**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].

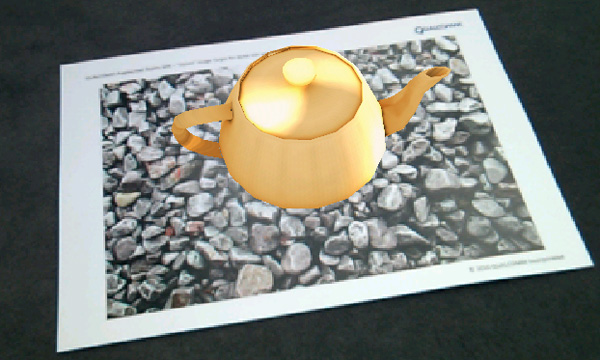


Figura 3.1: Reconocimiento de marcadores naturales o Target. [Qualcomm, 2010]

**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.

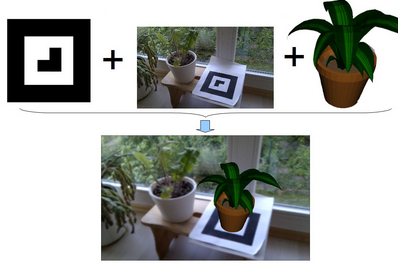


Figura 3.3: Uso de marcadores básicos [AndAR, 2010]

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].

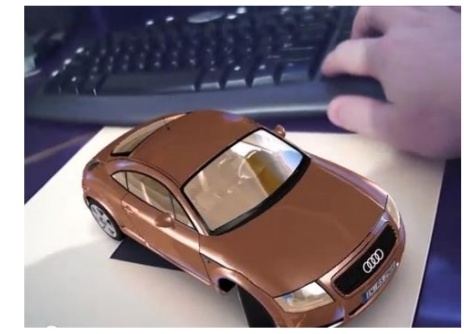


Figura 3.4: Ejemplo NyARToolkit [Diaz, 2013]

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

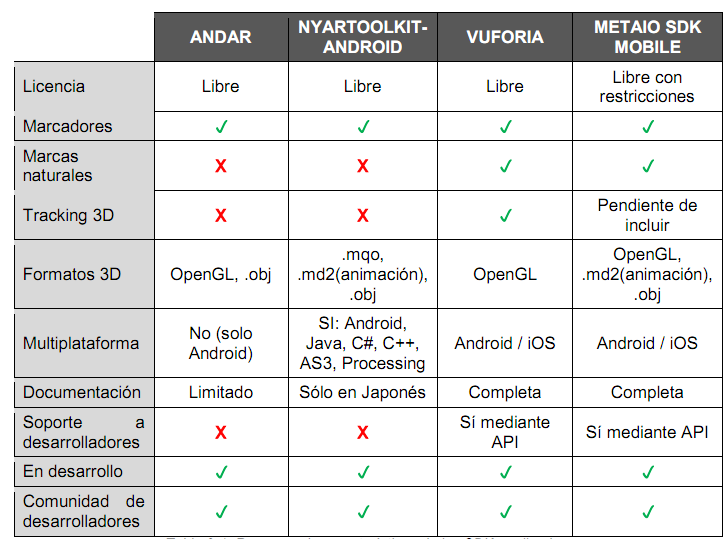


Figura 3.5: Características de las tecnologías de Realidad Aumentada [Mamolar, 2012].

**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.

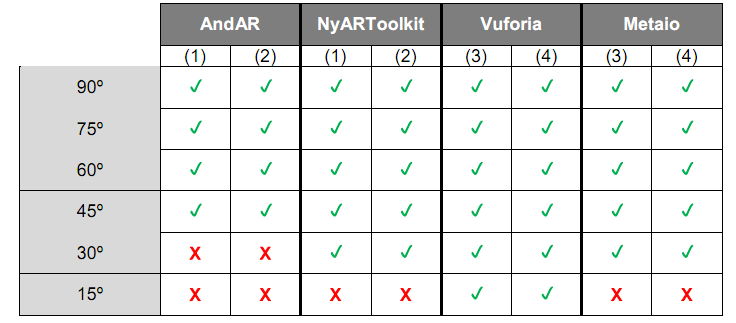


Figura 3.6: Análisis de la distorsión de la perspectiva. [Mamolar, 2012].

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.

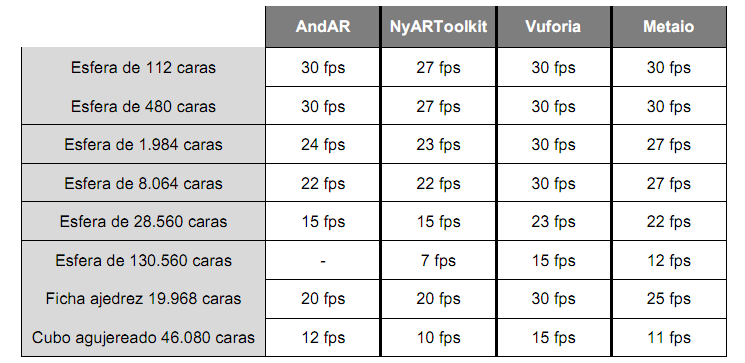


Figura 3.7: Renderizado en términos de frames por segundo según el numero de caras. [Mamolar, 2012].

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].

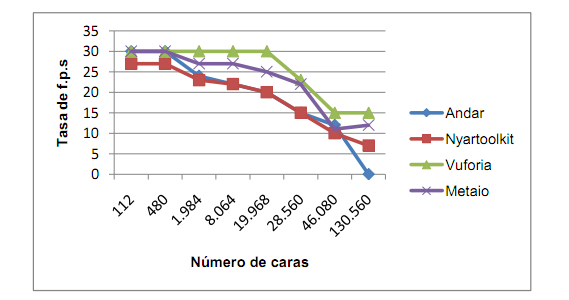


Figura 3.8: Fps en función de números de caras. [Mamolar, 2012].

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:

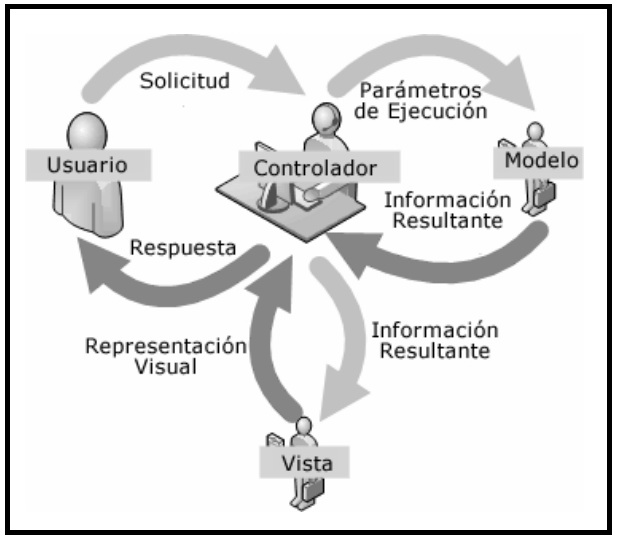


Figura 3.10: Ciclo de vida de MVC

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, a que 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.

En (Gamma et al, 1995) se cita la definición de Christopher Alexander sobre patrones: “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 ].

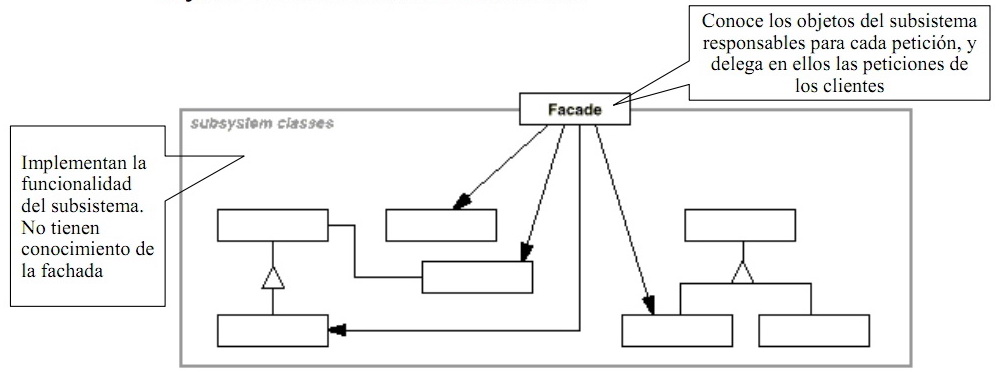


Figura 3.11 Implementación de patrón Facade.

Patron DAO

El patrón DAO 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.

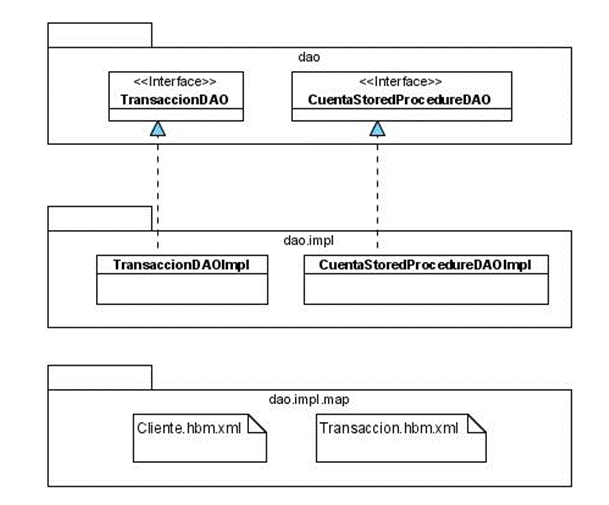


Figura 3.12 implementación de patrón Dao.

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. Por lo tanto Hibernate es el puente entre nuestra aplicación y la BD, sus funciones van desde la ejecución de sentencias SQL a través de JDBC hasta la creación, modificación y eliminación de objetos persistentes.

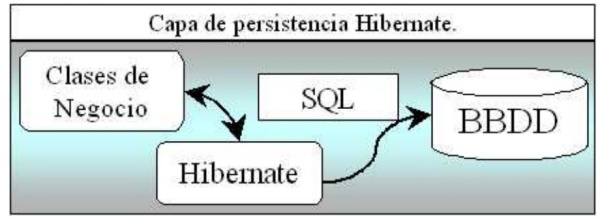


Figura 3.13 Implementación de Hibernate.

Con la creación de la capa de persistencia 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 BBDD 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 BBDD 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:

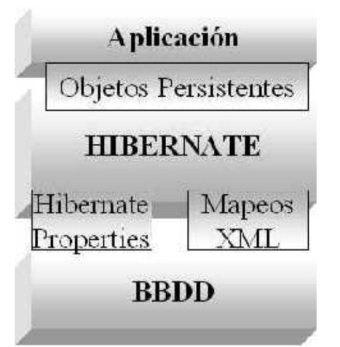


Figura 3.14 Configuración básica de hibernate.

Podemos observar como Hibernate utiliza la BBDD 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].