionar una

Plataforma de desarrollo para dispositivos como Smartphone o tablets. Al igual que otros sistemas operativos móviles como IOS, Symbian o Blackberry, fue pensado específicamente para Smartphone pero sin embargo Android presenta una peculiaridad que le ha permitido crecer a un ritmo mucho mayor que sus competidores hasta llegar ser la plataforma más extendida. Esta peculiaridad es que al estar basado en Linux, resulta una plataforma libre, gratuita y multiplataforma.

Otra característica que lo hace popular entre los desarrolladores de aplicaciones es el hecho de estar pensado en un sublenguaje de java que facilita el desarrollo de estas aplicaciones al ser un lenguaje muy extendido y conocido. Esto proporciona herramientas de acceso a componentes del teléfono o dispositivo tales como GPS o la agenda de una manera fácil e intuitiva. Además, Android proporciona una API que nos extiende la ya extensa API de java pero más enfocada a la optimización con el propio sistema operativo [Iñarreasagüés, 2012].

**Herramientas Necesarias:**

Java JDK: El JDK, Java Developmente Kit, se trata de un conjunto de herramientas software para la creación del programa en lenguaje Java. El JDK de Java es necesario para crea aplicaciones en Android e instalación del SDK de Android, por ello necesario para el desarrollo de nuestro proyecto.

Android SDK: Software Development Kit (SDK) o paquete de desarrollo de software propio de Android para la creación de aplicaciones para el sistema operativo Android. Consta de un depurador de código, bibliotecas para la programación en el lenguaje, un simulador de teléfonos, documentación y ejemplos de código de programación. El SDK nos proporciona un gestor de versiones del propio sistema operativo con el que podemos descargar las librerías adicionales de cada versión y nos notifica si alguno de los métodos usados están obsoletos. Además, nos permite acceder y controlar dispositivos Android correctamente conectados a nuestro ordenador.

Target Management System: Se trata de la herramienta online proporcionada por la arquitectura Vuforia para la creación de los patrones de reconocimientos utilizados por el SDK Vuforia para la identificación de los objetos y su posterior tratamiento. Para ello, la realidad aumentada basada en el visionado, a través de cualquier dispositivo con cámara, debe tener lo que se conoce como “DataSet”, con el que compara la imagen captada por la cámara . El “Target Management System” ofrece una herramienta basada en la web para los desarrolladores que utilizan el SDK Vuforia para crear esos “DataSet” a partir de una imagen importada a la herramienta. El objeto que nos crea, se quedará almacenado en la página de Vuforia y podremos descargarnos para su utilización en nuestra aplicación [IñarreaSagüés, 2012].

Para acceder a la herramienta por lo tanto, primero habremos de registrarnos

en la página de Vuforia como desarrolladores del mismo.

**Realidad aumentada vs. Realidad virtual**

La realidad virtual es una tecnología que abarca un amplio espectro de ideas. El termino de Realidad Virtual fue creado por Jaron Lanier, fundador de “VPL Research”, quien la definió como “un entorno generado por una computadora, interactiva, tridimensional en el cuál se introduce a la persona”. Hay tres puntos claves en esta definición:

* Este entorno virtual está generado por una computadora mediante una escena en tres dimensiones, la cual requiere una alta capacidad de gráficos por parte del ordenador para adecuar el nivel de realismo.
* El mundo virtual es interactivo pues el usuario requiere una respuesta en tiempo real desde el sistema para poder interactuar en él de una manera efectiva.
* El usuario está inmerso en el mundo virtual.

Una diferencia importante entre los sistemas de Realidad Virtual y los sistemas de Realidad Aumentada es la inmersión de la persona en el entorno. En los sistemas de Realidad Virtual, la persona se encuentra en un mundo totalmente virtual donde el entorno está bajo control del sistema. Sin embargo, los sistemas de Realidad Aumentada se encargar de “ampliar” la escena del mundo real manteniendo en el usuario una sensación de presencia en el mundo real. Las imágenes virtuales están mezcladas con la visión del mundo real, creando una visión aumentada dando al usuario la ilusión que los objetos de los mundos real y virtual coexisten.

En los sistemas de Realidad Virtual, el usuario está completamente inmerso en un mundo artificial y no hay manera de interactuar con objetos del mundo real. En cambio, en los sistemas de Realidad Aumentada, los usuarios pueden interactuar mezclando el mundo real y virtual de una forma natural. Así, la diferencia entre Realidad Virtual y Realidad Aumentada está en el tratamiento que hacen del mundo real [Rolando, 2012 ].

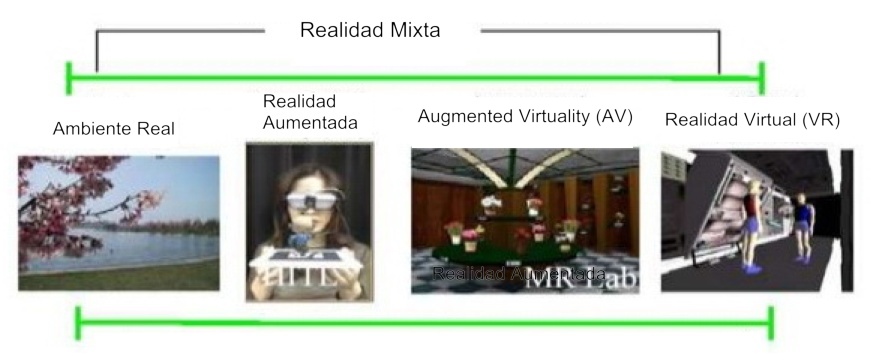


Figura 2.1 Múltiples realidad que existen en la actualidad [Rolando, 2012 ]

**Reconocimiento de marcador**

Los marcadores son hojas de papel o imágenes con símbolos que el software interpreta realizando una respuesta específica para un marcador específico.

Las aplicaciones con marcadores toman fotograma a fotograma de una cámara, bien sea de móvil o webcam, para procesarlo y localizar patrones de imagen conocidos como el mostrado más abajo. Una vez que el sistema localiza uno de los marcadores reconocibles, mezcla la imagen real con su parte virtual mostrando sobre el marcador el objeto que deseemos, tanto en tres dimensiones como en dos dimensiones [Rolando, 2012 ].

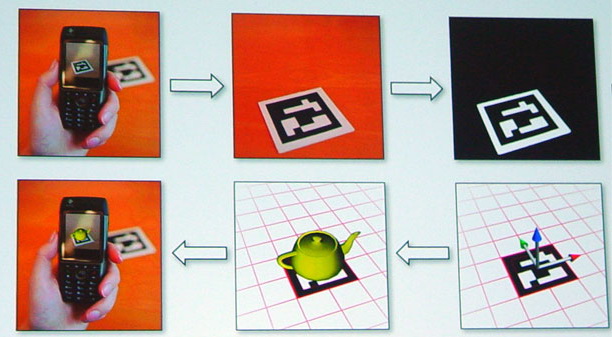
****

Figura 2.2 *Imagen que ilustra el funcionamiento del reconocimiento de marcadores [Sag,2013]*

**Realidad aumentada en dispositivos móviles**

Hoy en día los dispositivos móviles están al alcance de la mayoría de las personas, estos cuentan con las prestaciones necesarias para servir como soporte de estas aplicaciones y sacar el máximo provecho de ellas. La mayor parte de los dispositivos móviles adquiridos por los usuarios disponen de cámara de vídeo de alta resolución y alta capacidad de procesado. Este hecho, y los recientes estudios de mercado, hacen pensar que la integración de la RA en el creciente mercado de los dispositivos móviles forman un campo de investigación de interés [Clemens, 2011].

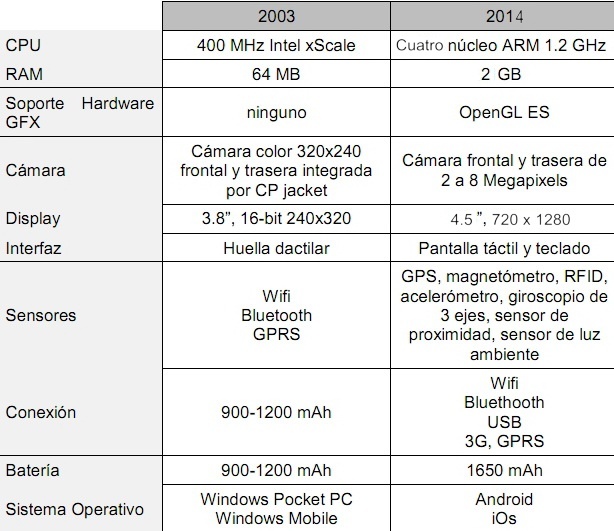


Figura 2.4 comparación del avance tecnológico en los dispositivos móviles [Clemens, 2011].

En la tabla comparativa se observa cómo han mejorado los dispositivos móviles en los aspectos más relevantes para RA: procesadores más rápidos, más memoria, mejores interfaces de entrada, pantallas más grandes y de mayor calidad gráfica, más sensores y mejora de las posibilidades de conexión. La tabla comparativa presentada es una actualización de la presentada por Clements & Schmalstieg.

Con la llegada de los sistemas operativos IOS y Android, el término Smartphone se acuñó para indicar la capacidad de estos dispositivos para la implementación de aplicaciones informáticas complejas. La mejora de interfaces, y en particular la introducción de la pantalla táctil, y el fácil acceso a las aplicaciones para los usuarios, explica su gran éxito comercial. Este fuerte interés comercial también ha traído consigo la mejora de las herramientas de desarrollo de software dedicado a Smartphone.

Gracias a los avances tecnológicos, los dispositivos móviles están cada vez mejor preparados para ejecutar las aplicaciones de ra cada vez más complejas.

**Tecnologías de Realidad aumentada**

En la actualidad existen diferentes tecnologías relacionadas a la implementación de un Prototipo de realidad aumentada para dispositivos móviles. Algunas tecnologías que podemos mencionar son:

* **Unity3D:** Es un motor gráfico 3D para PC y Mac que viene empaquetado como una herramienta para crear juegos, aplicaciones interactivas, visualizaciones y animaciones en 3D.
* **Vuforia:** Es un frameworks para el desarrollo de aplicaciones de realidad aumentada para Android e iOS. Se basa en el reconocimiento de marcas naturales incluyendo objetos 3D[Mamolar, 2012].
* **Metaio:** Es una plataforma de desarrollo de aplicaciones de realidad aumentada para dispositivos Android e iOS. Las aplicaciones se basan en el Reconocimiento de marcas naturales, e integra la gravedad en los módulos de reconocimiento para añadir precisión [Mamolar, 2012].
* **AndAR:** Es una librería creada en el 2010 para dispositivos Android y utiliza una API escrita en java [Mamolar, 2012].
* **NyARToolkit:** Es un SDK de código abierto para el desarrollo de aplicaciones de realidad aumentad basadas en el reconocimiento de marcadores[Mamolar, 2012].

Librería Vuforia

Según el análisis realizado en el capítulo 3, se eligió como tecnología del proyecto a Vuforia. Su funcionamiento se basa en la detección de ciertas imágenes usando la cámara del dispositivo (**trackables**). Las funciones de la librería proporcionan la posición y orientación de dichos trakeables a través de una matriz de 4×4 llamada **matriz de pose**.

**Esta** plataforma de desarrollo de software pone a disposición de los programadores de aplicaciones móviles un motor de reconocimiento de imágenes muy potente, así como un amplio abanico de herramientas diseñado para permitirles dar rienda suelta a su creatividad sin que se vean obligados a preocuparse por las limitaciones de índole técnica.



Figura 2.5 Arquitectura de la aplicación estándar [Mamolar, 2012].

El desarrollador debe disponer previamente de las imágenes que quiere reconocer con su aplicación. Las cuales deben ser subidas a la Web Management System de Vuforia y, o bien se crea una base de datos de imágenes alojada en la nube que se consultará en tiempo de ejecución, o bien se descarga dicha base de datos y se incluye en la aplicación. Durante la ejecución de la aplicación, esta interacciona con el motor de Vuforia para reconocer las imágenes almacenadas en la base de datos. Cuando el motor de Vuforia detecta una imagen en la escena, se puede proyectar el contenido virtual creado por el desarrollador.

OpenGL

Es una librería gráfica escrita originalmente en C que permite la manipulación de gráficos 3D a todos los niveles. Esta librería se concibió para programar en maquinas nativas Silicon Graphics bajo el nombre de GL (Graphics Library). Posteriormente se considero la posibilidad de extenderla a cualquier tipo de plataforma y asegurar así su portabilidad y extensibilidad de uso con lo que se llego al termino Open Graphics Library, es decir, OpenGL.

La librería se ejecuta a la par con nuestro programa independientemente de la capacidad gráfica de la maquina que usamos. Así esta librería puede usarse bajo todo tipo de sistemas operativos e incluso usando una gran variedad de lenguajes de programación. No obstante, su uso más extenso suele ser el lenguaje C o C++. [García Guevara, 2004 ].

Se divide en tres partes funcionales:

* La librería OpenGL, que proporciona todo lo necesario para acceder a las funciones de dibujado de OpenGL.
* La librería GLU (OpenGL Utility Library), una librería de utilidades que proporciona acceso rápido a algunas de las funciones más comunes de OpenGL., a través de la ejecución de comandos de más bajo nivel, pertenecientes a la librería OpenGL propiamente dicha [REDBOOK].
* GLX (OpenGL Extension to the X Window System) proporciona un acceso a OpenGL para poder interactuar con un sistema de ventanas X Window, y está incluido en la propia implementación de OpenGL (su equivalente en Windows es la librería WGL, externa a la implementación de OpenGL).

Además de estas tres librerías, la librería GLUT (OpenGL Utility Toolkit) proporciona una interfaz independiente de plataforma para crear aplicaciones de ventanas totalmente portables [GLUT] [García, 2003 ].

La librería OpenGL viene integrada en Unity3D y es la encargada del renderizado de los objetos 3D, manipular matrices de transformación activa y de otras tareas graficas en bajo nivel.

Arquitectura para Realidad Aumentada

Una aplicación de Realidad Aumentada basada en Vuforia, consta de una serie de componentes muchos de estos son singletons. Un singleton consiste en una clase diseñada para tener sólo una instancia (o un número limitado de ellas), por comodidad a la hora de emplear el lenguaje, se llamará instancia a un singleton.

• Cámara (camera): La instancia de la cámara se encarga de que cada fotograma capturado por la cámara digital se pase de forma eficiente al tracker.

• Convertidor de imágenes (Image Converter): La instancia del conversor de formato de pixel realiza la conversión entre el formato con el que trabaja la cámara a un formato adecuado para el renderizado en OpenGL ES y para el seguimiento (por ejemplo, luminancia).

• Tracker: El tracker contiene los algoritmos de visión computacional para detectar y seguir (detect & track) los objetos en los fotogramas capturados por la cámara. Basado en la imagen tomada por cámara, diferentes algoritmos se ocupan de detectar nuevas imágenes de referencia (Targets) o marcadores (Markers), y evaluar los botones virtuales (Virtual Buttons). Los resultados se almacenan en un objeto de estado que es utilizado por el procesador de vídeo de fondo y al que puede accederse desde el código de la aplicación. El tracker puede cargar múltiples conjuntos de datos (datasets), pero sólo uno puede estar activo a la vez.

• Procesador de vídeo de fondo (Video Background Rendered, singleton): La instancia del procesador de vídeo de fondo procesa la imagen capturada por la cámara que se encuentra almacenada en el objeto de estado.

• Código de la aplicación (Application Code): El desarrollador ha de inicializar todos los componentes anteriores y llevar a cabo tres pasos fundamentales en el código de la aplicación. Por cada fotograma procesado, el objeto de estado se actualiza y se llama al método de procesamiento de la aplicación. El desarrollador debe:

* Consultar el objeto de estado para los Targets y/o Markers nuevos que puedan aparecer en escena o si se actualiza su estado.
* Actualizar la lógica de la aplicación con nuevos datos de entrada.
* Renderizar la capa de Realidad Aumentada.

• Recursos de imágenes de referencia (Target Resources): Los Target Resources se crean mediante el Target Management System, que se encuentra disponible online. El dataset descargado contiene un fichero de configuración XML que permite al desarrollador configurar ciertas características de los trackables y un fichero binario que contiene la base de datos de los trackables. Estos elementos (XML y binario) son compilados por la aplicación a desarrollar en el paquete de instalación de la aplicación y los usa el SDK Vuforia en tiempo de ejecución.



Figura 2.6 Diagrama de flujo de datos en una aplicación

de ejemplo [Mamolar, 2012].

La figura muestra el diagrama de flujo de datos en una aplicación de ejemplo que usa el SDK de Vuforia. La cámara del dispositivo obtiene un frame que se convierte al formato requerido y se pasa al tracker, que puede tener distintas configuraciones. Este proyecto se ha basado en el tracker ImageTargets, es decir, reconoce un solo target o marcador en el frame. Esto es así porque la superficie plana se considera como un marcador-imagen, y suponemos que en la escena solo vamos a tener una única superficie plana dominante, por tanto no tendrá sentido poder reconocer dos marcadores, es decir, dos superficies planas distintas sobre las que proyectar nuestro modelo. El tracker obtiene la lista de marcadores que tiene que buscar de la base de datos alojada en la nube, o integrada en la aplicación, o se han podido crear los marcadores en tiempo de ejecución a partir de frames obtenidos por la cámara del dispositivo móvil. Esta última opción es la que se ha desarrollado en el proyecto. Finalmente, el tracker indica dónde debe renderizarse el modelo, y el nuevo frame se proyecta en la pantalla.

Procesos de diseño 3D

El modelado, consiste en construir objetos a partir de elementos geométricos. Los objetos 3D son sólidos virtuales formados por datos. Los datos tridimensionales se presentan en pantalla mediante líneas, puntos, degradados de color, imágenes o una combinación de todos ellos. Tradicionalmente, se ha modelado siempre en modo línea o en modo puntos. Sin embargo, desde hace unos años la potencia de los ordenadores y el uso de las capacidades 3D en tiempo real, ha permitido la posibilidad de trabajar en modo casi-fotográfico. Esta técnica se conoce normalmente como Open-GL. A pesar del atractivo indudable del modelado en Open-GL, muchas veces el modo línea puede ser la forma más efectiva de afrontar un proyecto. A menudo, se alterna de modo de visualización, a lo largo del proceso de modelado de los objetos.

El texturado, consiste en recubrir nuestros objetos, o mejor dicho la superficie de nuestros objetos, con imágenes que le proporcionen un aspecto lo más realista posible. Las texturas, pueden ser fotográficas o procedurales. Las primeras, son fotografías digitalizadas. Las segundas, son producidas mediante algoritmos matemáticos. También se pueden aplicar a los objetos, superficies rugosas o suaves para imitar esta propiedad de los objetos reales. Aprender a texturar, es básico para obtener un render mínimamente realista. Es necesario un trabajo importante de retoque y redimensionado de imágenes (mediante un programa como Gimp), antes de ponerse manos a la obra con Blender. Un proyecto concienzudo y una preparación meticulosa de las imágenes, nos evitará muchos quebraderos de cabeza. Los programas de 3D, son siempre difíciles, así que una buena planificación del trabajo permite utilizar las técnicas de 3D más sencillas y eficaces[Martinez, 2006].

Blender

Blender es un programa que integra una serie de herramientas para la creación de un amplio rango de contenidos 3D, con los beneficios añadidos de ser multiplataforma y tener un tamaño de unos 5MB.

Destinado a artistas y profesionales de multimedia, Blender puede ser usado para crear visualizaciones 3D, tanto imágenes estáticas como vídeos de alta calidad, mientras que la incorporación de un motor de 3D en tiempo real permite la creación de contenido interactivo que puede ser reproducido independientemente.

Originalmente desarrollado por la compañía 'Not a Number' (NaN), Blender es ahora desarrollado como “Software Libre”, con el código fuente disponible bajo la licencia GNU GPL [Roosendaal, 2004].

Características principales:

* Paquete de creación totalmente integrado, ofreciendo un amplio rango de herramientas esenciales para la creación de contenido 3D, incluyendo modelado, mapeado uv, texturizado, rigging, weighting, animación, simulación de partículas y otros, scripting, renderizado, composición, post-producción y creación de juegos.
* Multiplataforma, con una interfaz unificada para todas las plataformas basada en OpenGL, listo para ser usado en todas las versiones de Windows, Linux, OSX, FreeBSD, Irix y Sun, y otros sistemas operativos.
* Arquitectura 3D de alta calidad permitiendo un rápido y eficiente desarrollo.
* Canales de soporte gratuito vía http://www.blender3d.org .
* Una comunidad mundial de más de 250.000 usuarios.
* Tamaño pequeño de ejecutable para una fácil distribución
* Es un software libre.

Blender se convierte en la herramienta ideal para realizar modelado de objetos 3d necesarios para la aplicación.

**Motores Gráficos**

Un motor de grafico es un conjunto de herramientas que realizan cálculos geométricos y físicos utilizados en aplicaciones y videojuegos. Este conjunto de utilidades representa un simulador ágil en tiempo real que reproduce las características de las aplicaciones y videojuegos.

Puesto que las funcionalidades proporcionadas por un motor son limitada. La manipulación del motor se hace generalmente a través de un lenguaje de script o por la interfaz.

Algunas de los componentes que forman el motor pueden ser la carga, la animación de modelos, la detección de colisiones entres objetos, la física, interfaces gráficas de usuario o incluso herramientas de inteligencia artificial. Mientras que el contenido del una aplicación está compuesto por los modelos, texturas específicas, el comportamiento ante las colisiones de objetos o la forma que tienen los objetos de interactuar con el entorno.

Un motor de gráfico debe ofrece como mínimo las siguientes utilidades:

* Un motor 3D que permite la creación y visualización de un universo 3D.
* Un motor audio que permite la integración de elementos sonoros y música.
* Un motor físico que permita gestionar los comportamientos físicos de los objetos en un universo 3D.
* Herramientas de gestión de red.

Existen diferentes maneras de clasificar los motores de gráficos dependiendo de las características que nos interesan. Una de estas características es el coste, efectivamente existen motores gratuitos o de pago que pueden llegar hasta el medio millón de dólares. Estos últimos son destinados a estudios de desarrollo profesional con gran poder financiero. Estos motores ofrecen un marco de aplicación y en algunos casos un entorno de desarrollo integrado. Este entorno permite crear o importar todos los elementos que se quieren integrar en la aplicación y determinar mecánicas de la aplicación con la ayuda de los scripts.

Los motores gratuitos están destinados a los desarrolladores principiantes o independientes y no suelen tener interfaz global. Para utilizar estos motores hay que descargar un SDK , un kit de desarrollo que permite utilizarlos, pero se debe generar el programa entero a mano, escribiendo código: Crystal Space, Ogre 3D, Irrlicht.

Los motores intermediarios, en esta categoría se incluyen los motores con herramientas de desarrollo más potentes. Estos motores disponen de un entorno de desarrollo integrado (IDE) que permite la creación de programas manipulando los elementos en una interfaz gráfica y generando automáticamente una gran parte del código del juego. Este tipo está dirigido esencialmente a los profesionales independientes: Unity 3D, Torque 3D.

Los motores profesionales están dirigidos a los profesionales con gran presupuesto. Son los motores más completos y los más potentes del mercado, se mantienen al día por sus creadores y disponen de los últimos avances tecnológicos: Id Tech 4, Unreal Engine 3, Cry Engine 3.

Unity 3D

Unity 3D es un motor de creación de videojuegos 3D lanzado oficialmente como tal el 1 de Junio 2005. Este motor permite la creación de juegos y otros contenidos interactivos como diseños arquitectónicos o animaciones 3D en tiempo real.

Muchas personas interesadas por el desarrollo se topan con la dificultad de aprender los lenguajes de programación y los motores que los utilizan. Sin estudios de programación o de animación por ordenador, el aprendizaje de los conceptos, métodos y los principios necesarios para la creación de un videojuego se hace muy difícil.

Unity pone la potencia de su motor al servicio de los utilizadores permitiéndoles obtener un resultado de máxima calidad con un mínimo de esfuerzo.

Unity existe en versión profesional que se puede adquirir previo pago y una versión libre completamente gratuita que se puede descargar en la página Web de Unity. Esta última versión incluye menos funcionalidades pero aun así permite la creación de programas con muy buena calidad.

Unity es una aplicación 3D en tiempo real y multimedia además de ser motor 3D y físico utilizado para la creación de juegos en red, de animación en tiempo real, de contenido interactivo compuesto por audio, video y objetos 3D. Este motor no permite la modelización pero permite crear escenas que soportan iluminación, terrenos, cámaras, texturas[Ouazzani, 2012].

Unity presenta varias ventajas que hacen que sea uno de los motores de videojuego más cotizado del momento. En los siguientes párrafos se van a ir citando todas estas ventajas:

* Permite la importación de numerosos formatos 3D como 3ds Max, Maya, Cinema 4D, Cheetah3D y Softimage, Blender, Modo, ZBrush, FBX o recursos variados tales como texturas Photoshop, PNG, TIFF, audios y videos.
* Es compatible con las API gráficas de Direct3D, OpenGL y Wii. Además de ser compatible con QuickTime y utilizar internamente el formato Ogg Vorbis.
* Este software tiene la particularidad de incluir la herramienta de desarrollo MonoDevelop con la que se pueden crear scripts en JavaScript, C# y un dialecto de Python llamado Boo con los que extender la funcionalidad del editor, utilizando las API que proveé y la cual encontramos documentada junto a tutoriales y recursos en su web oficial.
* La estructura creada por Unity viene definida mediante escenas que representan alguna parte del juego.
* La aplicación permite acceder al Asset Store donde existe multitud de recursos gratuitos y de pago. Incluso se puede extender la herramienta mediante plugins que se obtienen en esta misma tienda.
* Dispone de una interfaz de desarrollo muy bien definida e intuitiva que permite.
* Existe en varias versiones en función de los módulos elegido, la versión más simple destinada a los amateurs es gratuita.

Unity resulta ser un herramienta que permite a sus desarrolladores obtener productos de manera sencilla y rápida.

Técnicas utilizadas en la obtención de requerimientos

Existen varias técnicas para la obtención de requerimiento, sin embargo, se mencionarán algunas de las más importantes. La técnica más apropiada para cada actividad dependerá del proyecto que esté desarrollándose.

Entrevistas y Cuestionarios

Las entrevistas y cuestionarios se emplean para reunir información proveniente de personas o de grupos. Durante la entrevista, el analista conversa con el encuestado; el cuestionario consiste en una serie de preguntas relacionadas con varios aspectos de un sistema.

Por lo común, los encuestados son usuarios de los sistemas existentes o usuarios en potencia del sistema propuesto. En algunos casos, son gerentes o empleados que proporcionan datos para el sistema propuesto o que serán afectados por él.

Las preguntas que deben realizarse en esta técnica, deben ser preguntas de alto nivel y abstractas que pueden realizarse al inicio del proyecto para obtener información sobre aspectos globales del problema del usuario y soluciones potenciales.

Con frecuencia, se utilizan preguntas abiertas para descubrir sentimientos, opiniones y experiencias generales, o para explorar un proceso o problema. Este tipo de preguntas son siempre apropiadas, además que ayudan a entender la perspectiva del afectado y no están influenciadas por el conocimiento de la solución.

El éxito de esta técnica combinada, depende de la habilidad del entrevistador y de su preparación para la misma. Los analistas necesitan ser sensibles las dificultades que algunos entrevistados crean durante la entrevista y saber cómo tratar con problemas potenciales. Asimismo, necesitan considerar no sólo la información que adquieren a través del cuestionario y la entrevista, sino también, su significancia [Pérez Huebe, 2005].

Lluvia de ideas

Este método comenzó en el ámbito de las empresas, aplicándose a temas tan variados como la productividad, la necesidad de encontrar nuevas ideas y soluciones para los productos del mercado, encontrar nuevos métodos que desarrollen el pensamiento creativo a todos los niveles, etc.

Pero pronto se extendió a otros ámbitos, incluyendo el mundo de desarrollo de sistemas; básicamente se busca que los involucrados en un proyecto desarrollen su creatividad, promoviendo la introducción de los principios creativos [Pérez Huebe, 2005].

Principios de la lluvia de ideas

* Aplazar el juicio y no realizar críticas, hasta que no se agoten las ideas, ya que actuaría como un inhibidor. Se debe crear una atmósfera de trabajo en la que nadie se sienta amenazado.
* Cuantas más ideas se sugieren, mejores resultados se conseguirán: “la cantidad produce la calidad”. Las mejores ideas aparecen tarde en el periodo de producción de ideas, será más fácil que se encuentren las soluciones y sé tendrá más variedad sobre la que hay que elegir.
* La producción de ideas en grupos puede ser más efectiva que la individual.
* Tampoco se debe olvidar que durante las sesiones, las ideas de una persona, serán asociadas de manera distinta por cada miembro, y hará que aparezcan otras por contacto.

Las personas que componen el grupo deben estar motivadas para solucionar el problema, y con un ambiente que propicie la participación de todos. Pueden sentirse confiados y con la sensación de que pueden hablar sin que se produzcan críticas. Todas las ideas en principio deben tener el mismo valor, pues cualquiera de ellas puede ser la clave para la solución. Es necesario prestar mucha atención a las frases que pueden coartar la producción de ideas.

Además durante la celebración no deben asistir espectadores. Se deben evitar todos los bloqueos que paralizan la idea: como son los hábitos o ideas preconcebidas, el desánimo o falta de confianza en sí mismo, el temor y la timidez [Pérez Huebe, 2005].

Prototipos

Los prototipos permiten al desarrollador crear un modelo del software que debe ser construido.

Al igual que todos los enfoques al proceso de desarrollo del software, el prototipado comienza con la captura de requerimientos. Desarrolladores y clientes se reúnen y definen los objetivos globales del software, identifican todos los requerimientos que son conocidos, y señalan áreas en las que será necesaria la profundización en las definiciones.

Luego de esto, tiene lugar un "diseño rápido". El diseño rápido se centra en una representación de aquellos aspectos del software que serán visibles al usuario (por ejemplo, entradas y formatos de las salidas). El diseño rápido lleva la construcción de un prototipo.

El prototipo es evaluado por el cliente y el usuario, y utilizado para refinar los requerimientos del software a ser desarrollado. Un proceso de iteración tiene lugar a medida que el prototipo es “puesto a punto" para satisfacer las necesidades del cliente y permitiendo al mismo tiempo una mejor comprensión del problema por parte del desarrollador [Pérez Huebe, 2005].

Existen principalmente dos tipos de prototipos:

Prototipo rápido: El prototipado rápido es un mecanismo para lograr la validación pre-compromiso. Se utiliza para validar requerimientos en una etapa previa al diseño específico. En este sentido, el prototipo puede ser visto como una aceptación tácita de que los requerimientos no son totalmente conocidos o entendidos antes del diseño y la implementación. El prototipo rápido puede ser usado como un medio para explorar nuevos requerimientos y así ayudar a "controlar" su constante evolución.

.

Prototipo evolutivo: Desde una perspectiva diferente, todo el ciclo de vida de un producto puede ser visto como una serie incremental de detallados prototipos acumulativos. Tradicionalmente, el ciclo de vida está dividido en dos fases distintas: desarrollo y mantenimiento.

La experiencia ha demostrado que esta distinción es arbitraria y va en contra de la realidad ya que la mayor parte del costo del software ocurre después de que el producto se ha entregado.

El punto de vista evolutivo del ciclo de vida del software considera a la primera entrega como un prototipo inicial en el campo. Modificaciones y mejoras subsecuentes resultan en nuevas entregas de prototipos más maduros.

Este proceso continúa hasta que se haya desarrollado el producto final. La adopción de esta óptica elimina la distinción arbitraria entre desarrollo y mantenimiento, resultando en un importante cambio de mentalidad que afecta las estrategias para la estimación de costos, enfoques de desarrollo y adquisición de productos [Pérez Huebe, 2005].

Casos de Uso

Los casos de uso son una técnica para la especificación de requerimientos funcionales. Un caso de uso es la descripción de una secuencia de interacciones entre el sistema y uno o más actores en la que se considera al sistema como una caja negra.

Los casos de uso son una técnica para especificar el comportamiento de un sistema: “Un caso de uso es una secuencia de interacciones entre un sistema y actores que usan alguno de sus servicios”.

Los actores son personas u otros sistemas que interactúan con el sistema cuyos requerimientos se están describiendo. Un actor puede participar en varios casos de uso y un caso de uso puede estar relacionado con varios actores.

Los casos de uso presentan ciertas ventajas sobre la descripción meramente textual de los requerimientos funcionales, ya que facilitan la elicitación de requerimientos y son fácilmente comprensibles por los clientes y usuarios. Además, pueden servir de base a las pruebas del sistema y a la documentación para los usuarios.

**Capitulo 3: Analisis y evaluacion de tecnologias para el desarrollo del prototipo.**

**Estudio de tecnologías de realidad aumentada**

Se llevó a cabo un análisis y evaluación de las librerías existentes para el desarrollo de aplicaciones de Realidad Aumentada para dispositivos Android. Se identificó las ventajas e inconvenientes entre cada una de las tecnologías involucradas y se optó por aquella que mejor se ajuste al desarrollo del sistema. En el estudio se consideraron cuatro librerías de código abierto:

**Vuforia:** Es un SDK desarrollado por Qualcomm, una empresa productora de chipsets para tecnología móvil. Esta librería fue publicada a finales de 2010 y está escrita en java y C++. Está disponible para Android e iOS y se basa en el reconocimiento de imágenes basado en características especiales, por lo que también soporta marcadores naturales (targets) o Realidad Aumentada sin marcadores. Además dispone de un plugin para interactuar con Unity3D y ofrece la posibilidad de crear botones virtuales para ampliar las vías de interacción con el usuario. [Mamolar, 2012].

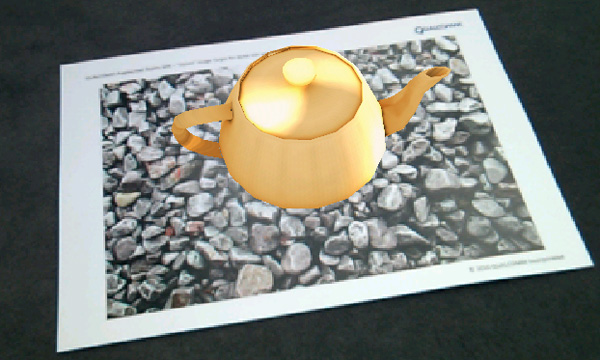


Figura 3.1: Reconocimiento de marcadores naturales o Target. [Qualcomm, 2010]

**Metaio Mobile SDK**: Es una librería escrita en Java, implementada por Metaio y liberada a finales de 2011. Hasta este año todas sus herramientas de desarrollo eran de pago, pero han liberado su SDK para móviles debido al éxito que estaba teniendo. La versión completa, de pago, incluye reconocimiento de caras y de QR (Quick Response Code). Aunque la versión gratuita no incluye estas funcionalidades y que las aplicaciones desarrolladas deben incluir una marca de agua de la empresa, este SDK incluye un potente reconocedor de marcadores naturales, es decir, se pueden desarrollar aplicaciones “markerless”. Por otro lado esta librería añade gravedad a los objetos aumentados, mejorando enormemente la experiencia del usuario. [Mamolar, 2012].



Figura 3.2: Reconocimiento del Patrón QR [Metaio, 2011].

AndAR: Es un proyecto que permite crear aplicaciones de Realidad Aumentada para la plataforma Android. Todo el proyecto se distribuye bajo la licencia GNU,  lo que significa que puede utilizarse libremente. Esta librería fue creada en 2010 por Tobias Domhan y fue escrita en java, también está basado en el proyecto ARtoolKit.

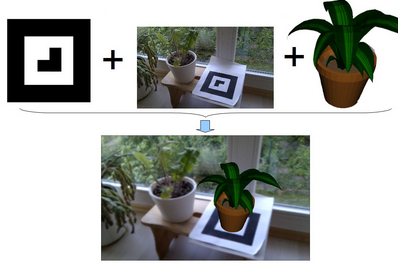


Figura 3.3: Uso de marcadores básicos [AndAR, 2010]

Esta librería funciona con marcadores básicos y maneja una plantilla de texto para gestionar los marcadores. Esta plantilla puede generarse mediante el software mk\_patt de ARToolKit, que sirve para convertir una imagen patrón en una plantilla que pueda ser reconocida por la librería. En la documentación de ARToolKit se puede encontrar la plantilla para crear nuestro propio marcador, puesto que éste debe tener unas proporciones exactas para poder ser reconocido por la librería. [Mamolar, 2012].

**NyARToolkit:** Es una librería también basada en ARToolKit creada por Ryo Lizuka en 2008. En este caso no fue desarrollada específicamente para desarrollo de aplicaciones Android, sino que el proyecto NyARToolkit es compatible con plataformas Java, C#, ActionScript 3, Silverlight 4, C++ y Processing, aunque no todos al mismo nivel en términos de estabilidad. NyARToolkit para Android ha sido desarrollada por un grupo de usuarios de Android en Japón y existe muy poca documentación online, y la que hay está escrita en japonés. Por eso, para entender esta librería hay que sumergirse directamente en el código fuente de Java para Android. Esta librería se encuentra aún en fase de desarrollo, pero no tiene una comunidad grande de desarrolladores detrás, por lo que los avances son muy lentos. En la última versión de NyARToolkit se puede importar formatos como el MD2 y OBJ que son compatibles con Blender. Esta librería utiliza un motor de renderizado externo llamado MIN3D, en lugar de cargar los objetos directamente. [Mamolar, 2012].

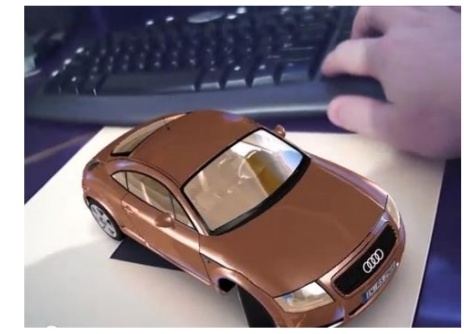


Figura 3.4: Ejemplo NyARToolkit [Diaz, 2013]

En la Figura 3.5 se presenta un resumen de las características de cada una de los SDK:

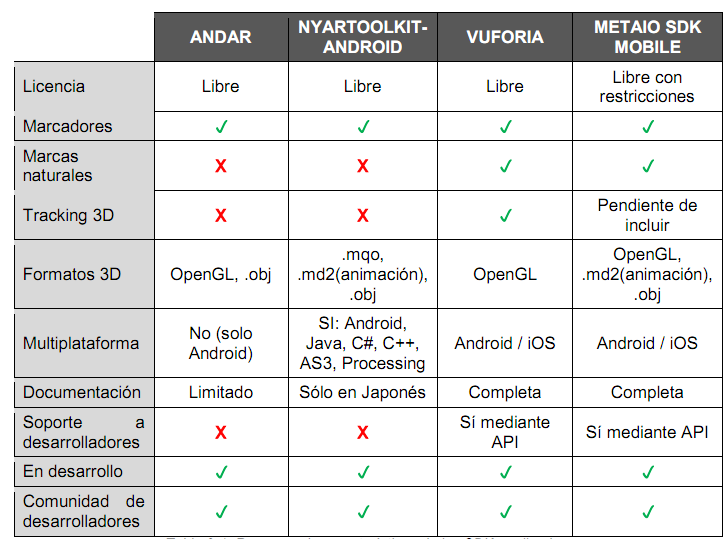


Figura 3.5: Características de las tecnologías de Realidad Aumentada [Mamolar, 2012].

**Análisis de tipos de Marcadores**

Las herramientas ANDAR y NYARTOOLKIT no permite marcadores naturales, por lo que las imágenes como fotografía no pueden ser utilizado con esta tecnología. Ambas librerías utilizan únicamente marcadores QR de color blanco y negro, limando la creatividad en cuanto al uso de los marcadores.

Por un lado Vuforia, dispone de un servicio web, en el que, previo registro, se puede crear los marcadores naturales o targets de manera fácil y sencilla. En el caso de Metaio el proceso es similar, aunque en lugar de ser un servicio on-line se puede descargar la versión demo de la herramienta “Metaio Creator”, una herramienta pensada para generar espacios de Realidad Aumentada rápidamente y sin necesidad de tener conocimientos de programación, todo funciona con una interfaz gráfica en la que se añaden marcadores y objetos 3D que la herramienta se encarga de transformar en una aplicación de Realidad Aumentada. Las aplicaciones generadas contienen una marca de agua de la empresa.

**Reconocimiento del marcador cuando éste es ocultado parcialmente**

Los marcadores QR que utilizan ANDAR y NYARTOOLKIT, si son parcialmente tapados, el sistema no podrá reconocerlo. Sin embargo los SDK que utilizan marcadores naturales si pueden hacerlo. No obstante, Vuforia tiene un mejor comportamiento en este sentido que Metaio. Como curiosidad decir que mientras Metaio y Vuforia sí que implementan la función ScreenShot de Android, andar y NyArtoolkit no lo hacen. Esto es una desventaja, pues la acción ScreenShot añade funcionalidades a las aplicaciones de Realidad Aumentada.

**Reconocimiento del marcador según la distorsión de la perspectiva**

En este caso se analizó el comportamiento de cada SDK frente a distorsiones de la perspectiva. La distorsión de perspectiva se ha representado mediante ángulos de visión de 90º a 15º.

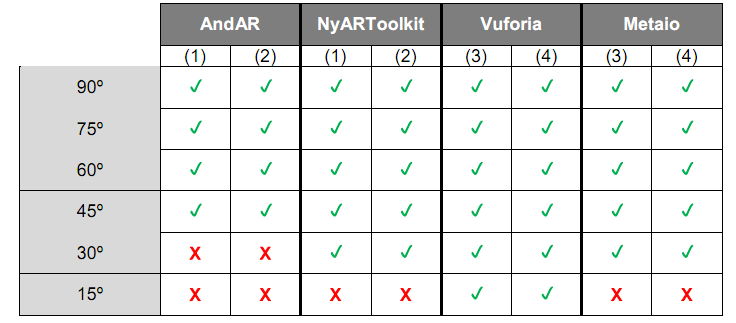


Figura 3.6: Análisis de la distorsión de la perspectiva. [Mamolar, 2012].

Se Observa que el reconocimiento de marcadores naturales es mucho más eficiente cuando se enfrenta a una inclinación pronunciada. Entre los dos SDK el más eficaz es Vuforia, que no deja de reconocer el target a diferentes ángulos de inclinación. [Mamolar, 2012].

**Tasa de frames por segundo en función del número de caras**

En este apartado se analiza el rendimiento de cada SDK en términos de fps (frames por segundo) modificando la escena aumentada. Este estudio está enfocado a la tasa de renderizado en función del número de caras del objeto, recordando que una cara del modelo es cada uno de los triángulos que lo forma. A medida que aumenta la complejidad del objeto a renderizar esta tasa suele bajar. Una de las características básicas de una aplicación de Realidad Aumentada es que sea en tiempo real, para ello la tasa de frames por segundo tiene que estar entre los 15 fps y 30 fps. [Mamolar, 2012].

Los valores recogidos de fps durante las ejecuciones con los diferentes objetos renderizado se muestran en la figura 3.7. En las pruebas, AndAR no ha sido capaz de renderizar el objeto, la aplicación reporta un error y se suspende la ejecución.

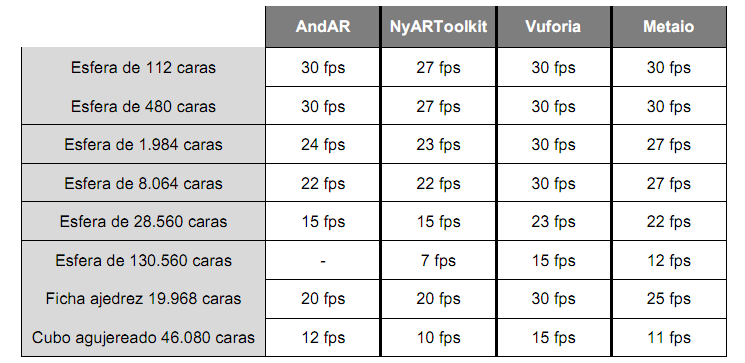


Figura 3.7: Renderizado en términos de frames por segundo según el numero de caras. [Mamolar, 2012].

A la vista de los resultados lo más llamativo es el bajo rendimiento de las librerías NyARToolkit y AndAR frente a las otras dos. Entre ellas NyARToolkit es la que peor resultado ofrece, incluso con el objeto más simple. Ello se debe a que NyARToolkit utiliza un motor de renderizado externo, min3D, lo cual ralentiza esta tarea enormemente. Esto es más llamativo cuando los objetos son más complejos. [Mamolar, 2012].

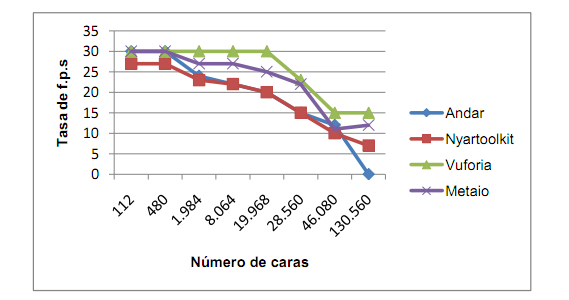


Figura 3.8: Fps en función de números de caras. [Mamolar, 2012].

El SDK que ofrece mejores resultados en todos los casos es Vuforia como lo expresa la figura 3.8. Una explicación es que Vuforia utiliza OpenGL directamente, que sólo contiene funciones de bajo nivel para mostrar y representar gráficos. El hecho de definir el objeto directamente desde primitivas, resulta mucho más eficiente computacionalmente. Además, el módulo de renderizado de Vuforia se encuentra en las funciones nativas en C++, a diferencia de Metaio que está escrito íntegramente en Java, lo que logra mayor eficiencia en cuanto a velocidad de renderizado. [Mamolar, 2012].

Por otro lado, señalar que Qualcomm, la empresa que está detrás de este SDK, es la desarrolladora de la familia de System on Chip (SoC) Snapdragon para la mayoría de dispositivos móviles. Es por ello que el SDK de Qualcomm Vuforia, genera los gráficos de manera óptima para sacar el máximo rendimiento a las prestaciones de Snapdragon. [Mamolar, 2012].

Según lo visto anteriormente, Vuforia ofrece mejores prestaciones que los demás SDK para el desarrollo de aplicaciones interactivas. Metaio al ser el segundo que mejores prestaciones, tiene una licencia libre pero con restricciones, lo que significa limitaciones en cuanto al desarrollo. Vuforia a diferencia de Metaio está completamente liberado. Por otro lado hay que señalar que, para el mismo target y el mismo objeto a renderizar, Metaio se comporta peor en algunas ocasiones produciéndose momentos de jitter y parpadeo. Esto se aprecia especialmente cuando el target se mueve rápidamente. Además hay que señalar que Andar y NyARToolkit utilizan marcadores QR de contraste blanco y negro, limitando la creatividad en cuanto al diseño de marcadores. En cambio Vuforia ofrece marcadores naturales. Se concluye que la mejor herramienta para el desarrollo de aplicaciones móviles con realidad aumentada es la librería Vuforia.

**Elección del entorno de desarrollo**

El prototipo esta divido en dos sistemas, una aplicación para dispositivos móvil y otra para escritorio. A continuación se analizará las tecnologías para el sistema móvil y posteriormente para el sistema de escritorio.

**Desarrollo del Sistema de Realidad Aumentada**

Debido a que Vuforia es la mejor opción para el desarrollo de aplicaciones con realidad aumentada, es necesario un entorno de desarrollo donde se integre ésta librería. Unity 3D y Eclipse ofrecen un ámbito de desarrollo con estas características.

**Unity 3D**

Unity es un motor gráfico 3D para PC y Mac que viene empaquetado como una herramienta para crear aplicaciones interactivas, juegos, visualizaciones y animaciones en 3D en tiempo real. El editor de Unity ofrece un completo editor visual para crear aplicaciones y juegos. El contenido de las aplicaciones es construido desde el editor y se programa usando un lenguaje de scripts. Esto scripts se compiladas usando una versión de JavaScript, C# o Boo, un dialecto de Python. [Collado, 2012].

**Eclipse:**

Es un entorno de desarrollo integrado, de [Código abierto](http://www.ecured.cu/index.php/C%C3%B3digo_abierto) y [Multiplataforma](http://www.ecured.cu/index.php/Multiplataforma). Está diseñada para ser extendida de forma indefinida a través de plug-ins.  Fue concebida desde sus orígenes para convertirse en una plataforma de integración de herramientas de desarrollo. No tiene en mente un lenguaje específico, sino que es un IDE genérico para soporte a múltiples lenguajes de programación. [Gallardo, 2012]

Se realizo una comparación entre los dos entornos para seleccionar la que mejor ajuste al desarrollo del proyecto:

|  |  |
| --- | --- |
| Unity 3D | Eclipse |
| Dispone de una editor grafico para trabajar con los modelos 3D. Usa archivos con extensión obj, fbx, etc. | No dispone de un editor, se debe trabajar con el modelo 3D mediante código, haciendo la tarea más compleja. Maneja los archivos xml para esta tarea. |
| Vuforia dispone de un plugin para trabajar con Unity 3D y es de fácil integración. | Se debe realizar una serie de configuraciones y pasos para dejar el entorno de desarrollo listo. |
| Es multilenguaje para crear aplicaciones Android: C#, javascript o BOO. Se puede utilizar más de dos lenguajes simultáneamente. Utiliza scripts para la programación de aplicaciones. | Para crear aplicaciones Android, utiliza el lenguaje java y código nativo C++. |
| Existe es versión gratuita. | Es libre |
| Es un motor dedica para crear aplicaciones interactivas y juegos. | Es un entorno de desarrollo, pero no está dedica o especializado para aplicaciones interactivas. El desarrollo es mas difícil. |
| Dispone de una tienda Online “Asset Store” que sirve para extender las funcionalidad de Unity 3D ·y descargar modelos 3D. | No dispone de una tienda Online. Se puede extender las funcionalidades mediante plug-ins. |
|  |  |

Tabla 4.1: Comparación entre entornos de desarrollo

El análisis de las características de ambas herramientas, inclina la balanza a favor de Unity 3D. Como [Unity3D](http://unity3d.com/) es una plataforma de contenidos 3D interactivos y [Vuforia](https://www.vuforia.com/) es el sistema de desarrollo (SDK) de realidad aumentada de Qualcomm, junto forman un binomio muy potente que permite crear desde simples visualizadores de realidad aumentada hasta complejos video juegos. Unity 3D y la herramienta de Realidad Aumentada Vuforia se usarán para desarrollar la aplicación para el dispositivo móvil.

**Análisis del Lenguaje de Programación.**

La herramienta Unity 3D elegida para el desarrollo del prototipo puede trabajar con tres tipos de lenguajes:

* C#: Es un lenguaje de programación orientado a objetos para la plataforma .NET.
* Javascript : Es un lenguaje de programación interpretado, lo que significa que no necesita ser compilado. Proviene del Java y se utiliza principalmente para la creación de páginas web.
* BOO, un derivado de Python: Es un lenguaje orientado a objetos, de tipos estáticos para la Common Language Infrastructure, con una sintaxis inspirada en Python y un énfasis en la extensibilidad del lenguaje y su compilador.

La extensión de Vuforia para Unity 3D traer una librería escrita íntegramente en C#, lo que simplifica el análisis del lenguaje de programación al uso de C#. Al ser un lenguaje de programación orientado a objetos, ofrece una gama de posibilidad para el uso de patrones de diseño como el Strategy, Factory, Facade, Dao y Singleton, permitiendo simplificar y mejorar la calidad de la arquitectura del sistema.

**Desarrollo de la Aplicación para Escritorio.**

La aplicación de escritorio tiene la función de gestionar los pedidos realizados por el cliente. Para el desarrollo de aplicaciones de escritorio se opto por Visual Studio 2010, una herramienta que soporta múltiples lenguajes, entre ellos C#. Este lenguaje se eligió según el análisis anterior y se utilizar junto con Visual estudio para crear la aplicación de escritorio.

**Visual Studio 2010:** Es un conjunto completo de herramientas de desarrollo para la generación de aplicaciones web ASP.NET, Servicios Web XML, aplicaciones de escritorio y aplicaciones móviles. Visual Basic, Visual C# y Visual C++ utilizan todos el mismo entorno de desarrollo integrado (IDE), que habilita el uso compartido de herramientas y hace más sencilla la creación de soluciones en varios lenguajes. Asimismo, dichos lenguajes utilizan las funciones de .NET Framework, las cuales ofrecen acceso a tecnologías clave para simplificar el desarrollo de aplicaciones web ASP y Servicios Web XML.

**ELECCION DE LA BASE DE DATOS Y SERVIDOR**

Existen dos bases de datos con las que trabaja la herramienta de desarrollo Unity 3D:

**SQLite:** Es un sistema de gestión de base de datos relacional, contenida en un biblioteca en C. No necesita un motor de base de datos a diferencia de Mysql, PostgreSQL u Oracle, es totalmente independiente que pasa a integrarse a la aplicación. El conjunto de la base de datos como las definiciones, tablas, índices y los propios datos son guardados como un fichero estándar en la maquina local o dispositivo móvil. Realiza operaciones de forma eficientes y es más rápido que Mysql y PostgreSQL. Funciona en muchas plataformas, no requiere un servidor y tampoco configuraciones adicionales para su funcionamiento. SQLite es Software Libre por lo tanto el código fuente es del dominio público y licencia GPL. [Canaza, 2014]

**Mysql:** Se trata de uno de los motores con mejor rendimiento, puesto que posee una gran velocidad a la hora de realizar las operaciones de gestión de la base de datos. Además, teniendo en cuenta sus requerimientos para la elaboración de bases de datos, ofrece un bajo consumo de recursos, pudiéndose ejecutar en equipos no demasiado complejos. Este gestor presenta una fácil instalación y configuración, y soporta gran variedad de sistemas operativos. Mysql es un gestor de base de datos de código abierto y usa la licencia GPL. [Sánchez, 2004].

El desarrollo de sistema comprende dos aplicaciones, una de escritorio y otra móvil. La primera es un programa de escritorio escrito en .Net que gestiona los datos enviados por la aplicación móvil. Es por ello que es necesario un servidor, que capture las peticiones enviadas por el dispositivo móvil. Este servidor debe estar alojado localmente, lo que implica que el dispositivo móvil, debe estar conectado a la red wifi del lugar. En base a lo descripto anteriormente y de las características de los dos gestores, se optó por Mysql, ya que cuenta con una variedad de servidores que incluyen este gestor. SQLite quedo descartado debido a que no cuenta con un motor de base de datos y además no es necesario un almacenamiento en el dispositivo móvil, ya que las peticiones de los pedidos son enviados a una maquina local para su almacenamiento. De la variedad de servidores disponibles que utilizan Mysql, se seleccionó XAMPP, debido a su facilidad de configuración y puesta en marcha.

**XAMPP:** Es un [servidor](http://es.wikipedia.org/wiki/Servidor) independiente de [plataforma](http://es.wikipedia.org/wiki/Plataforma_(inform%C3%A1tica)) de [software libre](http://es.wikipedia.org/wiki/Software_libre), que consiste principalmente en la base de datos Mysql, el servidor web [Apache](http://es.wikipedia.org/wiki/Servidor_HTTP_Apache) y los intérpretes para lenguajes de script: [PHP](http://es.wikipedia.org/wiki/PHP) y [Perl](http://es.wikipedia.org/wiki/Perl). El nombre proviene del acrónimo de **X** (para cualquiera de los diferentes sistemas operativos), **A**pache, Mysql, **P**HP, **P**erl. El programa está liberado bajo la licencia [GNU](http://es.wikipedia.org/wiki/GNU) y actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas. [Dvorski, 2007].

Unity 3D trabaja con el lenguaje PHP, mediante un archivo .php solicita un servicio al servidor, este recibe la petición, reúne la información necesaria consultando las bases de datos y responde con la información solicitada. XAMPP cuenta con todas las herramientas necesarios para el desarrollo del prototipo, un motor de base de datos Mysql y un intérprete para el lenguaje de script PHP.

**HERRAMIENTA DE MODELADO 3D**

Unity puede importar modelos 3D, huesos y animaciones prácticamente de cualquier aplicación 3D. En la Figura 3.9 se da a conocer los distintos tipos de formatos con lo que trabaja Unity 3D.



Figura 3.9: Formatos aceptados por Unity 3d [Serrano, 2011]

Se optó por Blender, ya que un programa informático multiplataforma, dedicado especialmente al modelado, animación y creación de gráficos tridimensionales. El programa es un software libre que actualmente es compatible con todas las versiones de Windows, Mac OS X, Linux, Solaris, FreeBSD e IRIX. [serrano, 2011]. Además brinda todas las características para el desarrollo del prototipo y es compatible con Unity 3d.

Elección de las herramientas de diseño grafico

Es necesaria contar con una herramienta para el diseño de marcadores, botones de interfaz de usuario y símbolos de lenguajes de señas para ser incorporados al prototipo.

Gimp: significa Image Manipulation Program, y es un software de distribución gratuito para trabajos de retoque fotográfico, creación y composición de imágenes. “no dispone de todas las ventajas que posee Photoshop”, aunque siempre se están añadiendo nuevas funcionalidades que facilitan las tareas de diseño. Aún así, no se debe considerar como un reemplazo de Photoshop ya que depende en parte de las tareas requeridas. [Méndez, 2013]

Photoshop: viene con muchas características de edición de gráficos, y es bien recibida por los aficionados y profesionales de edición de fotos. Podemos decir que Photoshop es el más potente software de edición de fotos en el mundo hasta hoy. Por lo general, la gente sólo tendrá que utilizar una pequeña parte de las funciones de edición para lograr grandes efectos fotográficos. Photoshop es una aplicación que dispone de calidad comercial, y como tal debe dar mejor rendimiento por el dinero que pagas. [Méndez, 2013]

El siguiente cuadro muestra las características principales de los editores de imágenes:

|  |  |
| --- | --- |
| Gimp | Photoshop |
| Es un software gratuito | Es pago. Pero existe su versión trial. |
| Bajos requerimientos de almacenamiento | Los requisitos de almacenamiento pueden ser excesivos. |
| Está disponible en windows, mac y linux | Sólo está disponible en windows y mac |
| No tiene un equipo de apoyo central. Pero puede realizar sus consultas en foros. | Apoyo de adobe es caro, pero tiene una mayor disponibilidad. |
| Es de código abierto. Eso significa que usted es libre de ver su código y hacer algunas modificaciones si lo desea. | No es de código abierto |
| Gimp 2.6 es compatible con color de 8 bits | Photoshop cs4 soporta hasta 32 bits. |
| Tiene las funciones básicas. Se pueden encontrar multitud de plugins en la web para cubrir esta carencia. | Tiene mayor conjunto de funciones |
| Su interfaz puede resultar difícil para novatos debido a las potentes funciones para la edición de fotos | Tiene una interfaz mucho más bonita y se ve más profesional. |

Tabla 4.2: Comparación de herramientas de diseño

Se eligió Gimp ya que es una herramienta libre y cuenta con las funciones básicas, suficiente para el desarrollo proyecto.

Patrón arquitectónico MVC

MVC (Model–View–Controller) es un patrón de diseño de arquitectura de software usado principalmente en aplicaciones que manejan gran cantidad de datos y transacciones complejas donde se requiere una mejor separación de conceptos para que el desarrollo esté estructurado de una mejor manera, facilitando la programación en diferentes capas de manera paralela e independiente. MVC sugiere la separación del software en 3 estratos: Modelo, Vista y Controlador, los cuales serán explicados en breve:

* Modelo: Es la representación de la información que maneja la aplicación. El modelo en sí son los datos puros que puestos en contexto del sistema proveen de información al usuario o a la aplicación misma.
* Vista: Es la representación del modelo en forma gráfica disponible para la interacción con el usuario. En el caso de una aplicación Web, la “Vista” es una página HTML con contenido dinámico sobre el cual el usuario puede realizar operaciones.
* Controlador: Es la capa encargada de manejar y responder las solicitudes del usuario, procesando la información necesaria y modificando el Modelo en caso de ser necesario

El ciclo de vida de MVC es normalmente representado por las 3 capas presentadas anteriormente y el cliente (también conocido como usuario). El siguiente diagrama representa el ciclo de vida de manera sencilla:

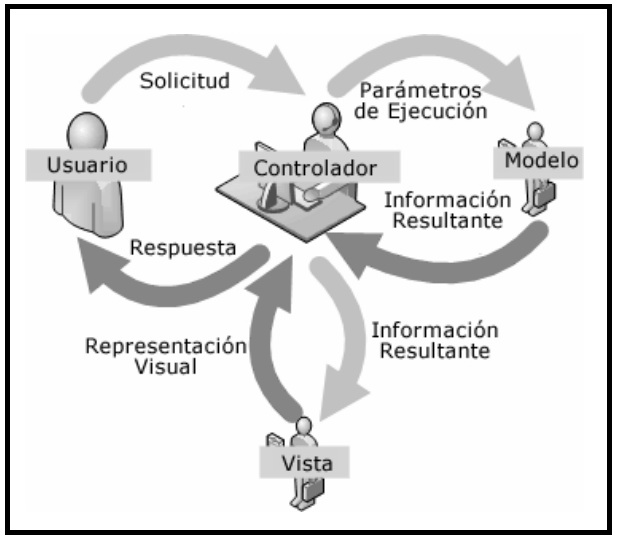


Figura 3.10: Ciclo de vida de MVC

El primer paso en el ciclo de vida empieza cuando el usuario hace una solicitud al controlador con información sobre lo que el usuario desea realizar. Entonces el Controlador decide a quién debe delegar la tarea y es aquí donde el Modelo empieza su trabajo. En esta etapa, el Modelo se encarga de realizar operaciones sobre la información que maneja para cumplir con lo que le solicita el Controlador. Una vez que termina su labor, le regresa al Controlador la información resultante de sus operaciones, el cual a su vez redirige a la Vista. La Vista se encarga de transformar los datos en información visualmente entendible para el usuario. Finalmente, la representación gráfica es transmitida de regreso al Controlador y éste se encarga de transmitírsela al usuario. El ciclo entero puede empezar nuevamente si el usuario así lo requiere.

Ventajas y Desventajas de MVC

Las principales ventajas de hacer uso del patrón MVC son:

* La separación del Modelo de la Vista, es decir, separar los datos de la representación visual de los mismos.
* Es mucho más sencillo agregar múltiples representaciones de los mismos datos o información.
* Facilita agregar nuevos tipos de datos según sea requerido por la aplicación ya que son independientes del funcionamiento de las otras capas.
* Crea independencia de funcionamiento.
* Facilita el mantenimiento en caso de errores.
* Ofrece maneras más sencillas para probar el correcto funcionamiento del sistema.
* Permite el escalamiento de la aplicación en caso de ser requerido.

Las desventajas de seguir el planteamiento de MVC son:

* La separación de conceptos en capas agrega complejidad al sistema.
* La cantidad de archivos a mantener y desarrollar se incrementa considerablemente.
* La curva de aprendizaje del patrón de diseño es más alta que usando otros modelos más sencillos.

Cabe mencionar que la comparación de ventajas y desventajas de MVC puede ser un tema muy subjetivo y se puede prestar como tema de debate, sin embargo, en términos generales la balanza se inclina a favor del MVC en vez de en su contra [Rivera López, 2008].

Patrones de Diseño

Un Patrón de Diseño (design pattern) es una solución repetible a un problema recurrente en el diseño de software. Esta solución no es un diseño terminado que puede traducirse directamente a código, sino más bien una descripción sobre cómo resolver el problema, la cual puede ser utilizada en diversas situaciones. Los patrones de diseño reflejan todo el rediseño y remodificación que los desarrolladores han ido haciendo a medida que intentaban conseguir mayor reutilización y flexibilidad en su software.

Los patrones documentan y explican problemas de diseño, y luego discuten una buena solución a dicho problema. Con el tiempo, los patrones comienzan a incorporarse al conocimiento y experiencia colectiva de la industria del software, lo que demuestra que el origen de los mismos radica en la práctica misma más que en la teoría.

Una definición sobre patrones cita: “cada patrón describe un problema que ocurre una y otra vez en nuestro entorno, así como la solución a ese problema, de tal modo que se pueda aplicar esta solución un millón de veces, sin hacer lo mismo dos veces”. Si bien ésta definición es sobre patrones de ciudades y edificios la idea es aplicable a la industria del software: encontrar una solución a un problema dentro de un contexto.

Un patrón de diseño nomina, abstrae e identifica los aspectos clave de una estructura de diseño común, lo que los hace útiles para crear un diseño orientado a objetos reutilizable. El patrón de diseño identifica las clases e instancias participantes, sus roles y colaboraciones, y la distribución de responsabilidades. Cada patrón de diseño se centra en un problema concreto, describiendo cuándo aplicarlo y si tiene sentido hacerlo teniendo en cuenta otras restricciones de diseño, así como las consecuencias, ventajas e inconvenientes de su uso.

En los últimos años los patrones han ido ganando aceptación, y se fueron extendiendo a otras áreas dentro del desarrollo y mantenimiento de software. Su utilización, si bien todavía le queda mucho camino por recorrer, comienza a tener suficiente madurez.

Los patrones de diseño proveen una forma efectiva para compartir experiencia con la comunidad de programadores de software orientado a objetos[Campo, 2009].

Patron Facade

El patrón Facade, o Fachada, sustituye las interfaces de una serie de clases bajo una sola interfaz por lo que provee un acceso unificado a un subsistema de una aplicación.

Los sistemas se estructuran en subsistemas formados por patrones y clases que los implementan, con dependencias entre ellos. Con la utilización de este patrón se intenta evitar que un cliente, al acceder un sistema (o subsistema) necesite acceder a más de una clase provocando dependencia de cada una de ellas.

Este patrón representa una capa controladora entre los clientes y la capa de negocio, abstrae la complejidad de esta última y le presenta al cliente una interfaz sencilla reduciendo el acoplamiento y dependencia entre ellos. Encapsula la complejidad de las interacciones entre objetos exponiendo sólo las interfaces requeridas y proporcionando un acceso uniforme a los clientes[Mestras,2004 ].

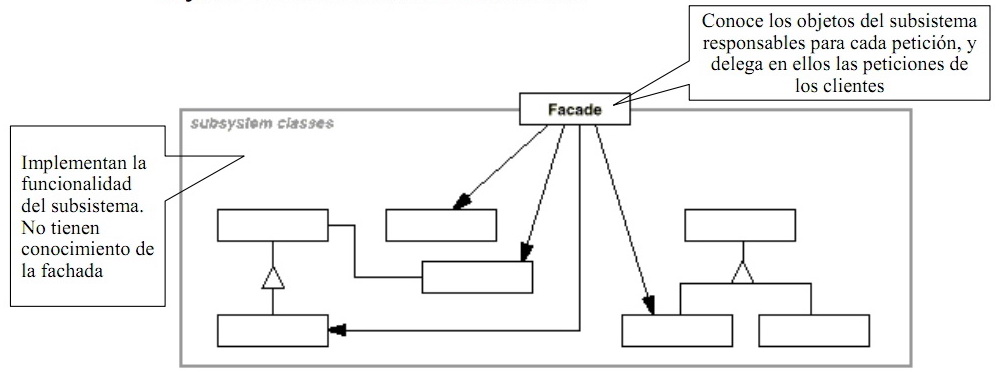


Figura 3.11 Implementación de patrón Facade.

Patron DAO

El patrón DAO (Objeto de Acceso a Datos)abstrae y encapsula todos los accesos a la fuente de datos aislando los objetos de negocio de una implementación particular de una implementación particular de la persistencia. El objeto DAO maneja la conexión con la fuente de datos para obtener y modificar datos implementando los mecanismos de acceso requeridos para trabaja con la fuente de datos.

Para acceder a datos resistentes en bases de datos relacionales, el API JDBC proporciona acceso y manipulación de datos utilizando sentencias SQL. Sin embargo, incluso en estos entornos de bases de datos relacionales, la actual sintaxis y formato de las sentencias SQL pueden variar dependiendo de la base de datos en particular.

Pero las diferencias son mayores cuando se utilizan distintos mecanismos para el almacenamiento de datos. Para evitar dependencias directas de nuestro código con los mecanismos de almacenamiento de datos es especialmente útil la aplicación de este patrón de diseño. Este patrón, al evitar estas dependencias, hace mucho menos complicada y tediosa la migración de la aplicación de un tipo de fuente de datos a otro.

El uso de un objeto de acceso a dato (DAO), proporciona una solución frente a la diversidad de almacenamiento persistente abstrayendo y encapsulando todo acceso a la fuente de datos. Este objeto controla y maneja la conexión con la fuente para almacenar y obtener datos.

Este patrón de acceso a datos proporciona una capa que oculta completamente posibles cambios en la implementación de la fuente de datos, permitiéndole adoptar diferentes esquemas de almacenamiento sin afectar a los clientes o componentes de negocio. Actúa como un adaptador entre un componente y la fuente de datos, permitiendo un acceso transparente a la fuente de datos.

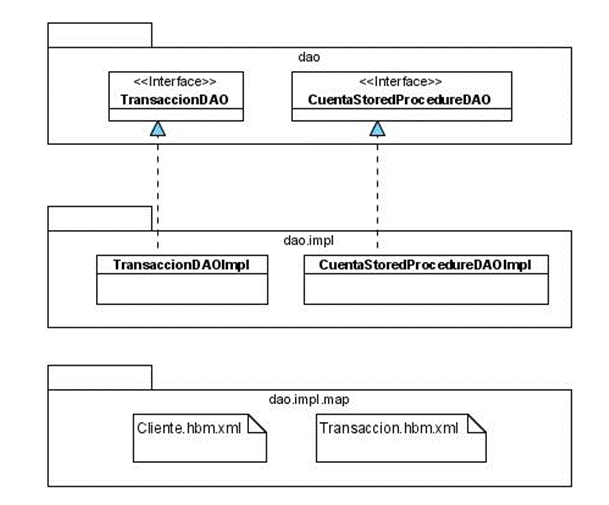


Figura 3.12 implementación de patrón Dao.

Los problemas de esta implementación siguen siendo el mantenimiento de la misma así como su portabilidad. Lo único que podemos decir es que tenemos el código de transacciones encapsulado en las clases DAO.

Lo que parece claro es que debemos separar el código de nuestras clases de negocio de la realización de nuestras sentencias SQL contra la BD. Lo que se necesita es un puente entre nuestra aplicación y la BD, este puente es el framework hibernate con funciones que van desde la ejecución de sentencias SQL a través de JDBC hasta la creación, modificación y eliminación de objetos persistentes.

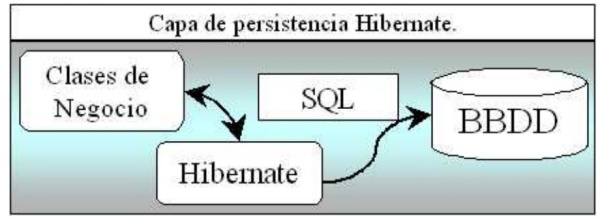


Figura 3.13 Implementación de Hibernate.

Con la creación de la capa de persistencia se consigue que los desarrolladores no necesiten conocer nada acerca del esquema utilizado en la BD. Tan solo conocerán el interface proporcionado por nuestro motor de persistencia. De esta manera conseguimos separar de manera clara y definida, la lógica de negocios de la aplicación con el diseño de la BD.

Esta arquitectura conllevará un proceso de desarrollo más costoso pero una vez se encuentre implementada las ventajas que conlleva merecerán la pena. Es en este punto donde entra en juego Hibernate. Como capa de persistencia desarrollada tan solo tenemos que adaptarla a nuestra arquitectura [Mestras,2004 ].

FrameWorks

Todas las aplicaciones estructuradas en capas comparten un conjunto de requerimientos básicos. A menudo se utiliza una capa de software llamada framework que concentra estos requerimientos y puede ser compartida entre aplicaciones. Sobre la capa del framework tendríamos la capa que consistiría en el código específico de la aplicación, que interacciona con el framework.

El framework se asienta sobre una plataforma proporcionando funcionalidades comunes de aplicación como el manejo de peticiones, invocación de métodos del modelo, y selección y composición de las vistas.

Las clases e interfaces del framework son estructurales. Los desarrolladores de aplicaciones usan, extienden, o implementan las clases e interfaces del framework para desarrollar funciones específicas de la aplicación. Por ejemplo, un framework puede ofrecer una clase abstracta que un desarrollador puede extender para ejecutar la lógica de negocio en respuesta a eventos de la aplicación.

Un framework hace más fácil el uso de tecnologías Web ayudando a los desarrolladores de aplicaciones en la lógica de negocio. El uso de un framework adecuado proporciona desacoplo de la presentación y la lógica, y por tanto separación de roles de desarrolladores y diseñadores de página. Proporcionan un punto central de control además de un conjunto de características personalizables como plantillas, etiquetas para la creación de vistas, localización, control de acceso y log in. Otra de las ventajas es la estabilidad que presentan, además de la facilidad que supone realizar pruebas de unidad y mantenimiento gracias a las interfaces consistentes del framework.

Se recomiendan utilizar frameworks existentes puesto que ahorra esfuerzo, tiempo y costes. Además existen frameworks de gran madurez en el mercado que nos permitirán centrarnos en la lógica de negocio propia de la aplicación [Mestras,2004 ].

Framework: Hibernate

Hibernate es una capa de persistencia objeto/relacional y un generador de sentencias sql. Te permite diseñar objetos persistentes que podrán incluir polimorfismo, relaciones, colecciones, y un gran número de tipos de datos. De una manera muy rápida y optimizada podremos generar BD en cualquiera de los entornos soportados : Oracle, MySql, etc. Y lo más importante de todo, es open source, lo que supone, entre otras cosas, que no tenemos que pagar nada por adquirirlo.

Uno de los posibles procesos de desarrollo consiste en, una vez tengamos el diseño de datos realizado, mapear este a ficheros XML siguiendo la DTD de mapeo de Hibernate. Desde estos podremos generar el código de nuestros objetos persistentes en clases Java y también crear BD independientemente del entorno escogido.

Hibernate se integra en cualquier tipo de aplicación justo por encima del contenedor de datos. Una posible configuración básica de hibernate es la siguiente:

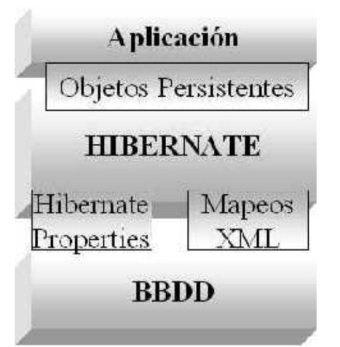


Figura 3.14 Configuración básica de hibernate.

Podemos observar como Hibernate utiliza la BD y la configuración de los datos para proporcionar servicios y objetos persistentes a la aplicación que se encuentre justo por arriba de él[Suárez González, 2013].

**CAPITULO 4- METODOLOGIA DE DESARROLLO.**

La metodología de desarrollo elegida para el presente proyecto es SCRUM. Este es un marco de trabajo para la gestión y [desarrollo de software](http://es.wikipedia.org/wiki/Desarrollo_de_software) basada en un proceso [iterativo e incremental](http://es.wikipedia.org/wiki/Desarrollo_iterativo_y_creciente) utilizado comúnmente en entornos basados en el [desarrollo ágil de software](http://es.wikipedia.org/wiki/Desarrollo_%C3%A1gil_de_software) [Schwaber, 2013].

Entre las principales características se puede destacar [Palacio, 2007]:

* La organización de tareas con prioridades, determinado inicialmente el desarrollo de las que poseen mayor relevancia para el proyecto.
* Scrum permite Inspeccionar rápida y periódicamente las etapas de desarrollo del software en la que se estará trabajando.
* Esta metodología promueve la comunicación entre los miembros del equipo y el cliente.
* Otra característica significativa de Scrum, es que permite gestionar los cambios. Cuando se presenten cambios en los requisitos , los mismos se podrán contemplar de manera sencilla agregándolos al Product Backlog.

Se trabajó siguiendo los principios de Scrum en todas las etapas del desarrollo de software: Formulación del Problemas, Relevamiento y Análisis de Requisitos, Diseño, Codificación y Pruebas.

A continuación se describirá con mayor detalle la metodología seleccionada por el equipo: En SCRUM hay tres aspectos fundamentales a diferenciar, los actores, los eventos o acciones y los artefactos. Los actores son los que ejecutarán las acciones para alcanzar los objetivos del proyecto y con ello se genera los artefactos que servirán para planificar o organizar el desarrollo del proyecto.

**Los actores**

**Product Owner:** representa a los *stakeholders* y conoce los requerimientos de alto nivel del producto marcando las prioridades del mismo.

**Scrum Master:** es la persona que asegura el seguimiento de la metodología guiando las reuniones y ayudando al equipo ante cualquier problema que pueda aparecer. Su responsabilidad es entre otras, la de hacer de paraguas ante las presiones externas.

**Scrum Team:** son las personas responsables de implementar la funcionalidad o funcionalidades elegidas por el Product Owner.

**Usuarios o Cliente:** son los beneficiarios finales del producto, y son quienes viendo los progresos, pueden aportar ideas, sugerencias o necesidades [Schwaber, 2013].

**Los eventos o acciones:**

Daily Scrum Meeting: Es una tarea iterativa que se realiza todos los días que dure el Sprint Backlog con el equipo de desarrollo. Es una reunión operativa, informal y ágil, de un máximo de 30 minutos, en la que se hace 3 preguntas a cada integrante del equipo:

* Que tareas se ha realizado desde la última reunión.
* Sobre qué se va a trabajar en el día actual.
* Identificación de obstáculos o riesgos que impiden o pueden impedir el normal avance del proyecto. El Scrum Master debe eliminar cualquier obstáculo que encuentre.

**Sprint Planning Meeting:** es una reunión que tiene por objetivo planificar el Sprint a partir del Product Backlog. El objetivo de esta reunión es la de mover las tareas del Product Backlog al Sprint Backlog. En esta reunión, suelen participar el Product Owner que es quien prioriza las tareas.

***Sprint Review o Demo*:** se revisa en unas 2 horas como máximo el Sprint finalizado. Al llegar a este punto, se debe tener "algo" que el Cliente o el Usuario pueda ver y tocar. En esta reunión, suelen asistir el Product Owner, el Scrum Master, el Scrum Team y personas que podrían estar involucradas en el proyecto. El Scrum Team es quién muestra los avances realizados en el Sprint.

**Sprint Retrospective**: Al finalizar un Sprint Backlog y el Sprint Review, se inicia el **Sprint Retrospective.** El Product Owner revisará con el equipo los objetivos marcados inicialmente en el Sprint Backlog concluido, se aplicarán los cambios y ajustes si son necesarios, y se marcarán los aspectos positivos (para repetirlos) y los aspectos negativos (para evitar que se repitan) del Sprint. [Schwaber, 2013].

**Los artefactos**:

Product Backlog: Corresponde con todas las tareas, funcionalidades y requerimientos a realizar por el equipo de desarrollo. El Product Owner es la persona que se encarga de marcar las prioridades y de mantener y actualizar la lista de tareas.

Sprint Backlog: Corresponde con una o más tareas que provienen del Product Backlog. El Sprint Backlog tiene una duración de 2 a 4 semanas que al momento de iniciar, este no puede ser alterado o modificado en su transcurso. Se tiene que esperar a que concluya el Spring Backlog para realizar la correspondiente modificación o alteración cuyas tareas formaría para del siguiente Spring Backlog. Cuando se ha finalizado un Sprint Backlog, se debe tener un entregable o algo que se pueda mostrar y que enseñe los avances acometidos en el Sprint.

Incremento:

El Incremento es la suma de todos los elementos del Product Backlog completados durante un Sprint y el valor de los incrementos de todos los Sprints anteriores. Al final de un Sprint, el nuevo Incremento debe estar “Terminado”, lo cual significa que está en condiciones de ser utilizado por el dueño del producto. [Schwaber, 2013].

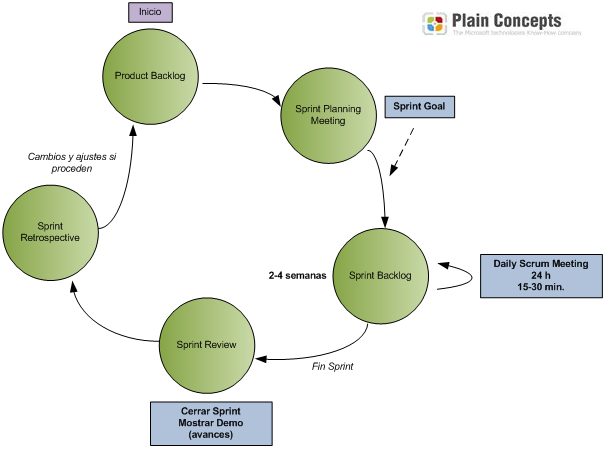
****

Figura 4.1: Flujo de Acciones en un proyecto de Scrum [Schwaber, 2013].

**Planning Poker**

 Es una técnica para calcular una estimación basada en el consenso, en su mayoría utilizada para estimar el esfuerzo o el tamaño relativo de las tareas de [desarrollo de software](http://es.wikipedia.org/wiki/Desarrollo_de_software). Esta técnica se utilizó para estimar las tareas del Product Backlog y determinar el tiempo que empleará cada sprint.

El Planning Poker se basa en una lista de tareas a ser estimado y una baraja de cartas. Un mazo típico contiene tarjetas mostrando la secuencia de Fibonacci incluyendo un cero: 0, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89. Otros mazos utilizan progresiones similares. La razón de utilizar la secuencia de Fibonacci es reflejar la incertidumbre inherente en la estimación. Un mazo que se encuentra en el mercado utiliza la siguiente secuencia: 0, ½, 1, 2, 3, 5, 8, 13, 20, 40, 100, y, opcionalmente, una tarjeta, una con signo de interrogación (?) y otra con signo de infinito (∞), que pueden ser usadas para declarar completa incertidumbre o desconocimiento de la característica. [Piedrasanta, 2011]

Los recursos para trabajar con la técnica Planning Poker son:

* Cartas Planning Poker
* Dos a cuatro participantes
* Un moderador o líder del proyecto.
* Un reloj o algún elemento que determine el tiempo.

**Procedimiento de uso**

En la reunión de la estimación a cada participante se le da un conjunto completo de tarjetas.

La reunión prosigue de la siguiente manera:

* Un moderador, que no jugará, preside la reunión. Puede ser el perfil del Líder de Equipo o Líder de Proyecto.
* El desarrollador con más conocimiento de una determinada característica proporciona una breve introducción sobre la misma. El equipo tiene la oportunidad de hacer preguntas y discutir para aclarar los supuestos y riesgos. Un resumen de la discusión es registrado por el moderador.
* Cada persona coloca una tarjeta boca abajo que representa su estimación. Las unidades utilizadas pueden ser variadas y definidas previamente. Pueden ser días de duración, días ideales o puntos de la historia. Durante el debate, los números no debe ser mencionados en absoluto.
* Todo el mundo muestra sus tarjetas de forma simultánea.
* A las personas con estimaciones altas y bajas se les da un tiempo para ofrecer su justificación para la estimación y la discusión continúa. Como máximo 15 minutos.
* Se repita el proceso de cálculo hasta que se alcance un consenso. El programador que probablemente tenga el entregable tiene una gran parte del voto de consenso, aunque el moderador puede negociar el consenso.
* Se puede utilizar un reloj de arena para asegurar que el debate sea estructurado. Una ronda de Poker finaliza cuando se llego a un acuerdo sobre la estimación o se termino el tiempo del reloj. Se puede dejar una funcionalidad sin consenso y volver después de terminar con las demás tareas. [Piedrasanta, 2011]

**Aplicación de la Metodología de Desarrollo**

**Roles identificados**

En base a los puntos mencionados en la descripción de Scrum se definieron los siguientes roles para el equipo de desarrollo:

* Product Owner: se conto con la colaboración del dueño del restaurante Pablo Chávez, con quien se definió el alcance y la lógica del sistema propuesto.
* Scrum Master: es el encargado de asegurar que el equipo siga las pautas definidas por la metodología y colabora para lograr el éxito del equipo en cada sprint definido. Este rol fue asumido por el tutor del proyecto Elizabeth Reinoso.
* Team: es el equipo de desarrollo encargado de llevar adelante las tareas definidas en cada sprint. Está conformado por Martin Carrizo y Walter Vale.

## Actividades realizadas

Las actividades realizadas durante el desarrollo del prototipo fueron:

* Product Backlog: es una lista priorizada de requisitos y básicamente se basó en el alcance definido para el proyecto.
* Sprint: son las iteraciones de trabajo con duraciones acotadas de tiempo, para el desarrollo de producto se consideró 4 semanas como la mejor opción para la duración de cada ciclo.
* Daily Scrum: es una reunión diaria breve de entre 10 a 15 minutos, donde el equipo da a conocer básicamente los avances, e inconvenientes del sprint en curso. De esta información pueden resultar, de ser necesario, una re planificación de las tareas del sprint que se está desarrollando. La Daily Scrum ha sido considerada por el equipo como fundamental para el éxito de los Sprints desarrollados.
* Retrospective: otro de los componentes clave para el logro de los objetivos, ya que es la revisión que hace el equipo de todo lo realizado en el sprint que acaba de finalizar, se busca determinar los errores y aciertos producidos en pos de una mejora continua del team.

**Product Backlog:**

Las tareas del Product Backlog se definieron en una reunión con el Product Owner, Scrum Master y Scrum Team. El Product Owner es quién dio a conocer los requerimientos de alto nivel y priorizó las tareas a desarrollarse. Todas las dudas del Scrum Team acerca de la ambigüedad de los requerimientos fueron disipadas por Product Owner. La reunión fue precedida y coordinada por el Scrum Master. La Tabla 4.1 fue el resultado de la reunión.

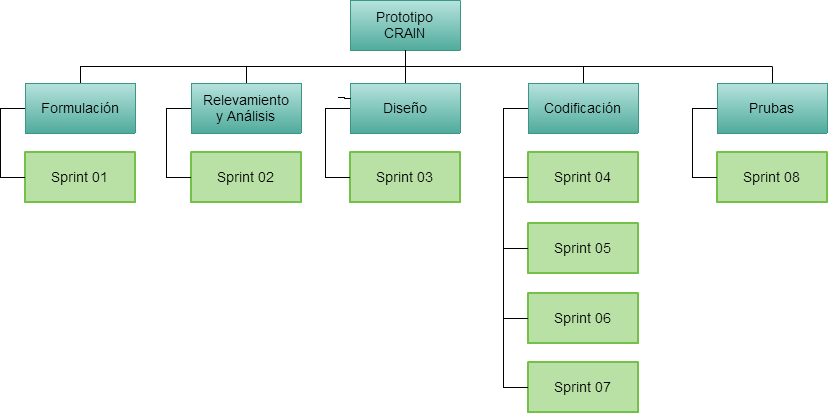
La tabla 4.1 está compuesta por la columna ID para identificar unívocamente las tareas del Product Backlog y la del Spring Backlog. El ESFUERZO ESTIMADO para finalizar una tarea fue calculado utilizando Planning Poker. El CRITERIO DE ACEPTACIÓN es la condición que debe lograrse para que la tarea adopte la categoría de finalizada o terminada.

|  |  |  |  |
| --- | --- | --- | --- |
| **PRODUCT BACKLOG ITEMS** | | | |
| **ID** | **TAREA** | **ESFUERZO ESTIMADO (Hs)** | **CRITERIO DE ACEPTACIÓN** |
| **Etapa de Formulación** | | | |
| 1 | Investigación de Antecedentes | 8 | Realizar una Investigación de Antecedentes de realidad aumentada aplicada a personas con capacidades diferentes en otras ciudades. |
| 2 | Recopilación del material bibliográfico y de investigación. | 16 | Reunir el material necesario para la elaboración del prototipo. |
| 3 | Análisis, estudio y selección de las tecnologías apropiadas . | 15 | Investigar las herramientas, Framework y librerías de realidad aumentada más óptimas para el desarrollo del prototipo. Realizar una comparación de las tecnologías y justificar la elección en un documento. |
| 4 | Investigación de la simbología de lenguaje de señas. | 16 | Se debe investigar la simbología del lenguaje de señas y de profesionales especializados en el tema. |
| 5 | Documentación | 10 | Se debe documentar la Etapa de Formulación en el documento Final. |
| **Etapa Relevamiento y Análisis** | | | |
| 6 | Entrevista | 2 | Entrevistar al dueño del restaurante o Product Owner para recabar información. |
| 7 | Reunión Informal. | 2 | Realizar reuniones para aplicar la técnica de tormentas de ideas para obtener requerimientos. |
| 8 | Análisis de Documentación existente. | 4 | Analizar la documentación que nos provea el dueño del restaurante. |
| 9 | Definición de requerimientos funcionales y no funcionales del prototipo | 8 | Obtener una descripción completa del comportamiento del sistema a través del Product Owner en lenguaje natural. |
| 10 | Especificación de Requerimientos | 10 | Realizar diagrama de casos de usos con su correspondiente detalle. |
| 11 | Diseño preliminar de la interfaz de Usuario (Prototipo estático). | 4 | Se debe realizar una interfaz estática en papel para ser consultado al Product Owner, tanto para el Sistema de Gestión de Pedido y para el Sistema de Atención Aumentado. |
| 12 | Documentación | 24 | Se debe documentar la Etapa de Relevamiento y Análisis en el documento Final. |
| **Etapa Diseño** | | | |
| 13 | Diseño de la arquitectura de software. | 7 | Se debe crear un diseño de alto nivel de la estructura, funcionamiento e interacción entre las partes del software. |
| 14 | Diseño de Diagramas | 16 | Se debe realizar Diagrama de Base de Datos, Diagrama de Secuencias, Diagrama de Clases, Diccionario de Datos, Diagrama de Entidad Relación (DER) |
| 15 | Diseño de las interfaces del sistema. | 20 | Se debe diseñar el icono, botones, splash y carta gourmet para el Sistema de Atención Aumentada. Se debe diseñar los background para el fondo y botones del Sistema de Gestión de Pedido. |
| 16 | Manual de usuario. | 15 | Se dejará listo un manual de usuario para el Sistema de Atención Aumentado y el Sistema de Gestión de Pedido. |
| 17 | Documentación | 16 | Se debe documentar la Etapa de Diseño en el documento Final. |
| **Etapa Codificación** | | | |
| 18 | Login Administrador | 8 | El usuario administrador debe ingresar al Sistema de Gestión de Pedido usando un usuario y contraseña. |
| 19 | Consulta de Pedido de Administrador | 16 | El usuario administrador podrá consulta los pedidos realizados por sus clientes. La consulta misma se hará entre un intervalo de fechas. |
| 20 | Imprimir consulta de Pedido | 20 | El usuario administrador podrá generar un reporte de un pedido realizado. Este reporte debe ser utilizado por el chef para la elaboración de los platos. |
| 21 | Agregar video | 10 | El usuario administrador podrá agregar un nuevo video de las recomendaciones del chef. El alta de los videos debe almacenarse en la base de datos del servidor. |
| 22 | Cargar Video | 10 | El usuario administrador debe seleccionar de una lista de videos, aquel que será accedido por el Sistema de Atención Aumentada. El video elegido debe cambiar su estado de inactivo a activo. |
| 23 | Eliminar video | 8 | El usuario administrador del sistema podrá eliminar un video de la base de datos si lo requiere. |
| 24 | Agregar Diario | 10 | El usuario administrador podrá dar de alta un nuevo URL de Diario Digital. |
| 25 | Eliminar Diario | 8 | El usuario administrador podrá dar de baja un URL de Diario Digital. |
| 26 | Consulta de Pedido por Cliente | 20 | El usuario cliente podrá consultar el pedido realizado y el importe a pagar. |
| 27 | Alta de Pedido | 13 | El usuario cliente puede realizar un pedido utilizando el Sistema de Atención Aumentado. El pedido tiene que ser almacenado en la base de datos del servidor. |
| 28 | Modificar Pedido | 8 | El Sistema de Atención Aumentado debe ofrecer la posibilidad de modificar el pedido. |
| 29 | Modificar el precio del plato | 8 | El usuario administrador del sistema podrá modificar el precio del plato de la carta gourmet. Únicamente el campo precio del formulario debe ser modificado. |
| 30 | Consulta de Diarios Digitales | 16 | El usuario cliente tendrá acceso a los diarios digitales más leídos de Jujuy. Estos Diario debe ser accedido mediante el Sistema de Atención Aumentada |
| 31 | Consultar Videos | 15 | El usuario cliente tendrá acceso a los video de las recomendaciones del chef o de entretenimiento. |
| 32 | Navegación por la carta gourmet | 20 | El usuario cliente podrá navegar por la carta gourmet apreciando los diferentes platos o modelo 3D. Debe incorporarse botones para la navegación por categoría y para el desplazamiento por los ítems de la carta gourmet. |
| 33 | Reproducir Videos de Señas | 24 | El usuario cliente podrá reproducir un video de lenguajes de señas del menú correspondiente. |
| 34 | Configuración de Servidor | 20 | El usuario administrador podrá configurar la dirección IP donde se encuentre alojada el servidor. Esta dirección es ingresada en el Sistema de Atención Aumentado. |
| 35 | Documentación | 16 | Se debe documentar la Etapa de Codificación en el documento Final. |
| **Etapa Prueba** | | | |
| 36 | Elaboración de Plan de Pruebas. | 16 | Se debe elaborar los casos de pruebas, los datos de entradas y las salidas esperadas de ambos sistemas. |
| 37 | Ejecución de Pruebas | 4 | Se ejecutaron las pruebas en ambos sistemas y se compararon con los resultados esperados. Se realizaron pruebas unitarias e integrales de los módulos, con el fin de detectar falencias en los mismos |
| 38 | Depuración de Software | 12 | Se corrigieron defectos e introdujeron mejoras en los módulos |
| 39 | Documentación | 20 | Se debe documentar la Etapa de Codificación en el documento Final. |
|  | Total | 512 |  |

Tabla 4.1: Listado de tareas del Product Backlog

**MAPA DE DESARROLLO DEL PROYECTO**

El desarrollo de las tareas establecidas en el proyecto estará centrado en 5 Etapas, como se muestra a continuación:

****

**Sprint Backlog**

Cada iteración del Sprint Backlog fue definido en una reunión con el Product Owner, el Scrum Team y Scrum Máster. Al finalizar la iteración, se realizó una demo mostrando los resultados al Product Owner. Esta reunión es llamada Sprint Review o Demo. Se definió una venta de tiempo de aproximadamente 1 mes para finalizar cada Sprint dicando 8 horas /semanales por alumno.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sprint** | **Id** | **Tarea** | **Esfuerzo Estimado**  **hs.** | **Asignación** | **Resultado** |
| **Sprint 01**  **13/11/13 al 12/12/13** | 1 | Investigación de Antecedentes | 8 | Walter | Finalizado |
| 2 | Recopilación del material bibliográfico y de investigación. | 16 | Martin | Finalizado |
| 3 | Análisis, estudio y selección de las tecnologías apropiadas . | 15 | Ambos | Finalizado |
| 4 | Investigación de la simbología de lenguaje de señas. | 16 | Ambos | Finalizado |
| 5 | Documentación | 10 | Martin | Finalizado |
| TOTAL | | 65 hs. | | |
| **Sprint 02**  **07/02/14 al 18/03/14** | 6 | Entrevista | 2 | Walter | Finalizado |
| 7 | Reunión Informal. | 2 | Martin | Finalizado |
| 8 | Análisis de Documentación existente. | 4 | Walter | Finalizado |
| 9 | Definición de requerimientos funcionales y no funcionales del prototipo | 8 | Ambos | Finalizado |
| 10 | Especificación de Requerimientos | 10 | Ambos | Finalizado |
| 11 | Diseño preliminar de la interfaz de Usuario (Prototipo estático). | 4 | Martin | Finalizado |
| 12 | Documentación | 24 | Ambos | Finalizado |
| TOTAL | | 54 hs. | | |
| **Sprint 03**  **31/04/14 al 30/05/14** | 13 | Diseño de la arquitectura de software. | 7 | Ambos | Finalizado |
| 14 | Diseño de Diagramas | 16 | Walter | Finalizado |
| 15 | Diseño de las interfaces del sistema. | 20 | Ambos | Finalizado |
| 16 | Manual de usuario. | 15 | Martin | Finalizado |
| 17 | Documentación | 16 | Ambos | Finalizado |
| TOTAL | | 74 hs. | | |
| **Sprint 04**  **9/06/14 al 1/07/14** | 18 | Login Administrador | 8 | Martin | Finalizado |
| 19 | Consulta de Pedido de Administrador | 16 | Walter | Finalizado |
| 20 | Imprimir consulta de Pedido | 20 | Walter | Finalizado |
| 21 | Agregar video | 10 | Martin | Finalizado |
| 22 | Cargar Video | 10 | Martin | Finalizado |
| TOTAL | | 64 hs. | | |
| **Sprint 05**  **2/07/14 al 08/08/14** | 23 | Eliminar video | 8 | Ambos | Finalizado |
| 24 | Agregar Diario | 10 | Ambos | Finalizado |
| 25 | Eliminar Diario | 8 | Martin | Finalizado |
| 26 | Consulta de Pedido por Cliente | 20 | Walter | Finalizado |
| 27 | Alta de Pedido | 13 | Martin | Finalizado |
| 28 | Modificar Pedido | 8 | Walter | Finalizado |
| TOTAL | | 67 hs. | | |
| **Sprint 06**  **11/08/14 al 09/09/14** | 29 | Modificar el precio del plato | 8 | Martin | Finalizado |
| 30 | Consulta de Diarios Digitales | 16 | Walter | Finalizado |
| 31 | Consultar Videos | 15 | Walter | Finalizado |
| 32 | Navegación por la carta gourmet | 20 | Martin | Finalizado |
| TOTAL | | 59 hs. | | |
| **Sprint 07**  **10/09/14 al 11/10/14** | 33 | Reproducir Videos de Señas | 24 | Martin | Finalizado |
| 34 | Configuración de Servidor | 20 | Walter | Finalizado |
| 35 | Documentación | 16 | Ambos | Finalizado |
| TOTAL | | 60 hs. | | |
| **Sprint 08**  **14/10/14 al 13/11/14** | 36 | Elaboración de Plan de Pruebas. | 16 | Martín | Finalizado |
| 37 | Ejecución de Pruebas | 4 | Walter | Finalizado |
| 38 | Depuración de Software | 12 | Walter | Finalizado |
| 39 | Documentación | 20 | Ambos | Finalizado |
| TOTAL | | 52 hs. | | |

**Sprint Retrospective**

Al finalizar un Sprint Backlog y después de realizarse al Demo, se inicia el **Sprint Retrospective.** El Product Owner revisará con el equipo los objetivos marcados inicialmente en el Sprint Backlog concluido, se aplicarán los cambios y ajustes si son necesarios, y se marcarán los aspectos positivos (para repetirlos) y los aspectos negativos (para evitar que se repitan) del Sprint. Además se incluirá aspectos a mejorar por parte del equipo, que serán aplicados en los posteriores sprint.

|  |  |  |  |
| --- | --- | --- | --- |
| Retrospective | | | |
| **SPRINT** | **A favor (+)** | **En contra (-)** | **A mejorar** |
| **Sprint 01** | **\***Se obtuvo buenos resultados en las investigación realizadas, pudiendo definir con certeza las tecnologías a utilizar.  \*Contar con el equipamiento necesario como PCs y celulares con sistema operativo Android.  \* Se pudo definir con precisión la formulación del problema a resolver. | \*No contar con un lugar físico con los servicios adecuados como internet. | \*Buscar en casas de amigos y familiares un lugar adecuado para el desarrollo del proyecto. |
| **Sprint 02** | \*Contar con la predisposición de Pablo Chaves quien será el Product Owner del Sistema.  Contar con documentación existente provista por el Product Owner.  \*Comenzar en etapas temprana la redacción del marco teórico del proyecto final. | \*Falta de comunicación y organización en cuanto a la ejecución de las tareas. | \*Implementar las daily para mejorar la comunicación y organización de las tareas. |
| **Sprint 03** | \*Contar con experiencia en el manejo de herramientas de diseño como Adobe Photoshop para la elaboración de interfaces gráficas.  \* Los miembros del Scrum Team ya cuenta con la experiencia en el uso de Scrum. | \* Se dedico demasiado tiempo en completar las tareas asignadas (superior a las estimadas).  \*No estar de acuerdo en el diseño de las interfaces del sistema. | \*Definir con mayor detalles las tareas a realizar por el equipo.  \*Definir la interfaz a priori para no llegar a la desavenencia en cuanto al diseño. |
| **Sprint 04** | \* Conocimientos en los lenguajes de programación orientado a objetos como java o C#.  \*Contar con una variedad de ejemplos de aplicaciones de realidad aumentada. | \*No contar con un sistema de control de versiones y de almacenamiento de archivos. | \*Se debe implementar SVN para la gestión y versión de archivos para el proyecto |
| **Sprint 05** | \*Mejora de la Comunicación y trabajo en equipo. | \*Falta de documentación del código. | \*Documentar el código al terminar un método o clase. |
| **Sprint 06** | \*Uso de patronos de diseño para mejorar la arquitectura del software. | \*Dificultad para agregar lenguajes de señas al prototipo. | \* Buscar más contenido sobre la comunidad sorda de San Salvador de Jujuy. Incorporación de lenguajes de señas mediante incrustación de video en la carta gourmet. |
| **Spring 07** | \*Cumplir con las tareas asignadas en tiempo y forma según la planificación. | Ninguna | Ninguna |
| **Sprint 08** | \*Disponer de un buen conjunto de pruebas.  \*Contar con Personas con discapacidad deferentes para ejecutar las pruebas.  \* Se encontraron defectos que posteriormente fueron solucionados. | \*Dificultad en cuanto a las correcciones de errores. | \*Hacer revisión de código para la optimización de recursos del sistema. |

En la tabla 4.3 se detalla las Retrospective de los 8 Sprint realizados.