

**INTEGRANTES DEL EQUIPO:**

|  |  |
| --- | --- |
| ROBERTO ANTONIO AGUILAR VÁSQUEZ | 13700342 |

**ASESOR:**

ING. JOSE

**PROYECTO:**

ISRA-MOPROSOFT

**CARRERA**

INGENIERIA EN SISTEMAS COMPUTACIONALES

**ESPECIALIDAD**

DESARROLLO DE APLICACIONES MÓVILES

**INICIO DEL PROYECTO:**

30-01-2017

**CIERRE DEL PROYECTO:**

26-07-2017

[Capítulo 1: Generalidades del proyecto. 5](#_Toc496963135)

[1.1.- Introducción 5](#_Toc496963136)

[1.2.- Descripción de la empresa. 5](#_Toc496963137)

[1.2.1.- Lugar donde se desarrollará el proyecto. 5](#_Toc496963138)

[1.2.2.- Giro y tipo de capital. 6](#_Toc496963139)

[1.2.3.- Objetivo general de la empresa. 6](#_Toc496963140)

[1.2.4.- Organización administrativa. 7](#_Toc496963141)

[1.2.5.- Descripción de los procesos que desarrolla. 8](#_Toc496963142)

[1.2.6.- Nombre del Departamento donde se realizará el proyecto. 8](#_Toc496963143)

[1.2.7.- Descripción del Departamento. 8](#_Toc496963144)

[1.2.9.- Descripción de las principales actividades del área del proyecto. 8](#_Toc496963145)

[1.3.- Objetivos. 8](#_Toc496963146)

[1.3.1.- Objetivo General. 8](#_Toc496963147)

[1.3.2.- Objetivos específicos. 8](#_Toc496963148)

[1.4.- Justificación. 9](#_Toc496963149)

[Capítulo 2: Marco teórico. 10](#_Toc496963150)

[2.1.- MoPoSoft. 10](#_Toc496963151)

[2.1.1.- Historia. 10](#_Toc496963152)

[2.1.2.- Modelo de procesos. 10](#_Toc496963153)

[2.1.3.- Modelo MoPoSoft. 11](#_Toc496963154)

[2.1.4.- Herramientas de gestión de proyectos. 13](#_Toc496963155)

[2.2.- Tecnologías de desarrollo web. 14](#_Toc496963156)

[2.2.1.-Herramientas. 14](#_Toc496963157)

[2.2.1.1.- Eclipse. 14](#_Toc496963158)

[2.2.1.2.- SQL developer. 14](#_Toc496963159)

[2.2.1.3.- Maven. 14](#_Toc496963160)

[2.2.1.4.- Github. 14](#_Toc496963161)

[2.2.2.- Frontend. 14](#_Toc496963162)

[2.2.2.1.- Definición. 14](#_Toc496963163)

[2.2.2.2.- Css. 14](#_Toc496963164)

[2.2.2.3.- Html 5. 14](#_Toc496963165)

[2.2.2.4.- Bootstrap. 14](#_Toc496963166)

[2.2.2.5.- Angular 2 14](#_Toc496963167)

[2.2.2.6.- Jsp. 14](#_Toc496963168)

[2.2.2.7.- JavaScript. 14](#_Toc496963169)

[2.2.3.- Back end 14](#_Toc496963170)

[2.2.3.1.- ¿Qué es un Backend? 14](#_Toc496963171)

[2.2.3.2.- ¿Cómo funciona un Backend? 15](#_Toc496963172)

[2.2.3.3.- Técnicas usadas en Backend y recomendaciones finales. 17](#_Toc496963173)

[2.2.3.4.- Lenguajes para Back-End 18](#_Toc496963174)

[2.2.3.3.- Servlets. 19](#_Toc496963175)

[2.2.4.- Modelo MVC 20](#_Toc496963176)

[2.2.4.1.- Definición. 20](#_Toc496963177)

[2.2.4.2.- RestFull. 22](#_Toc496963178)

[2.2.4.2.1.- ¿Que es REST ? (Nivel 1 Recursos). 23](#_Toc496963179)

[2.2.4.2.2.- ¿Que es REST? (Nivel 2 HTTP Verbs). 23](#_Toc496963180)

[2.2.4.2.3.- HATEOAS (Nivel 3). 24](#_Toc496963181)

[2.2.4.3.- Spring. 25](#_Toc496963182)

[2.2.4.3.1.- ¿*Qué es Spring?* 25](#_Toc496963183)

[2.2.4.3.2.- INVERSION DE CONTROL (IOC). 26](#_Toc496963184)

[2.2.4.3.3.- INYECCIÓN DE DEPENDENCIA (DI) EN SPRING 26](#_Toc496963185)

[2.2.4.3.4.- Módulos 27](#_Toc496963186)

[2.2.4.4.- Struts. 28](#_Toc496963187)

[2.2.4.4.1.- Que es Struts? 28](#_Toc496963188)

[2.2.4.4.2.- Funcionamiento de Struts 29](#_Toc496963189)

[2.3.- Gestor de base de datos(SQL developer). 31](#_Toc496963190)

[2.3.1.- Definición. 31](#_Toc496963191)

[2.3.2.- Instalación y configuración. 32](#_Toc496963192)

[2.3.3.- Diseño de bases de datos: Conceptos, Operaciones, Objetos y Estructura. 39](#_Toc496963193)

[2.4.- Metodología de desarrollo. 39](#_Toc496963194)

[2.4.1.- Metodología Agil. 39](#_Toc496963195)

[2.4.2.- Definición. 39](#_Toc496963196)

[2.4.3.- Análisis y Diseño. 39](#_Toc496963197)

[2.4.3.1.- Análisis. 39](#_Toc496963198)

[2.4.3.2.- Diseño. 39](#_Toc496963199)

[2.4.3.3.- Construcción e integración. 39](#_Toc496963200)

[2.4.3.4.- Pruebas. 40](#_Toc496963201)

[2.4.3.5.- Cierre. 40](#_Toc496963202)

[Capítulo 3: Desarrollo. 40](#_Toc496963203)

[Capítulo 4: Resultados. 41](#_Toc496963204)

[Capítulo 5: Conclusiones. 41](#_Toc496963205)

[Capítulo 6: Competencias desarrolladas. 41](#_Toc496963206)

[Bibliografía. 41](#_Toc496963207)

[sdkjsd 41](#_Toc496963208)

# Capítulo 1: Generalidades del proyecto.

## 1.1.- Introducción

## 1.2.- Descripción de la empresa.

### 1.2.1.- Lugar donde se desarrollará el proyecto.

* + **Nombre:**

Meltsan Solutions S. A. de C. V.

* + **Giro y tipo de capital:**

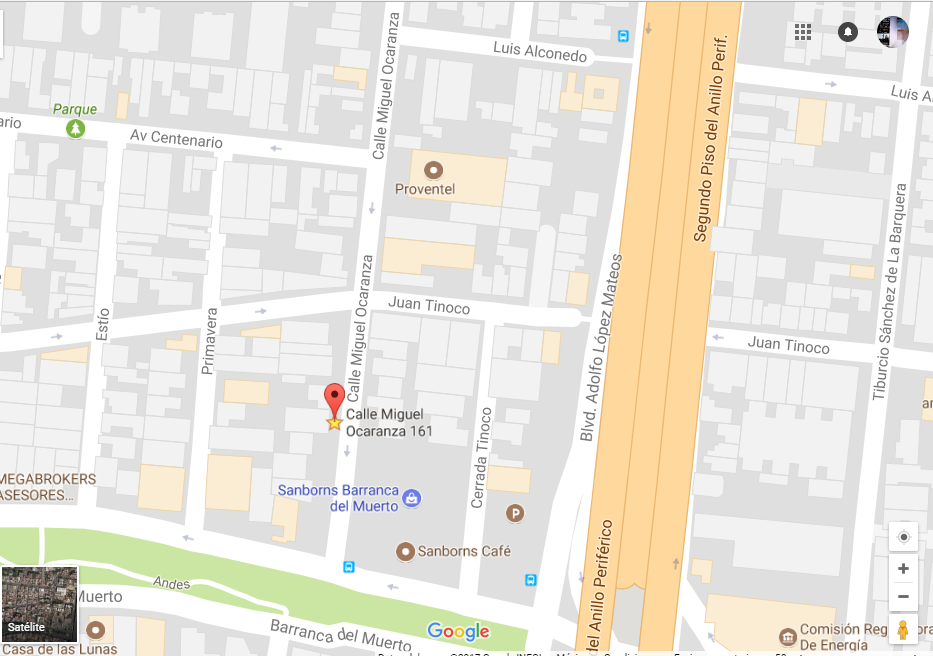
Servicios

* + **Dirección y mapa de localización:**

Calle Miguel Ocaranza 161

Merced Gómez

01600 Ciudad de México, CDMX



**Figura 1: Localización**



**Figura 2: Entrada de la empresa.**

### 1.2.2.- Giro y tipo de capital.

Servicios

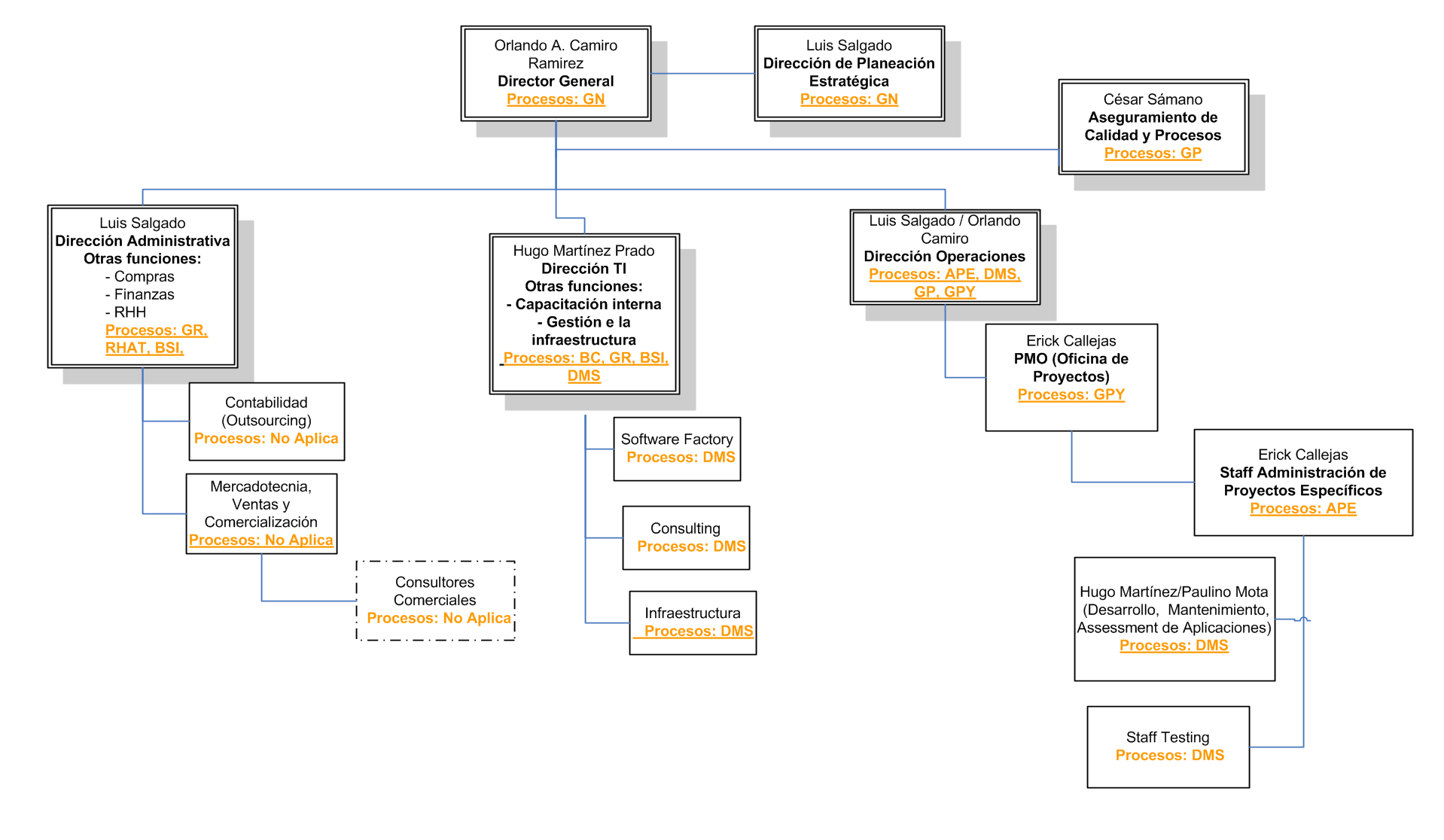
### 1.2.3.- Objetivo general de la empresa.

Misión: Agregar valor a los procesos de negocio de organizaciones del sector, asegurador y de servicios, proporcionando consultoría relacionada con TI y la comercialización de productos de software libre y propietario, haciendo uso de las mejores prácticas del mercado y buscando relaciones sólidas con todo los participantes de nuestra cadena de valor.

Visión: Lograr para el año 2018 ser en México uno de los principales proveedores de tecnologías de Software en el sector asegurador y de servicios, implantando un modelo de negocio propio basado y certificado en Moprosoft nivel 3; incluyendo dentro del portafolio de servicio el aseguramiento de calidad de software y consultoría de procesos de procesos, además de generar la comercialización de software Empaquetado.

Valores: Meltsan valorará y reconocerá públicamente: Lealtad, Honestidad, Responsabilidad, Calidad, Compromiso y persistencia, Integridad, Uso adecuado de los recursos, Flexibilidad para crítica complementaria y trabajo en equipo, Autoconocimiento constante.

### 1.2.4.- Organización administrativa.

****

### 1.2.5.- Descripción de los procesos que desarrolla.

Es una empresa Mexicana de TI con más de 10 años de experiencia en diferentes sectores públicos.

Además de sentirse atraídos por los nuevos retos y en un enfoque de satisfacción total de la cadena de valor, lo cual los hace mejores en lo que hacen al habitar a los clientes en el logro de sus objetivos a través de soluciones innovadoras de TI.

### 1.2.6.- Nombre del Departamento donde se realizará el proyecto.

Proceso DMS (Desarrollo y mantenimiento).

### 1.2.7.- Descripción del Departamento.

En el departamento de Proceso DMS (Desarrollo y mantenimiento). Se encarga de Producir componentes de software en base a la información generada en las etapas de análisis y diseño.

Se encarga de codificar o programar lo que el departamento de Diseño y análisis lo envía una ves desarrollado el proceso de programación pasara posteriormente a las etapas de prueba.

### 1.2.9.- Descripción de las principales actividades del área del proyecto.

En el departamento de Proceso DMS (Desarrollo y mantenimiento). Se encarga de Producir componentes de software en base a la información generada en las etapas de análisis y diseño.

Se encarga de codificar o programar lo que el departamento de Diseño y análisis lo envía una ves desarrollado el proceso de programación pasara posteriormente a las etapas de prueba.

## 1.3.- Objetivos.

### 1.3.1.- Objetivo General.

Crear un sistema (Aplicación web) de gestión bajo la normativa MoProSoft para el manejo de presupuestos de los proyectos de software, ayudando a evitar desvíos en el rubro financiero, llevando un mayor control de su información durante todo el ciclo de vida de sus proyectos a partir del 2017.

### 1.3.2.- Objetivos específicos.

* + - Conocer el funcionamiento general de la norma mexicana MoProSoft en las organizaciones.
    - Conocer el manejo del rubro financiero en los proyectos de las empresas que se dedican a desarrollar software.
    - Identificar las herramientas para el desarrollo del sitio web y el sistema en general.

* + - Elaborar una base de datos con sus respectivas tablas donde se almacenará la información para los reportes de actividades semanal.
    - Diseñar los prototipos del sistema para el manejo de presupuestos para la aproximación del sistema final.
    - Implementación y construcción del sistema para el manejo de REPORTE DE ACTIVIDADES SEMANAL
    - Realización de pruebas unitarias e integrales del sistema para el manejo de presupuestos.

## 1.4.- Justificación.

Según estadísticas realizadas en el 2005 por ESANE consultores, el número aproximado de empresas de la industria mexicana del software podría ser del orden de 1500 empresas revelando que el 91.09% de las organizaciones que desarrollan software son PYMES es decir pequeñas y medianas.

Actualmente la Normalización y Certificación en Electrónica (NYCE) muestra en su página principal un total de 365 empresas de México que obtuvieron un nivel de certificación de MoProSoft hasta el año 2012, lo que indica que esta norma está empezando a tener auge en las empresas, mismas que no tienen una forma de controlar los presupuestos con cada fase de los proyectos (NYCE, 2012) llevándolos a no cumplir con éxito los proyectos por los cuáles se está a cargo.

Hoy en día sabemos que muchas de las empresas de desarrollo de software de México son pequeñas y medianas (PYMES), las cuales cuentan con menos de 100 empleados. La mayoría de las empresas no tiene una cultura de trabajar con una metodología o norma estandarizada que les permita llevar a cabo el proceso de desarrollo.

Asimismo se obtendrá un mayor control de la información como el proceso de presupuesto que permitirá a las empresas desarrolladoras de software a llevar un control del presupuesto de sus proyectos basándome en un módulo de reporte de actividades semanal, evitando desvíos financieros.

# Capítulo 2: Marco teórico.

## 2.1.- MoPoSoft.

### 2.1.1.- Historia.

El tema de desarrollo de software es y ha sido de gran importancia para las pequeñas y medianas empresas (PyMES). En el año 2005, el Instituto de Ingeniería de Software (Software Engineering Institute SEI) llevó a cabo la creación del Consorcio Internacional para la Investigación del Proceso (The International Process Research Consortium IPRC) con el objetivo de definir la ruta de investigación para los años próximos. Su primer taller fue definido como tema principal La mejora de procesos en las PyMES o Small y Middle Enterprise (SME) (Oktaba, 2005).

Con respecto al gobierno de México, su primera acción fue a través de la Secretaría de Economía. En el 2002 inició con el Programa de Desarrollo de la Industria del Software (ProSoft). El objetivo de ProSoft es fortalecer a la industria de software en México, con la característica de que un 90% de las organizaciones de este sector son Pymes [SG, 2004].

Posteriormente para lograr el fortalecimiento con base a la estrategia ProSoft “Alcanzar niveles internacionales en capacidad de procesos”, se desarrolló un modelo propio llamado Modelos de Procesos para la Industria de Software (MoProSoft) y su método de evaluación de Procesos de Software (EvalProsoft) reconocidos en el 2005 como una norma mexicana (Oktaba, 2006).

El origen del estándar MoProSoft es la necesidad de cumplir con la estrategia número 6 de ProSoft de la Secretaría de Economía, relativa a “alcanzar niveles internacionales de capacidad de procesos” por parte de las pequeñas empresas mexicanas desarrolladoras de software (NYCE, 2013).

Es así como surge MoProSoft, por iniciativa de la Secretaría, académicos y empresarios mexicanos, encabezados por la Dra. Hanna Oktaba, profesora de la Facultad de Ciencias de la UNAM. MoProSoft es un modelo que fomenta la estandarización de su operación a través de la incorporación de las mejoras prácticas en gestión e ingeniería de software (Ventura Miranda Y Peñaloza Báez, 2006).

### 2.1.2.- Modelo de procesos.

En la actualidad, el software es una herramienta necesaria en las empresas porque les permite establecer dinámicas laborales, convivencia en todo el mundo, entre otras cosas. Sin embargo para desarrollar software o simplemente para desarrollar un producto en particular, la calidad del mismo está relacionada con el proceso por el que es generado.

Un proceso es un conjunto de prácticas relacionadas entre sí, llevadas a cabo a través de roles y por elementos automatizados, que mediante recursos y a partir de insumos, producen un satisfactor de negocio para el cliente (Ventura Miranda Y Peñaloza Báez, 2006).

Un modelo de procesos es un conjunto estructurado de elementos que describen las características de procesos efectivos y de calidad, indicando “qué hacer”, no “cómo hacer” ni “quien lo hace” (Ventura Miranda Y Peñaloza Báez, 2006).

Algunos estándares o modelos de procesos que hoy en día son reconocidos internacionalmente son:

Capability Maturity Model Integrated (CMMI) Implica un marco de trabajo en donde se describen las prácticas esenciales de un proceso, fue creado en Estados Unidos y es uno de los modelos que se ha convertido en una referencia internacional de la capacidad de los procesos de desarrollo de software. CMMI Define 25 áreas de proceso de ámbito de la ingeniería de software y la ingeniería de sistemas (Ventura Miranda Y Peñaloza Báez, 2006).

ISO 9000 es una familia de normas que sirve como guía a las organizaciones de todo tipo y tamaño en la implantación y operación de Sistemas de Gestión de la Calidad. ISO 9000 en su versión 2000 tiene un fuerte enfoque en los procesos y en la satisfacción del cliente; además de que es certificable en México. Sin embargo, por su generalidad, es abstracto y difícil de adecuar al desarrollo de software (Ventura Miranda Y Peñaloza Báez, 2006).

Estos modelos ofrecen un conjunto de procesos para que las diferentes empresas puedan adoptarlos y así estandarizar sus actividades, asegurando productos de software de mayor calidad.

En México, la sexta estrategia de ProSoft está dedicada a mejorar la capacidad de procesos de las empresas en el sector de las TI y el modelo MoProSoft es una de sus iniciativas.

### 2.1.3.- Modelo MoPoSoft.

MoProSoft es un modelo integrado por 9 procesos, donde las salidas de cada proceso están claramente dirigidas como entradas de otros; las prácticas de planeación, seguimiento y evaluación se incluyen en todos los procesos de gestión y administración ; los objetivos, los indicadores, la mediciones y la metas cuantitativas fueron incorporados de manera congruente y práctica en todos los procesos; las verificaciones, validaciones y pruebas están incluidas dentro de las actividades de los procesos; y existe una base de conocimientos que resguarda todos los documentos y productos generados (Ventura Miranda Y Peñaloza Báez, 2006).

Este modelo está constituido por tres categorías principales como se muestra en la Figura 1, mismos que son:

* Alta Dirección (DIR)
* Gestión (GES)
* Operación (OPE)

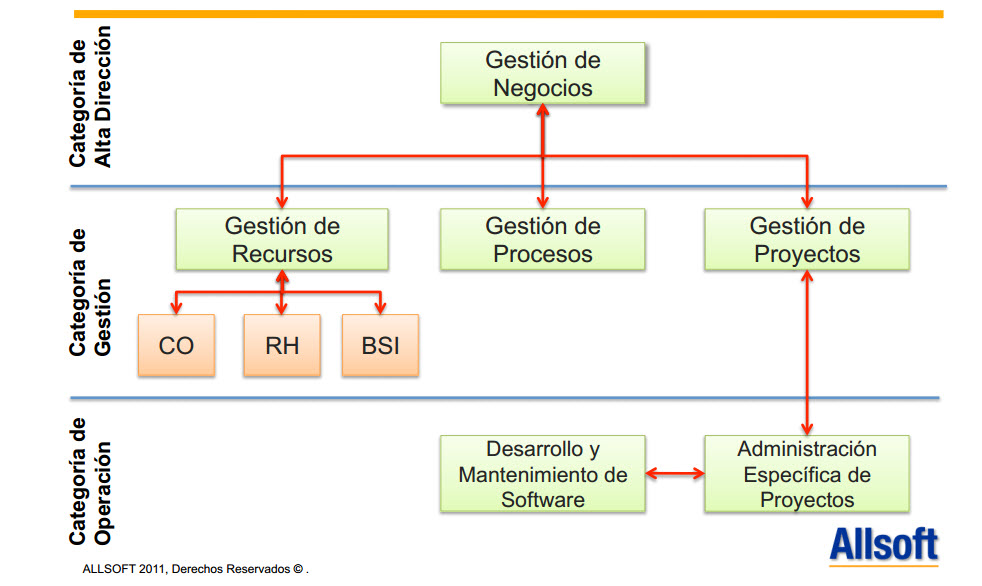


Figura 1 Categorías principales de MoProSoft

Alta Dirección (DIR). Esta categoría se enfoca a las prácticas relacionadas con la gestión del negocio, también proporciona los lineamientos a los procesos de la categoría de gerencia y recibe la retroalimentación requerida a partir de la información generada por estos (Guardati y Ponce, 2011).

Gerencia (GES). Esta categoría se enfoca a las prácticas relacionadas con la gestión de procesos, proyectos y recursos de acuerdo a los lineamientos establecidos por la categoría Alta Dirección. También proporciona los elementos necesarios a los procesos de la categoría operación, recibe y evalúa la información generada por la categoría de operación y comunica los resultados a la categoría de Alta Dirección (Guardati y Ponce, 2011).

Operación (OPE). Esta categoría se enfoca a las prácticas de los proyectos de desarrollo y mantenimiento de software. Realiza actividades según los elementos dados por la categoría de Gerencia y comunica a éste sus resultados (Guardati y Ponce, 2011).

Este modelo fue elaborado con el propósito de que las PyMes sean más competitivas y permanezcan más tiempo en el mercado. Provee un conjunto de procesos los cuales pueden ser adoptados, ajustándolos a las características de cada organización. MoProSoft es fácil de entender y uno de los beneficios que proporciona a quien lo implementa es que no es costoso.

### 2.1.4.- Herramientas de gestión de proyectos.

Productividad, eficiencia, optimización son algunas de las palabras importantes para las empresas. El hablar de optimización nos dirige a la palabra software, mismo que permite a las empresas a hacer la vida algo más fácil y ayudar a realizar tareas que se ejecutan a diario. Sin embargo, el hecho de que se cuente con una herramienta no significa que la organización sea más productiva. El utilizar una herramienta para gestionar las tareas no va a solventar el caos organizativo, sino que se requerirá de constancia y una metodología, estándar o modelo que ayude a sacar el valor a la herramienta que se haya seleccionado.

Algunas de las herramientas más utilizadas para gestionar los proyectos son:

Trello. Esta herramienta es fácil de usar, ofrece herramientas para registrar tareas, adjuntar archivos para dotar de contexto las tareas subiéndolos directamente al sistema o bien vincular archivos almacenados en Google Drive o Dropbox, se pueden incluir en las tareas checklist para enumerar requisitos o pasos a cumplir antes de dar por finalizada una tarea, vistas de calendario, un Card Aging que permite localizar las tareas que llevan más tiempo en el sistema y votaciones una funcionalidad que permite sondear la opinión de los compañeros de equipo y debatir dentro de un card y así evitar decenas de correos (Bitelia, 2013).

[GanttProject](http://www.ganttproject.biz/): Un programa de escritorio multiplataforma que corre sobre Windows, Mac OS X y Linux, totalmente gratuito. Incluye diagramas de Gantt, asignación de las personas que trabajarán en el proyecto, y permite exportar los diagramas como imágenes, mientras genera informes en PDF y HTML. Permite interoperar con Microsoft Project, importando y exportándolos a sus formatos

Podemos concluir que MoProSoft es un modelo que ha empezado a tener auge en México además de que es un modelo que fue creado específicamente para las pequeñas y medianas empresas que son la mayoría de las empresas que hay en México y que los sistemas que ayuden a automatizar la información en una empresa siempre será un gran aporte.

## 2.2.- Tecnologías de desarrollo web.

### 2.2.1.-Herramientas.

#### 2.2.1.1.- Eclipse.

#### 2.2.1.2.- SQL developer.

#### 2.2.1.3.- Maven.

#### 2.2.1.4.- Github.

### 2.2.2.- Frontend.

#### 2.2.2.1.- Definición.

#### 2.2.2.2.- Css.

#### 2.2.2.3.- Html 5.

#### 2.2.2.4.- Bootstrap.

#### 2.2.2.5.- Angular 2

#### 2.2.2.6.- Jsp.

#### 2.2.2.7.- JavaScript.

### 2.2.3.- Back end

#### 2.2.3.1.- ¿Qué es un Backend?



Ilustración 1 BackEnd

Antes de empezar a hablar de Backend, explicaremos brevemente el concepto **Frontend**, ya que, siendo ambos conceptos indivisibles, el segundo nos será mucho más cercano y nos ayudará a comprender el primero más fácilmente.

En pocas palabras**, el Frontend de una app es la parte que el usuario ve y toca.**En él, se incluyen imágenes, botones, menús, transiciones… **Es la capa más superficial de la app, cuyos elementos son incapaces de funcionar por sí mismos.**Es aquí donde el Backend interviene, dando vida a todo lo anterior.

**El Backend,**también conocido como CMS o Backoffice, es **la parte de la app que el usuario final no puede ver. Su función es acceder a la información que se solicita, a través de la app, para luego combinarla y devolverla al usuario final.**

Ahora, que ya conoces las diferencias y las relaciones entre el Frontend y el Backend de una app, estamos listos para seguir profundizando en los procesos y las tareas que lleva a cabo el segundo.

#### 2.2.3.2.- ¿Cómo funciona un Backend?

**Las funciones del Backend**son las siguientes:

* **Acceder a la información que se pide, a través de la app:**cuando usamos una aplicación móvil pedimos información de manera continua, no importa si la app es de búsqueda de información, un juego o una red social. Esto implica que una parte de la app (el Backend), tiene que ser capaz de encontrar y acceder a la información que solicitemos. El proceso de búsqueda de datos no es fácil, ya que estos se almacenan en grandes bases de datos (en plural), que se encuentran, además, protegidos para no exponer lo que en nuestra área se denomina información sensible. En este punto, un Backend bien diseñado debe ser capaz no sólo de encontrar la información precisa que el usuario requiere, sino también de acceder a ella de manera segura.
* **Combinar la información encontrada y transformarla:**Una vez encontrada, el Backend combina la información para que resulte útil al usuario. Pongamos, como ejemplo, una aplicación de transporte y una orden de búsqueda: “cómo llegar del trabajo a casa”.

En este caso, nuestra aplicación necesitará acceder a las bases de datos no sólo de todas las compañías de autobuses de la ciudad, sino también las de las empresas de taxis, metro y, por supuesto, Google Maps. La ingente cantidad de información con la que el Backend trabaja, hace que su diseño deba ser sumamente preciso, ya que debe ser capaz de encontrar y filtrar lo que es relevante de lo que no, para luego combinarlo de manera útil.

* **Devolver la información al usuario:**Finalmente, el Backend **envía la información relevada de vuelta al usuario.**Pero, ¿cuántos usuarios son capaces de leer datos escritos en código puro? Pocos. Es por ello que el Backend necesita de traductores capaces de convertir los datos escritos en código a lenguaje humano. Es aquí donde intervienen las famosas APIs, trabajando en conjunto con el Frontend.

En pocas palabras, las APIs son las herramientas encargadas de transportar la información desde el Backend hasta el Frontend, que es donde el proceso final de traducción toma forma, y donde la información escrita en código se convierte en los diseños, las imágenes, las letras y los botones que el usuario final entiende y con los que puede interactuar.

Este proceso es hecho por en Frontend en dos fases, que tienen lugar en dos subcapas que conforman su estructura:

* **La subcapa lógica, relacionada con el lenguaje específico,**en el que la app ha sido desarrollada. Como ya sabemos, las apps pueden ser desarrolladas para distintos sistemas operativos -iOS, Android, WindowsPhone…- que requieren ser escritas en distintos lenguajes de código. En este punto, la subcapa lógica del Frontend se encarga de hacer una primera traducción del lenguaje en el que las APIs envían la información al lenguaje específico de cada sistema operativo. Este es un proceso que, a pesar de ocurrir en el Frontend de nuestra app, el usuario final no ve. Podríamos llamarlo el “Backend del Fronted”.

* **La subcapa visual, relacionada con el diseño estético de la app,**en la que se encuentran todos los elementos que el usuario final puede ver. La capa visual del Frontend es donde, finalmente, se produce la conversión de la información a los elementos con el que el usuario final se relaciona, y a los que hacíamos referencia al comenzar este artículo.

Para concluir con las funciones y procesos del Backend,**Ignacio Barrea, director y cofundador de Ticapps, empresa especializada en consultoría y desarrollo mobile apunta: “El desarrollo de un Backend sólido es clave para conseguir una buena experiencia de usuario.**Podemos desarrollar los mejores y más novedosos diseños, podemos tener la mejor idea para un negocio mobile pero, al final, **si los cimientos de la app fallan, nuestra aplicación acabará siendo un fracaso”.**

#### 2.2.3.3.- Técnicas usadas en Backend y recomendaciones finales.

**Las técnicas utilizadas para el desarrollo del Backend en término de lenguajes de programación son muchas, pero todas tiene un objetivo en común: deben ser intuitivas y comprensibles para los desarrolladores.** Gran parte del código de la app se encuentra escrito en el Backend, lo que implica que para cualquier actualización o modificación de la misma, los desarrolladores tendrán que volver a trabajar sobre el mismo. Por eso, para facilitar su trabajo y conseguir mejores resultados, el código debe ser claro e intuitivo.

Entre los lenguajes más populares de Backend se encuentran Ruby, Python, SQL, PHP y Java. T**odos tienen ventajas e inconvenientes, por lo que dependerá de las necesidades específicas de nuestra app el escoger uno u otro.**Sin ánimo de profundizar en los lenguajes de programación, **desde Ticapps recomendamos usar aquellos que cumplan con los siguientes criterios:**

* **Escalabilidad:**cuando hablamos de la escalabilidad del Backend, nos referimos a la **flexibilidad**del mismo**para integrar nuevas estructuras y códigos**. Este es un aspecto esencial, especialmente si nuestra app ha sido diseñada para utilizarse a largo plazo. ¿Por qué? **Porque los cambios en mobile son inevitables.**Cada día nuevos sistemas operativos y dispositivos móviles son lanzados al mercado, por no hablar de que nuestro modelo de negocio mobile puede -y debe- evolucionar junto a mismo. En un contexto tan dinámico, la flexibilidad de nuestra app y su Backend es clave para el éxito.
* **Seguridad: debido a la constante interacción del Backend con las bases de datos, desarrollar el código siguiendo prácticas seguras es sumamente importante,**más aún si nuestra app está diseñada para manejar información sensible, como datos personales, financieros o médicos. Para el desarrollo de un Backend seguro, desde Tictapps recomendamos trabajar sólo con desarrolladores cualificados, hacer uso de conexiones seguras -como las famosas HTTPS- y utilizar bases de datos encriptadas.
* **Robustez: con esto nos referimos a la “fuerza” del Backend, o su capacidad para funcionar en cualquier contexto.** Imaginen, por ejemplo, que desarrollamos una app para ser un mobile commerce. En este caso, la aplicación deberá desarrollarse teniendo en cuenta situaciones inesperadas, como el registro y uso masivo de la app en fechas de venta clave -Navidad, Rebajas-. Un Backend desarrollado de manera robusta nos asegura que, ante este tipo de situaciones, nuestra app siga funcionando, evitando los famosos crashes, que dan lugar a malas experiencias de uso y por lo tanto a la temida desinstalación de la app.

#### 

#### 2.2.3.4.- Lenguajes para Back-End

Para el backend existen muchas opciones, todo depende del lenguaje de programación o de las necesidades del rendimiento. Una tendencia al alza es alejarse del backend, convirtiendo esa capa en una API consumida por el frontend y las aplicaciones móviles. Sin embargo, utilizar un framework es a menudo más simple y más rápido, y sigue siendo una opción válida para un montón de aplicaciones web.

**Node.js:** es la principal manera de ejecutar JS fuera del navegador. Ha tenido varias actualizaciones este año, que aumentaron su rendimiento y ampliaron su soporte a ES6. Node cuenta con frameworks para la creación rápida de APIs, servidores, aplicaciones de escritorio e incluso bots, y una vasta comunidad que crea todo tipo de módulos imaginables. Puedes echarle un vistazo a estos frameworks: Express, Koa, Next, Nodal.

**PHP**: Es un lenguaje web y cuenta con un gran número de frameworks donde elegir. Gracias a su excelente documentación y características, Laravel es una excelente opción, por ejemplo. Zend Framework lanzó la versión 3, que supuso una gran actualización para este marco orientado a los negocios. Symfony también vio un montón de nuevas actualizaciones este año, lo que la convierte también en una opción a tener muy en cuenta.

**Ruby:** el framework Rails es definitivamente un must. La versión 5.0 fue lanzada este 2016, con soporte para Web Sockets, modo API y mucho más. Sinatra también es una muy buena opción para aplicaciones pequeñas, y su versión 2.0 se espera para algún momento de 2017.

**Python:** definitivamente debes echar un vistazo a Django y a Flask. Django 1.10 se lanzó en agosto introduciendo la búsqueda por texto completo en Postgres y muchas más características.

**Java:** también cuenta con frameworks populares donde elegir. Play y Spark son dos opciones sólidas, y como bonus, decir que también pueden utilizarse con Scala.

#### 2.2.3.3.- Servlets.

¿Qué son los servlets?

Los Servlets son módulos escritos en Java que se utilizan en un servidor, que puede ser o no ser servidor web, para extender sus capacidades de respuesta a los clientes al utilizar las potencialidades de Java. Los Servlets son para los servidores lo que los applets para los navegadores, aunque los servlets no tienen una interfaz gráfica.

Los servelts pueden ser incluidos en servidores que soporten la API de Servlet (ver servidores). La API no realiza suposiciones sobre el entorno que se utiliza, como tipo de servidor o plataforma, ni del protocolo a utilizar, aunque existe una API especial para HTTP.

Los Servlets son un reemplazo efectivo para los CGI en los servidores que los soporten ya que proporcionan una forma de generar documentos dinámicos utilizando las ventajas de la programación en Java como conexión a alguna base de datos, manejo de peticiones concurrentes, programación distribuida, etc. Por ejemplo, un servlet podría ser responsable de procesar los datos desde un formulario en HTML como registrar la transacción, actualizar una base de datos, contactar algún sistema remoto y retornar un documento dinámico o redirigir a otro servlet u alguna otra cosa.

Propiedades.

* **Manejo de Sesiones:** Se puede hacer seguimiento de usuarios a través de distintos servlets a través de la creación de sesiones.
* **Utilización de Cookies:** Las *cookies* son pequeños datos en texto plano que pueden ser guardados en el cliente. La API de servlets permite un manejo fácil y limpio de ellas.
* **Multi-thread:** Los servlets soportan el acceso concurrente de los clientes, aunque hay que tener especial cuidado con las variables compartidas a menos que se utilize la interfaz **SingleThreadModel**.
* **Programación en Java:** Se obtienen las características de multiplataforma o acceso a APIs como JDBC, RMI, etc.

### 2.2.4.- Modelo MVC

#### 2.2.4.1.- Definición.

Modelo-Vista-Controlador (MVC) es una propuesta de diseño de software utilizada para implementar sistemas donde se requiere el uso de interfaces de usuario. Surge de la necesidad de crear software más robusto con un ciclo de vida más adecuado, donde se potencie la facilidad de mantenimiento, reutilización del código y la separación de conceptos.

MVC es un "invento" que ya tiene varias décadas y fue presentado incluso antes de la aparición de la Web. No obstante, en los últimos años ha ganado mucha fuerza y seguidores gracias a la aparición de numerosos frameworks de desarrollo web que utilizan el patrón MVC como modelo para la arquitectura de las aplicaciones web**.**

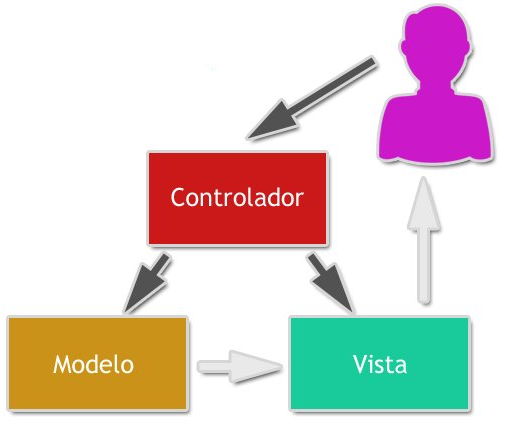


Ilustración 2.4.1 Diagrama de MVC

**Modelos**

Es la capa donde se trabaja con los datos, por tanto, contendrá mecanismos para acceder a la información y también para actualizar su estado. Los datos los tendremos habitualmente en una base de datos, por lo que en los modelos tendremos todas las funciones que accederán a las tablas y harán los correspondientes selects, updates, inserts, etc.

No obstante, cabe mencionar que cuando se trabaja con MCV lo habitual también es utilizar otras librerías como PDO o algún ORM como Doctrine, que nos permiten trabajar con abstracción de bases de datos y persistencia en objetos. Por ello, en vez de usar directamente sentencias SQL, que suelen depender del motor de base de datos con el que se esté trabajando, se utiliza un dialecto de acceso a datos basado en clases y objetos.

**Vistas**

Las vistas, como su nombre nos hace entender, contienen el código de nuestra aplicación que va a producir la visualización de las interfaces de usuario, o sea, el código que nos permitirá renderizar los estados de nuestra aplicación en HTML. En las vistas nada más tenemos los códigos HTML y PHP que nos permite mostrar la salida.

En la vista generalmente trabajamos con los datos, sin embargo, no se realiza un acceso directo a éstos. Las vistas requerirán los datos a los modelos y ellas se generará la salida, tal como nuestra aplicación requiera.

**Controladores**

Contiene el código necesario para responder a las acciones que se solicitan en la aplicación, como visualizar un elemento, realizar una compra, una búsqueda de información, etc.

En realidad, es una capa que sirve de enlace entre las vistas y los modelos, respondiendo a los mecanismos que puedan requerirse para implementar las necesidades de nuestra aplicación. Sin embargo, su responsabilidad no es manipular directamente datos, ni mostrar ningún tipo de salida, sino servir de enlace entre los modelos y las vistas para implementar las diversas necesidades del desarrollo.

#### 2.2.4.2.- RestFull.

**¿Que es REST?**

Esta pregunta es una de las más habituales en nuestros días. Para algunas personas REST es una arquitectura, para otras es un patrón de diseño, para otras un API. **¿Que es REST exactamente?** REST o Representational State Transfer es un **ESTILO** de Arquitectura a la hora de realizar **una comunicación entre cliente y servidor.**



Ilustración 3Representación cliente y servidor.

##### 2.2.4.2.1.- ¿Que es REST ? (Nivel 1 Recursos).

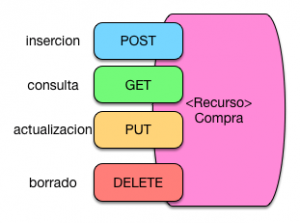
El siguiente **paso es lo que se denomina nivel 1** , en vez de tener servicios con métodos diversos declaramos Recursos. **¿Qué es un Recurso?** . Un recurso hace referencia a un concepto importante de nuestro negocio**(**Facturas ,Cursos , Compras etc)**.**Este estilo permite un primer nivel de organización permitiendo acceder a cada uno de los recursos de forma independiente, favoreciendo la reutilización y aumentando la flexibilidad.

[](https://www.arquitecturajava.com/wp-content/uploads/QueesRESTFactura-1.png)

##### 2.2.4.2.2.- ¿Que es REST? (Nivel 2 HTTP Verbs).

Hasta este momento para realizar las peticiones se usa GET o POST indistintamente.  En el nivel 2 las operaciones**pasan a ser categorizadas de forma más estricta**. Dependiendo de cada tipo de operación se utilizará **un método diferente de envío.**

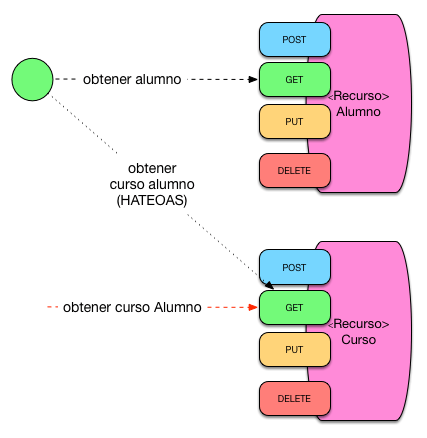
1. **GET:** Se usara para solicitar consultar a los recursos
2. **POST:** Se usará para insertar nuevos recursos
3. **PUT :** Se usará para actualizar recursos
4. **DELETE :** Se usará para borrar recursos

[](https://www.arquitecturajava.com/wp-content/uploads/QueesRESTVerbos.png)

Este es el nivel en el que hoy en día **se encuentran muchas de las Arquitecturas REST**.

##### 2.2.4.2.3.- HATEOAS (Nivel 3).

Todavía nos queda un nivel más que es el que se denomina**HATEOAS (Hypertext As The Engine Of Application State). ¿Para qué sirve este nivel?** Supongamos que queremos acceder a un recurso Alumno vía REST. Si tenemos**una Arquitectura a nivel 2** primero accederemos a ese recurso **utilizando GET.** En segundo lugar, deberemos acceder al recurso de Cursos para añadir al alumno **al curso (línea roja)**.

[](https://www.arquitecturajava.com/wp-content/uploads/QueesRESTUsoHATEOAS.png)

Esto implica que el cliente que accede a los servicios REST**asume un acoplamiento muy alto, debe conocer la url del Alumno y la del Curso**. Sin embargo,**si el recurso del Alumno contiene un link al recurso de Curso esto no hará falta.** Podríamos tener una estructura JSON como la siguiente:

{nombre:pedro, apellidos:”gomez”,**cursos:”http://miurl/cursos”**}

Podremos acceder directamente al recurso de Curso utilizando las propiedades del Alumno esto es **HATEOAS**. De esta forma se aumenta la flexibilidad **y se reduce el acoplamiento.** Construir arquitecturas sobre estilo REST no es sencillo **y hay que ir paso a pas.**

##### 

#### 2.2.4.3.- Spring.

##### 2.2.4.3.1.- ¿Qué es Spring?

**Spring es un framework del lenguaje de programación java**, y un framework en programación es el resultado de la evolución de la ingeniería del software, estos son creados por programadores para programadores, con la finalidad de estandarizar el trabajo, resolver, agilizar y manejar los problemas y complejidades que van apareciendo en el mundo de la programación, a medida las exigencias van creciendo. Creando así, en la comunidad de desarrolladores, un abanico de posibilidades para una creación cada vez más evolucionada de aplicaciones.

Spring nos permite desarrollar aplicaciones de manera más rápida, eficaz y corta, saltándonos tareas repetitivas y ahorrándonos líneas de código.

Spring framework es muy extenso y crece día a día para ayudar al desarrollo de aplicaciones web. A continuación, les vamos a explicar una de sus funciones básicas, la inyección de dependencias de Spring (Spring Di).

##### 2.2.4.3.2.- INVERSION DE CONTROL (IOC).

**Inversión de control es un término** que siempre ha generado bastante confusión en la comunidad. Pero lo que hay que saber es que es una expresión que se usa para referirse al **cambio en el flujo de ejecución y vida de los objetos con respecto a la programación tradicional**, básicamente de lo que se trata es de invertir la forma en que se controla la aplicación, lo que antes dependía del programador, como por ejemplo, el orden en que se llaman los métodos para darle un comportamiento particular a la aplicación, ahora depende completamente de otro ente, en este caso, el framework, todo esto con la finalidad de crear aplicaciones más complejas y con funcionamientos más encapsulados y automáticos.

##### 2.2.4.3.3.- INYECCIÓN DE DEPENDENCIA (DI) EN SPRING

**La**[inyección de dependencia](http://curiotek.com/2015/01/16/java-que-es-la-inyeccion-de-dependencias/), **es un tipo de inversión de control**, donde el manejo de las propiedades de un objeto son inyectadas a través de un constructor, un setter, un servicio, etc. Creando de esta manera un control diferente para un comportamiento más conveniente en nuestra aplicación, como se mencionó anteriormente.

Sabiendo esto, una de las **características principales de spring framework** es el **Spring container**. El *Spring Container* es un **IoC Container** (Contenedor de Inversion de control, por sus siglas en inglés), estos son frameworks o partes de un framework, en este caso, que**se encargan de realizar inyección de dependencia sin la necesidad de realizar la definición en java (o el lenguaje de POO que estemos utilizando) a través de archivos de metadata**, por ejemplo, un **xml**.

##### 2.2.4.3.4.- Módulos

Spring Framework comprende diversos módulos que proveen un rango de servicios:

Contenedor de inversión de control: permite la configuración de los componentes de aplicación y la administración del ciclo de vida de los objetos Java, se lleva a cabo principalmente a través de la inyección de dependencias.

Programación orientada a aspectos: habilita la implementación de rutinas transversales.

**Acceso a datos:** se trabaja con RDBMS en la plataforma java, usando Java Database Connectivity y herramientas de Mapeo objeto relacional con bases de datos NoSQL.

**Gestión de transacciones:** unifica distintas APIs de gestión y coordina las transacciones para los objetos Java.

**Modelo vista controlador:** Un framework basado en HTTP y servlets, que provee herramientas para la extensión y personalización de aplicaciones web y servicios web REST.

**Framework de acceso remoto:** Permite la importación y exportación estilo RPC, de objetos Java a través de redes que soporten RMI, CORBA y protocolos basados en HTTP incluyendo servicios web (SOAP).

**Convención sobre Configuración:** el módulo Spring Roo ofrece una solución rápida para el desarrollo de aplicaciones basadas en Spring Framework, privilegiando la simplicidad sin perder flexibilidad.

**Procesamiento por lotes:** un framework para procesamiento de mucho volumen que como características incluye funciones de registro/trazado, manejo de transacciones, estadísticas de procesamiento de tareas, reinicio de tareas, y manejo de recursos.

**Autenticación y Autorización:** procesos de seguridad configurables que soportan un rango de estándares, protocolos, herramientas y prácticas a través del subproyecto Spring Security (antiguamente Acegi).

**Administración Remota:** Configuración de visibilidad y gestión de objetos Java para la configuración local o remota vía JMX.

**Mensajes:** Registro configurable de objetos receptores de mensajes, para el consumo transparente desde la a través de JMS, una mejora del envío de mensajes sobre las API JMS estándar.

**Testing:** Soporte de clases para desarrollo de unidades de prueba e integración.

#### 2.2.4.4.- Struts.



#### 2.2.4.4.1.- Que es Struts?

Struts es una herramienta de soporte para el desarrollo de [aplicaciones Web](https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_Web) del [patrón](https://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o) [MVC](https://es.wikipedia.org/wiki/Modelo_Vista_Controlador) bajo la plataforma [Java EE](https://es.wikipedia.org/wiki/Java_EE) (Java Enterprise Edition). Struts se desarrollaba como parte del [proyecto Jakarta](https://es.wikipedia.org/wiki/Jakarta_Project) de la [Apache Software Foundation](https://es.wikipedia.org/wiki/Apache_Software_Foundation), pero actualmente es un proyecto conocido como *Apache Struts*.

Struts permite reducir el tiempo de desarrollo. Su carácter de "[*software libre*](https://es.wikipedia.org/wiki/Software_libre)" y su compatibilidad con todas las plataformas en las que Java Entreprise esté disponible lo convierten en una herramienta altamente disponible.

El 5 de marzo de 2013, Struts 1 llegó al final de su vida y dejó de estar oficialmente soportado.

Struts 2 es un nuevo framework (anteriormente conocido como WebWork 2) que introdujo algunas mejoras sobre Struts 1, de cara a simplificar las tareas más comunes en el desarrollo de aplicaciones web, así como mejorar su integración con [AJAX](https://es.wikipedia.org/wiki/AJAX), etc.

#### 2.2.4.4.2.- Funcionamiento de Struts

Struts se basa en el patrón de arquitectura de software Modelo-Vista-Controlador (MVC) el cual se utiliza ampliamente y es considerado de gran solidez. De acuerdo con este Framework, el procesamiento se separa en tres secciones diferenciadas llamadas el modelo, las vistas y el controlador.

Este framework también tiene como objetivo ayudar a los desarrolladores de aplicaciones Web en Java a crear proyectos de calidad siguiendo una norma o estándar. Este framework también ayuda a los desarrolladores a organizar la lógica de la aplicación.

La elección del framework Struts 2 se basa en los siguientes puntos:

**Fiabilidad:** el proyecto se desarrolló en mayo de 2000 y desde entonces ha tenido un seguimiento. Este proyecto tiene una reputación excelente y mejora constantemente sus defectos.

**Flexibilidad:** cada acción puede ser personalizada, los archivos de configuración son muy flexibles en términos de utilización y las validaciones son fáciles de aplicar.

**Rendimiento:** la arquitectura diseñada por Struts 2 está basada en WebWork. Tiene un rendimiento especialmente alto y su mantenimiento es sencillo gracias a la separación por capas.

Las principales características del framework Struts 2 son las siguientes:

los tipos de conversiones automáticas para las colecciones procedentes de consultas HTTP; los archivos de configuración modulables que se usan por paquetes; las anotaciones Java 5 que reducen las líneas de código para la configuración; el uso de etiquetas permite aplicar temas o plantillas (templates); el uso del lenguaje de expresión OGNL; el uso opcional del complemento interceptor, que permite ejecutar consultas complejas y largas como tareas en segundo plano con el envío múltiple (actualización de páginas); la integración simple de herramientas como JSTL, Spring o Hibernate.

El framework Struts 2 es una segunda generación de framework MVC. La principal ventaja del concepto de los interceptores es la flexibilidad del conjunto y la configuración propuesta. Struts 2 responde también al principio de empaquetamiento de la acciones. Cuando declaramos clases de acción con un archivo XML o anotaciones Java, el framework organiza todos estos componentes en forma de un software de paquete (packages). Los paquetes de Struts 2 son similares a los de Java.

Este mecanismo permite igualmente agrupar las acciones por dominio. Las URL de la aplicación se asociarán a los paquetes donde están declaradas las acciones.

## 2.3.- Gestor de base de datos(SQL developer).

### 2.3.1.- Definición.

Un Sistema Gestor de Bases de Datos (SGBD) o DGBA (Data Base Management System) es un conjunto de programas no visibles que administran y gestionan la información que contiene una base de datos. A través de él se maneja todo acceso a la base de datos con el objetivo de servir de interfaz entre ésta, el usuario y las aplicaciones.

Gracias a este sistema de software invisible para el usuario final, compuesto por un lenguaje de definición de datos, un lenguaje de manipulación y de consulta, es posible gestionar los datos a distintos niveles. Tanto almacenar, modificar y acceder a la información como realizar consultas y hacer análisis para generar informes.

A su vez, el SGBD puede entenderse como una colección de datos relacionados entre sí, estructurados y organizados dentro del ecosistema conformado por ese conjunto de programas que acceden a ellos y facilitan su gestión. Frente al anterior sistema de gestión de archivos, -un conjunto de programas que definían y trabajaban sus propios datos-, el acceso a los datos es independiente de los programas que los gestionan, una gran ventaja de cara a tratar grandes volúmenes de información.

Básicamente, el gestor controla cualquier operación ejecutada por el usuario contra la base de datos. Para ello, se utilizan herramientas específicas, como sistemas de búsqueda y de generación de informes, además de distintas aplicaciones.

Igualmente, permite la definición de bases de datos a distintos niveles de abstracción, así como su manipulación, garantizando tanto la seguridad como la integridad y consistencia de los mismos. Siempre de forma separada a los programas o aplicaciones que los gestionan, sin tener que modificar éstos.

En cuanto a su tipología, suelen agruparse a partir de criterios relacionados con el modelo de datos (dentro de éste encontramos los SGBD relacionales, EnRed, jerárquicos u orientados a objetos), y también es posible diferenciarlos según sean o no propietarios, así como a partir de elementos como el número de usuarios (monousuarios y distribuidos) o de sitios, en cuyo caso serían centralizados o distribuidos.

Oracle SQL Developer es la herramienta gráfica gratuita que proporciona Oracle para que no sea necesario utilizar herramientas de terceros (como el conocido TOAD, o el PL/SQL Developer) para desarrollar, o simplemente para ejecutar consultas o scripts SQL, tanto DML como DDL, sobre bases de datos Oracle.

Oracle SQL Developer es la herramienta gráfica gratuita que proporciona Oracle para que no sea necesario utilizar herramientas de terceros (como el conocido TOAD, o el PL/SQL Developer) para desarrollar, o simplemente para ejecutar consultas o scripts SQL, tanto DML como DDL, sobre bases de datos Oracle.

### 2.3.2.- Instalación y configuración.

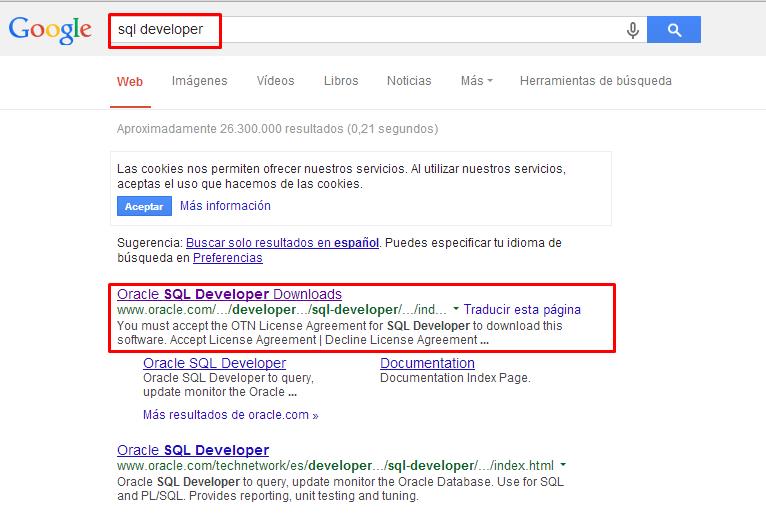
Instalar Oracle SQL Developer

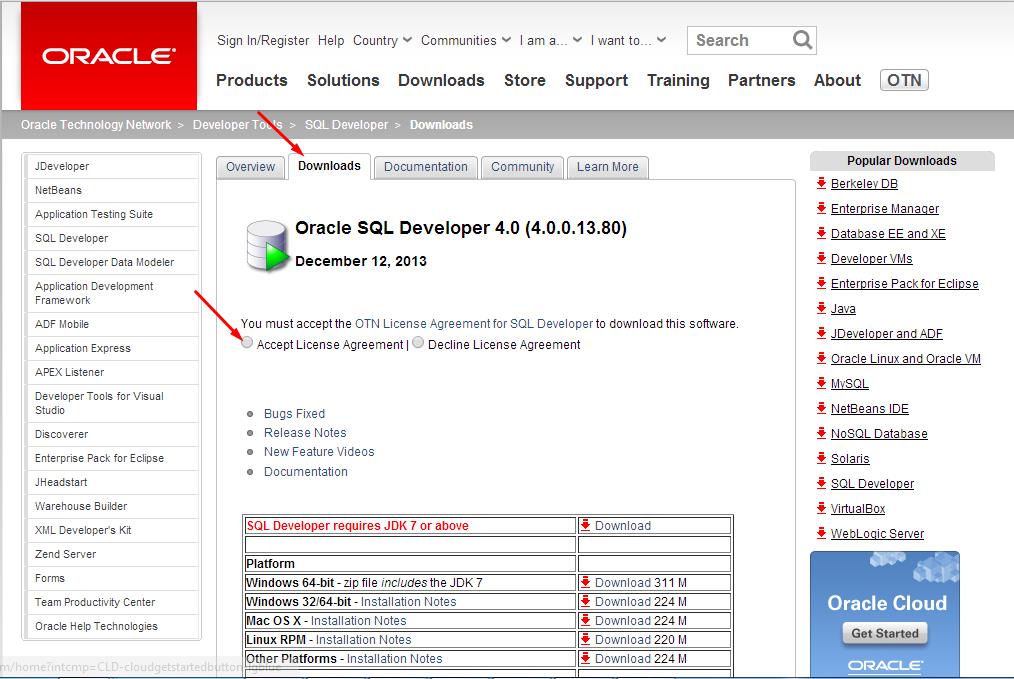
Vamos a ver como instalar de forma rápida el SQL Developer en entorno Windows.

SQL Developer en un programa que no requiere de instalación, se descomprime, se ejecuta y se utiliza.

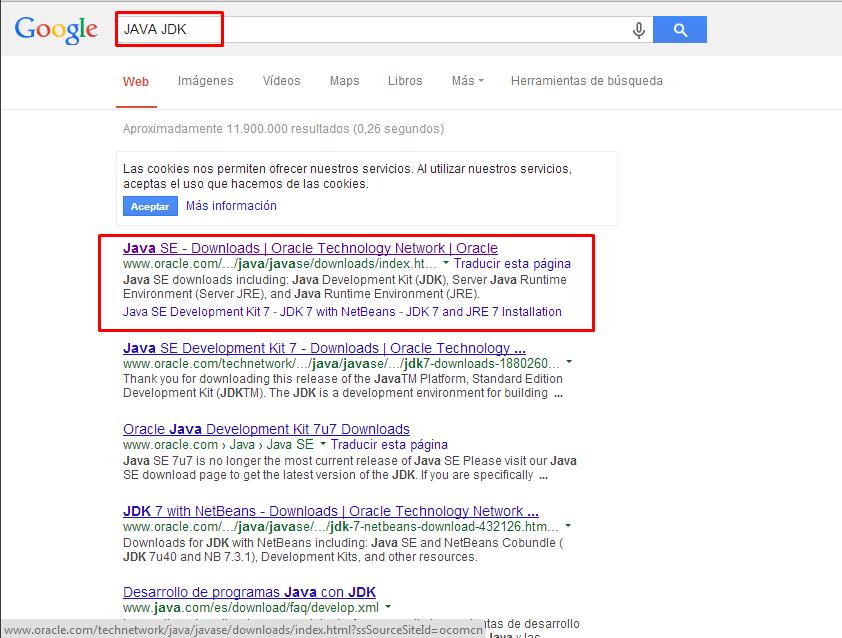
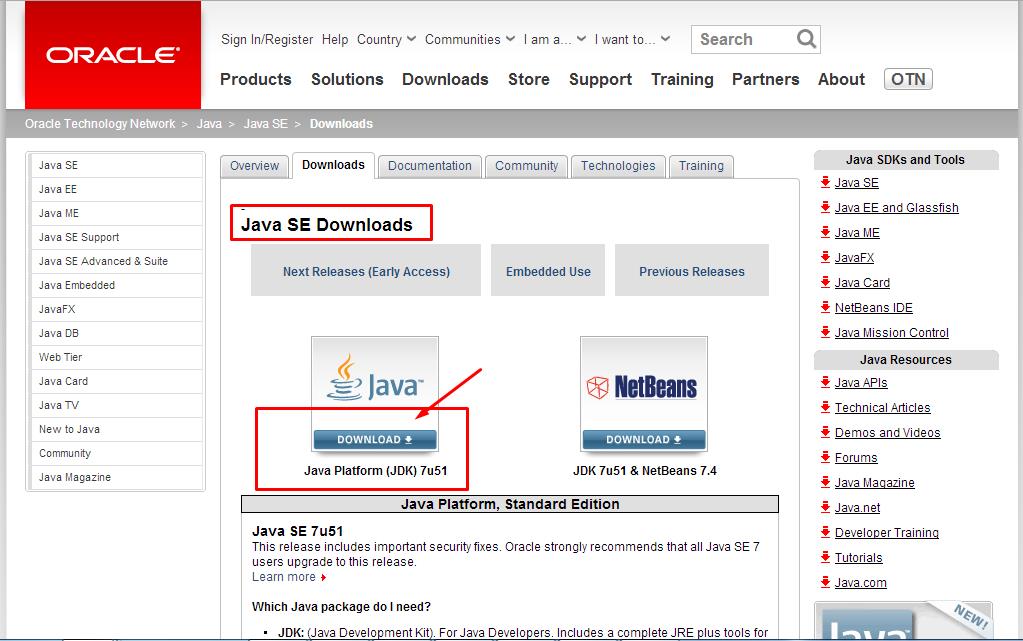
El SQL Developer solo requiere del SDK de java, en el caso caso del SQL Developer 4.0 requiere como mínimo el JDK 7.

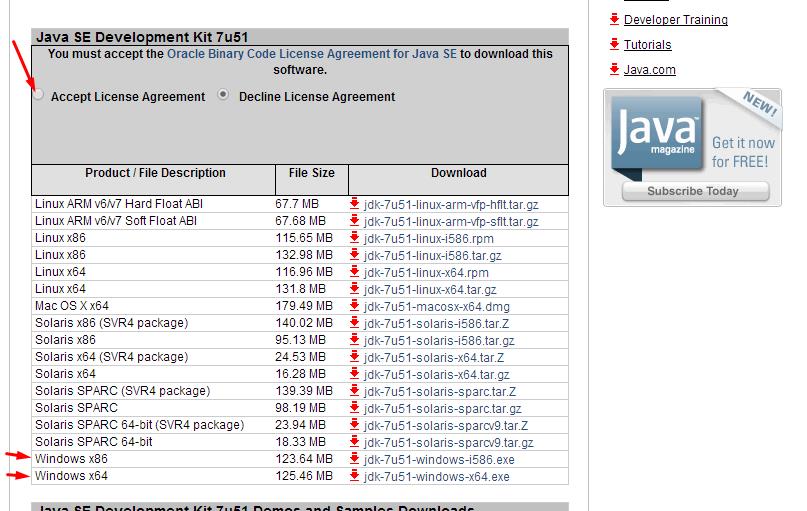
Veamos los pasos para instalar el SQL Developer

* 1. Abrimos nuestro amigo www.google.es y escribimos “SQL Developer” Pulsamos en la primera entrada que nos aparezca (cerciórate que el el domino sea www.oracle.com) [](http://www.informaticosmurcia.es/wp-content/uploads/2014/02/SQL-Developer-instalacion1.jpg)
  2. Pulsamos en “Download”, Aceptamos los términos y nos descargamos la versión que más no interese (Windows 32 bits o 64 bits).

Abrimos www.google.es y escribimos “JAVA JDK” Pulsamos en la primera entrada que nos aparezca (cerciórate [](http://www.informaticosmurcia.es/wp-content/uploads/2014/02/SQL-Developer-instalacion2.jpg)

Ahora nos vamos a descargar el JDK de java, para ello seguimos los siguientes pasos

* 1. que el el domino sea www.oracle.com)[](http://www.informaticosmurcia.es/wp-content/uploads/2014/02/SQL-Developer-instalacion3.jpg)[](http://www.informaticosmurcia.es/wp-content/uploads/2014/02/SQL-Developer-instalacion4.jpg)
  2. Aceptamos la licencia y nos descargamos la versión que mas nos interese

[](http://www.informaticosmurcia.es/wp-content/uploads/2014/02/SQL-Developer-instalacion5.jpg)

Una vez que tengamos los dos ficheros descargados instalamos el JDK, no voy a poner pantallazos porque es muy sencillo instalarlo.

Una vez instalado el JDK descomprimimos el SQL Developer.

Lo ejecutamos y nos saldrá una ventanita como esta para que le indiquemos donde esta el JDK de java si no lo ha encontrado el automáticamente.

[](http://www.informaticosmurcia.es/wp-content/uploads/2014/02/SQL-Developer-instalacion6.jpg)

Una vez puesto el path del Java JDK el programa SQL Developer arrancará.

Configuración:

Paso 1



Haz clic derecho en "Conexiones" debajo de la ficha de conexiones en el lado izquierdo de la ventana.

Paso 2



Haz clic en "Nueva conexión". Un cuadro de diálogo se abrirá.

Paso 3



Escribe un nombre en la caja "Nombre de conexión". Puede ser cualquier cosa que elijas.

Paso 4



Escribe el nombre de usuario y contraseña para tu base de datos en los cuadros de texto correspondientes.

Paso 5



Marca la casilla de verificación "Guardar contraseña" para que SQL Developer guarde la contraseña. Desactiva la casilla de verificación si deseas introducir la contraseña cada vez que te conectas.

Paso 6



Escribe la dirección de IP o nombre de dominio de la base de datos en la caja "Nombre de anfitrión".

Paso 7



Escribe el número de puerto al cual la base de datos está conectada en la caja "Puerto".

Paso 8



Escribe el ID del sistema de la base de datos en la caja "SID".

Paso 9



Haz clic en el botón "Testear". Debes ver un mensaje que diga "éxito".

Paso 10



Haz clic en el botón "Guardar" para guardar la conexión.

Paso 11



Haz clic en el botón "Conectar" para conectar y empezar a usar la base de datos

.

### 2.3.3.- Diseño de bases de datos: Conceptos, Operaciones, Objetos y Estructura.

## 2.4.- Metodología de desarrollo.

### 2.4.1.- Metodología Agil.

#### 2.4.2.- Definición.

#### 2.4.3.- Análisis y Diseño.

##### 2.4.3.1.- Análisis.

La fase de análisis es una de las más importantes etapas del ciclo de vida de un proyecto, pues en esta etapa se detalla y determina el alcance final del proyecto, así como los requerimientos del sistema que se construirá, proponiendo una alternativa para satisfacer los requerimientos.

El objetivo de esta etapa de análisis es descubrir qué es lo que realmente se necesitará y así llegar a una comprensión adecuada de los requerimientos.

Para el proyecto de tesis en esta fase se realizarán actividades como:

* Definir detalladamente los requerimientos del módulo de presupuestos.
* Diseñar una solución funcional para satisfacer los requerimientos definidos.
* Definir la estrategia de pruebas.

##### 2.4.3.2.- Diseño.

La fase del diseño consiste en estudiar posibles alternativas de implementación para el sistema que se ha de construir y se ha de decidir la estructura general que tendrá el sistema, es decir su diseño arquitectónico.

Su objetivo es definir cómo se planea construir la solución en relación al alcance y requerimientos definidos en la fase de análisis.

Esta fase aborda temas como diseñar la base de datos y diseñar las aplicaciones que permitirán que un usuario utilice el sistema que se construirá, como la interfaz de usuario del sistema y sus distintos componentes en que se descompone.

Para el proyecto de tesis se realizarán las siguientes actividades:

* Descripción de elementos de diseño los cuales servirán de base para la construcción.
* Definir la estructura de los componentes de software.
* Generación de casos y matrices de pruebas.

##### 2.4.3.3.- Construcción e integración.

Una vez que las funciones que desempeñará nuestro sistema se han definido y se ha decidido cómo se va a organizar sus distintos módulos o componentes es el momento de pasar a la etapa de desarrollo.

En la etapa de construcción y desarrollo se seleccionan las herramientas adecuadas como el entorno de desarrollo que facilite el trabajo y un lenguaje de programación apropiado para el sistema que se ha de construir, todo esto dependiendo del diseño del sistema.

Para el proyecto de tesis se realizarán las siguientes actividades:

* Producir componentes de software, acorde a su especificación.
* Integrar componentes, asegurándose de que el producto una vez integrado funcione correctamente a través de la ejecución de las matrices de pruebas.

##### 2.4.3.4.- Pruebas.

El objetivo de realizar pruebas es detectar los errores que se hayan podido cometer durante las fases anteriores del proyecto y así poder corregirlos, evitando que esto lo realice el usuario final.

Las pruebas se realizarán por:

Pruebas de unidad: Este tipo de pruebas consiste en comprobar el funcionamiento correcto de cada componente o bien por módulos. En este caso la persona encargada de realizar las pruebas busca solamente situaciones límite de la implementación del componente.

Pruebas de integración: Son las que se realizan cuando se van juntando todos los módulos que integran el sistema y así detectar errores en sus interfaces.

Para el proyecto de tesis se realizarán las siguientes actividades:

* Probar, verificar y validar que los productos finales, cumplen con el propósito final para el que fue diseñado en el ambiente requerido (Pruebas unitarias, Pruebas de integración).
* Obtener producto de software y documentar su descripción formal.

##### 2.4.3.5.- Cierre.

El objetivo de la fase de cierre es definir las estrategias y actividades para liberar el producto en un ambiente productivo. Para este proyecto de tesis esta fase consistirá en la evaluación y aceptación del sistema generado y así su finalización.

# Capítulo 3: Desarrollo.

# Capítulo 4: Resultados.

# Capítulo 5: Conclusiones.

# Capítulo 6: Competencias desarrolladas.

# Bibliografía.