

**UNIVERSIDAD TECNOLÓGICA DEL CENTRO DE VERACRUZ**

**PROGRAMA EDUCATIVO DE:**

TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES

**NOMBRE DEL PROYECTO:**

“Aplicación de estimación de software basada en métricas”

METRISOFT

**PRESENTA:**

Benítez Martínez Janet Anel

Florentino Altamirano Leobardo

Hernández Méndez José Antonio

Lugar y fecha: Cuitláhuac Veracruz 17 de Febrero del 2018

Contenido

[Introducción 3](#_Toc507217565)

[Antecedentes de la empresa 4](#_Toc507217566)

[Historia de IMATIONWEB 4](#_Toc507217567)

[¿Qué Hacemos? 4](#_Toc507217568)

[¿Porque elegirnos? 4](#_Toc507217569)

[Descripción de la problemática 5](#_Toc507217570)

[Objetivos del Proyecto 6](#_Toc507217571)

[Objetivo General 6](#_Toc507217572)

[Objetivos Específicos 6](#_Toc507217573)

[Justificación 7](#_Toc507217574)

[Alcance y limitaciones 10](#_Toc507217575)

[Alcances 10](#_Toc507217576)

[Limitaciones 10](#_Toc507217577)

[Marco Teórico 11](#_Toc507217578)

[Calidad 11](#_Toc507217579)

[Medición 11](#_Toc507217580)

[Medida 11](#_Toc507217581)

[Modelos de calidad de software 11](#_Toc507217582)

[Factores de calidad de McCall 12](#_Toc507217583)

[Métrica 14](#_Toc507217584)

[Indicador 14](#_Toc507217585)

[Métricas del proceso 14](#_Toc507217586)

[Métricas del proyecto 15](#_Toc507217587)

[Métricas del producto 15](#_Toc507217588)

[Roles y responsabilidades 15](#_Toc507217589)

[Origen de Java 17](#_Toc507217590)

[Programación Orientada a Objetos 19](#_Toc507217591)

[Base de Datos (Hidalgo, s.f.) 20](#_Toc507217592)

[JasperReports (EcuRed, s.f.) 22](#_Toc507217593)

[JavaFX Scene Builder (Blog Bitix, 2015) 22](#_Toc507217594)

[PosgreSQL (Ubuntu, s.f.) 29](#_Toc507217595)

[Cronograma 34](#_Toc507217596)

[Bibliografía 35](#_Toc507217597)

# Introducción

En el presente proyecto se trata de resolver la problemática presentada en clase en cuanto al diseño de una aplicación de estimación de software basada en métricas, tiene como objetivo la creación e implementación de un software desarrollado en un lenguaje de programación orientado a objetos (POO) capaz de ejecutarse gráficamente, cumpliendo así con diversos requerimientos del usuario de manera correcta.

En este documento se expondrá más a detalle este proceso, al igual que diversos complementos añadidos durante su creación, ideas, comentarios y ejemplos, por otra parte se mostrarán las diversas problemáticas a resolver.

Se pretende mostrar los objetivos del proyecto donde se describirá el objetivo general y los objetivos específicos a diversos plazos. A partir del diccionario de base de datos, el diagrama de entidad-relación y las herramientas de desarrollo a utilizar se podrán consultar las diversas metodologías que fueron implementadas durante la creación del mismo, al igual que palabras claves y conceptos fundamentales.

Se dan a conocer los roles que desempañan cada uno de los integrantes del equipo, y como se le describe a cada uno de ellos.

# Antecedentes de la empresa

## Historia de IMATIONWEB

Somos una empresa Mexicana que Brindamos Soluciones Integrales de Diseño y Desarrollo Web, Digital y Sistemas a la Medida con la finalidad de posicionar el negocio de nuestros clientes en Internet. Desde nuestros inicios en el año 2006 hemos promovido la innovación en ideas, procesos, y tecnología contribuyendo al cambio, la productividad y la competitividad en los negocios.

## ¿Qué Hacemos?

Nuestra principal experiencia se centra principalmente en crear las mejores páginas web optimizadas y que sean visibles tanto en pc como dispositivos móviles, y posicionar tu negocio en las primeras páginas de los buscadores, para lograr el objetivo de todo negocio, el cual es generar ventas. Además diseñamos y desarrollamos sistemas a la medida y aplicaciones web.

## ¿Porque elegirnos?

Somos una empresa exigente con nosotros mismos, buscamos siempre en cada proyecto, la plena satisfacción del cliente, manteniéndonos actualizados para ofrecer una solución óptima. Estamos certificados en nuestras áreas de servicio que ofrecemos, y contamos con la experiencia necesaria para hacer de tu proyecto un negocio rentable en internet.

# Descripción de la problemática

En la actualidad, la mayoría de las empresas de software requieren soporte de diferentes de subprocesos que conforman el proceso general de desarrollo y por lo regular no cuentan con un sistema a la medida que les permita realizar un control sobre el análisis y medición de sus proyectos en proceso de desarrollo para compararlos con las líneas base y tomar acciones en caso de que lo requieran.

La empresa IMATIONEB requiere una aplicación que permita registrar los diferentes proyectos, consultarlos y modificarlos de ser necesario, al igual que registrar las métricas establecidas y aplicar con ellas dichas formulas en el momento que se seleccione para se seleccione para un periodo determinado.

Es necesario contar con un repositorio de medidas y mediciones de todos los proyectos, de igual forma se deberá graficar el comportamiento de un proyecto con respecto a sus métricas y es importante que las interfaces gráficas de la aplicación sean sencillas y amigables. Indispensable controlar el acceso a la aplicación sólo para usuarios registrados.

Entre las funciones que debe tener la herramienta están:

* Administrar Usuarios: El administrador de la aplicación podrá administrar los usuarios (agregar, modificar, eliminar, buscar, listar, iniciar sesión).
* Administrar Proyectos: El administrador de la aplicación podrá administrar los proyectos (agregar, modificar, eliminar, buscar, listar). Un usuario normal sólo podrá listar y consultar.
* Administrar la recolección de medidas y aplicación de métricas
* Generar reportes y gráficas

# Objetivos del Proyecto

## Objetivo General

Desarrollar las funciones necesarias que permitan la realización de la creación de una aplicación para la estimación de software basada en métricas y así mejorar la calidad del software.

## Objetivos Específicos

* Identificar las necesidades específicas de evolución que surgen en cada etapa de desarrollo de un sistema software.
* Conocer técnicas específicas de interacción persona-ordenador en las diferentes fases del desarrollo del software.
* Comprender los principales métodos, modelos y técnicas aplicables al desarrollo de los sistemas colaborativos.
* Implementar modelos espaciales y temporales para el diseño de bases de datos.
* Saber especificar, diseñar e implementar software gráfico eficiente, haciendo uso de unidades de procesamiento de gráficos programables.
* Hacer una aplicación de escritorio amena y amigable para el usuario.
* Cumplir con los cinco principios de usabilidad.

# Justificación

La calidad del software es una preocupación a la que se dedican muchos esfuerzos. Sin embargo, el software casi nunca es perfecto. Todo proyecto tiene como objetivo producir software de la mejor calidad posible, que cumpla, y si puede supere las expectativas de los usuarios. Debido a esto, con la herramienta que se está desarrollando la empresa tendría una mejor evaluación en cuanto a la Calidad de Software, dicho esto; la empresa se ha visto en la necesidad de examinar este proyecto de desarrollo de software, el cual le sería de gran utilidad para resolver los problemas de evaluación de software en cuanto a calidad y así destacar entre los demás competidores.

Las empresas requieren de una aplicación que permita registrar los diferentes proyectos, consultarlos, registrar medidas en función de las métricas establecidas y aplicar las fórmulas en el momento que se seleccione para un periodo determinado.

Entre las funciones que debe tener la herramienta están:

* Administrar Usuarios: El administrador de la aplicación podrá administrar los usuarios (agregar, modificar, eliminar, buscar, listar, iniciar sesión).
* Administrar Proyectos: El administrador de la aplicación podrá administrar los proyectos (agregar, modificar, eliminar, buscar, listar). Un usuario normal sólo podrá listar y consultar.
* Administrar la recolección de medidas y aplicación de métricas
* Generar reportes y gráficas

Este proyecto propone la creación de una aplicación de estimación de software basada en métricas, es decir, que se adapte a los requerimientos del cliente, y a su vez mejore el rendimiento y velocidad para medir la calidad de un proyecto de desarrollo de software.

Esto repercutirá considerablemente en administración y en la vida laboral del personal, puesto que al acceder a esta aplicación le será de suma importancia para gestionar de manera, rápida y segura todo manejo contable de la empresa.

**Beneficios**

1. Se detectan problemas rápidamente.
2. Se crean y se siguen estándares de trabajo
3. Se verifica que los objetivos individuales vayan acordes con los objetivos de la organización.
4. Se recomiendan métodos para realizar el trabajo.
5. Se evita incurrir en costos innecesarios.
6. Se planea la calidad.

**Beneficiados directos**

* Clientes.
* Personal
* Empresa

**Beneficiados indirectos**

* Personal de calidad.

**Requerimientos funcionales:**

* Registro de usuarios, clientes y proyectos.
* Impresión de reportes y gráficas.

**Requerimientos no funcionales:**

* Validaciones.
* Requerimientos del sistema.
* Seguridad lógica.
* Seguridad de datos.
* Responsiva.
* Diseño de la interfaz.

# Alcance y limitaciones

## Alcances

* Será un software de escritorio orientado a la estimación de software basada en métricas.
* Será de interfaz intuitiva y agradable al usuario.
* La aplicación contendrá diferentes módulos para la administración usuarios, proyectos, reportes y gráficas.
* Los registros de datos serán almacenados de forma persistentes en una base de datos local.
* Tendrá tipos de usuarios.
  + Administrador
  + Líder SQA
  + Miembro SQA

## Limitaciones

* Esta aplicación sólo deberá contener la funcionalidad requerida.
* Solo será una aplicación de escritorio (local).
* El tiempo de desarrollo de la aplicación comprenderá del cuatrimestre enero– abril del presente año.
* La aplicación no permitirá agregar métricas nuevas.

# Marco Teórico

## Calidad

Es el cumplimiento de los requisitos de funcionalidad y desempeño explícitamente establecidos, de los estándares de desarrollo explícitamente documentados y de las características implícitas que se esperan de todo software desarrollado profesionalmente”.

Es el grado en que un conjunto de características cumplen con los requisitos y satisfacción del cliente.

## Medición

Es el proceso mediante el cual se asignan números o símbolos a los atributos de entidades reales para definirlas de acuerdo con reglas claramente establecidas. Es el acto de determinar una medida.

## Medida

Proporciona indicación cuantitativa de la extensión, la cantidad, la dimensión, la capacidad o el tamaño de algún atributo de un producto o proceso.

## Modelos de calidad de software

**Métricas de Chidamber y CF Kemerer**

Consta de un conjunto de 6 métricas de productos específicas para código OO; todas menos una se aplican a las clases y tratan de medir la complejidad, acoplamiento, cohesión, herencia y comunicación inter-clases.

**Modelo MOSCA (Modelo Sistémico de Calidad)**

Modelo que consta de 4 niveles (dimensiones, categorías, características y métricas que contienen un total de 715 métricas.

**Modelo de FURPS**

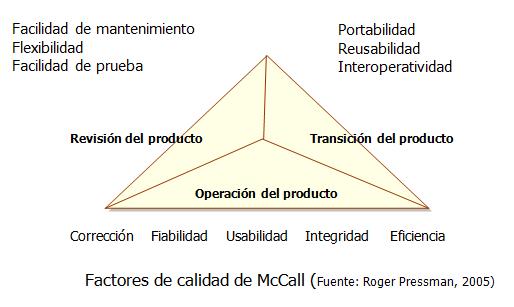
Modelo desarrollado por HP, se utiliza para establecer métricas de calidad para todas las actividades del proceso de desarrollo de un software, inclusive de un sistema de información.

**Modelo de Dromey**

Modelo que resalta el hecho de que la calidad del producto es altamente determinada por los componentes del mismo (incluyendo guías de usuario, diseños y código).

## Factores de calidad de McCall

Los factores que determinan la calidad del software se clasifican en tres grupos: operación del producto (características operativas), revisión del producto (capacidad para experimentar cambios) y transición del producto (capacidad para adaptarse a nuevos entornos).



**Operación del producto**

**Corrección.** El grado en que el programa cumple con su especificación y satisface los objetivos del cliente.

**Fiabilidad.** El grado en que una aplicación lleva a cabo las operaciones especificadas con la precisión requerida.

**Usabilidad.** El esfuerzo necesario para aprender, operar y preparar los datos de entrada de un programa e interpretar la salida.

**Integridad.** Es el grado de control sobre el acceso al software o a los datos a personal no autorizado.

**Eficiencia.** La cantidad de recursos hardware y software para que una aplicación realice las operaciones con los tiempos de respuesta adecuados.

**Revisión del producto**

**Facilidad de mantenimiento.** Es el esfuerzo requerido para localizar y corregir errores en una aplicación.

**Flexibilidad.** El esfuerzo requerido para modificar una aplicación en funcionamiento.

**Facilidad de prueba.** El esfuerzo requerido para probar una aplicación para asegurar que cumpla con lo especificado en los requisitos.

**Transición del producto**

**Portabilidad.** El esfuerzo requerido para transferir la aplicación a otro hardware o sistema operativo.

**Reusabilidad.** El grado en que una aplicación (o partes de ella) puede reutilizarse en otras aplicaciones.

**Interoperabilidad.** El esfuerzo necesario para comunicar la aplicación con otras aplicaciones o sistemas informáticos.

## Métrica

Es la medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo determinado. Un ingeniero de software recopila medidas y desarrolla métricas para obtener los indicadores.

**Ventaja:**

* Ayudan a tomar decisiones más fundamentadas.
* Apoyan a analizar el trabajo desarrollado.
* Permiten conocer si se ha mejorado o no con respecto a proyectos anteriores.
* Facilitan la detección de áreas con problemas para poder remediarlos a tiempo y realizar mejores estimaciones.

## Indicador

* Es una métrica o una combinación de métricas que proporcionan conocimientos acerca del proceso de software, un proyecto de software o el propio producto.
* Permiten al jefe de proyecto o a los ingenieros de software ajustar el proceso, proyecto o el producto para que las cosas mejoren.

## Métricas del proceso

* Las métricas del proceso son recopiladas de los proyectos durante un largo período de tiempo.
* La medición del proceso implica las mediciones de las actividades relacionadas con el software mediante atributos como el esfuerzo, el costo y los defectos encontrados.

## Métricas del proyecto

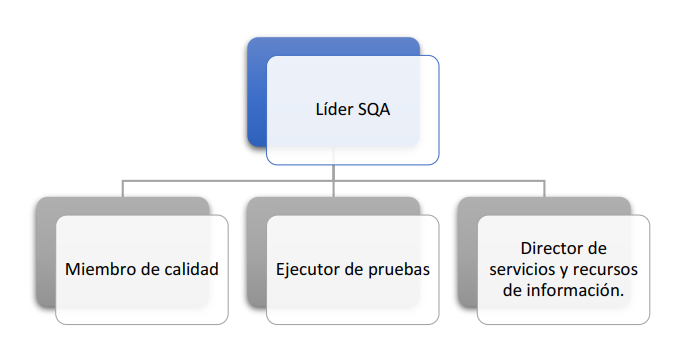
* Es importante definir mediciones que ayuden al mejoramiento del proyecto, ya que éste engloba todos los recursos, actividades y artefactos que se organizan para lograr un producto de software.
* La primera aplicación de métricas de proyectos en la mayoría de los proyectos de software ocurre durante la estimación. Las métricas recopiladas de proyectos anteriores se utilizan como una base desde la que se realizan las estimaciones del esfuerzo y del tiempo para el actual trabajo del software.

## Métricas del producto

* Las métricas del producto se centran en las características del software pero no en cómo fue producido.
* Un producto es el software o sistema funcionando y el conjunto de todos sus artefactos, documentos, modelos, módulos, o componentes que lo conforman.
* Por lo anterior, las métricas del producto deben medir cada uno de éstos.

## Roles y responsabilidades

La estructura organizacional influye y controla la calidad del software, en este caso será de tipo jerárquica para el desarrollo del proyecto. Los roles que se pueden identificar son: líder de SQA, miembro de calidad, ejecutor de pruebas, director de servicios y recursos de información.



**Responsabilidades:**

**Líder SQA:**

* Asegurar que los objetivos de calidad del proyecto se cumplan.
* Planificar las actividades de SQA.
* Obtener los recursos para la ejecución del Plan de SQA.
* Realizar el monitoreo y seguimiento a las actividades planificadas de SQA.

**Miembro de calidad:**

* Construir los activos de proceso que los miembros del equipo del proyecto deben utilizar para desarrollar el software.
* Coordinar y facilitar revisiones.

**Ejecutor de pruebas:**

* Ejecutar las tareas definidas dentro del plan de SQA.
* Realizar revisiones a los productos para verificar su conformidad a los estándares.

**Director de servicios y recursos de información:**

* Proveer y garantizar los recursos para el desempeño del rol de calidad.
* Realizar revisiones periódicas de las actividades y los resultados de SQA.

## Origen de Java

***Sun Microsystems*** desarrolló, en **1991**, el lenguaje de programación orientado a objetos que se conoce como **Java.** El objetivo era utilizarlo en un **set-top box**, un tipo de dispositivo que se encarga de la recepción y la descodificación de la señal televisiva. El primer nombre del **lenguaje** fue **Oak**, luego se conoció como **Green** y finamente adoptó la denominación de **Java.**

La intención de Sun era crear un lenguaje con una estructura y una sintaxis similar a **C y C++,** aunque con un modelo de objetos más simple y eliminando las herramientas de bajo nivel.

Los pilares en los que se sustenta Java son cinco: la programación orientada a objetos, la posibilidad de ejecutar un mismo **programa** en diversos sistemas operativos, la inclusión por defecto de soporte para trabajo en red, la opción de ejecutar el código en sistemas remotos de manera segura y la facilidad de uso.

Lo habitual es que las aplicaciones Java se encuentren compiladas en un **bytecode** (un fichero binario que tiene un programa ejecutable), aunque también pueden estar compiladas en código máquina nativo.

Sun controla las especificaciones y el desarrollo del lenguaje, los compiladores, las máquinas virtuales y las bibliotecas de clases a través del **Java Community Process**. En los últimos años, la empresa (que fue adquirida por [**Oracle**](http://www.oracle.com/)) ha liberado gran parte de las tecnologías Java bajo la licencia **GNU GPL**.

La aplicación de Java es muy amplia. El lenguaje se utiliza en una gran variedad de dispositivos móviles, como teléfonos y pequeños electrodomésticos. Dentro del ámbito de Internet, Java permite desarrollar pequeñas aplicaciones (conocidas con el nombre de applets) que se incrustan en el código HTML de una página, para su directa ejecución desde un navegador; cabe mencionar que es necesario contar con el plug-in adecuado para su funcionamiento, pero la instalación es liviana y sencilla.

Ya que sabemos un poco de los antecedentes y como fue el origen de Java ahora debemos conocer porque es útil y cuáles son las ventajas que tiene java para que las podemos aplicar en nuestro proyecto integrador y algunas de las ventajas son las siguientes:

* *Es muy flexible*: Java es un lenguaje especialmente preparado para la reutilización del código; permite a sus usuarios tomar un programa que hayan desarrollado tiempo atrás y actualizarlo con mucha facilidad, sea que necesiten agregar funciones o adaptarlo a un nuevo entorno.
* *Funciona en cualquier plataforma*: a diferencia de los programas que requieren de versiones específicas para cada sistema operativo (tales como Windows o Mac), las aplicaciones desarrolladas en Java funcionan en cualquier entorno, dado que no es el sistema quien las ejecuta, sino la máquina virtual (conocida como *Java Virtual Machine o JVM*).
* *Su uso no acarrea inversiones económicas*: programar en Java es absolutamente gratis; no es necesario adquirir ninguna licencia, sino simplemente descargar el kit de desarrollo (Java Development Kit o JDK) y dar riendas sueltas a la imaginación.
* *Es de fuente abierta*: Java ofrece el código de casi todas sus librerías nativas para que los desarrolladores puedan conocerlas y estudiarlas en profundidad, o bien ampliar su funcionalidad, beneficiándose a ellos mismos y a los demás.
* *Es un lenguaje expandible*: continuando con el punto anterior, cada programador tiene la libertad de revisar y mejorar el código nativo de Java, y su trabajo puede convertirse en la solución a los problemas de muchas personas en todo el mundo. Infinidad de desarrolladores han aprovechado esta virtud del lenguaje y continúan haciéndolo.  
  Ahora que conocemos el origen y ventajas de java que nos ayudaran para la implementación de nuestro proyecto integrador debemos conocer una parte importante que es la Programación Orientada a Objetos que de igual forma nos servirá

## Programación Orientada a Objetos

La programación orientada a objetos (POO), es una forma de programar la cual utiliza objetos para diseñar y programar aplicaciones. Usa diferentes técnicas como son la herencia, abstracción, polimorfismo y encapsulamiento y además trabaja con diferentes lenguajes de programación.

Los principios de la POO son:

-Modularidad: Reducir el código en unos más pequeños, trabajar por partes

-Reusabilidad: Es decir darle varios usos a algo

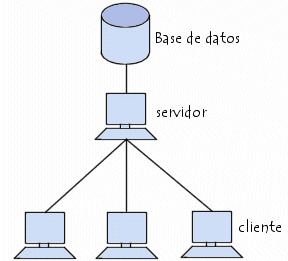
En nuestro proyecto integrador trabajaremos con la programación orientada a objetos, tomaremos parte del problema “componentes”, determinándolos, analizándolos y estudiándolos para luego abordarlos y programarlos con éxito.

El propósito de la programación orientada a objetos es dar solución a un problema específico, diciéndolo en partecitas y analizando cada una de ellas para llegar satisfactoriamente a la solución.

Concluimos que ya que sabemos que es la programación orientada a objetos (POO) ahora si podemos utilizarla por consiguiente nos ayudara para implementar nuestro proyecto en marcha y usaremos diferentes técnicas para poder realizar nuestros programas.

## Base de Datos (Hidalgo, s.f.)

Una **base de datos** (cuya abreviatura es BD) es una entidad en la cual se pueden almacenar datos de manera estructurada, con la menor redundancia posible. Diferentes programas y diferentes usuarios deben poder utilizar estos datos. Por lo tanto, el concepto de base de datos generalmente está relacionado con el de [red](http://es.ccm.net/contents/initiation/concept.php3), ya que se debe poder compartir esta información. Generalmente se habla de un "Sistema de información" para designar a la estructura global que incluye todos los mecanismos para compartir datos.

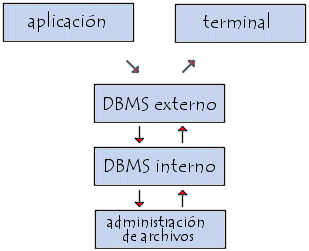


**¿Por qué utilizar una base de datos?**

Una base de datos proporciona a los usuarios el acceso a datos, que pueden visualizar, ingresar o actualizar, en concordancia con los derechos de acceso que se les hayan otorgado. Se convierte más útil a medida que la cantidad de datos almacenados crece. Una base de datos puede ser local, es decir que puede utilizarla solo un usuario en un equipo, o puede ser distribuida, es decir que la información se almacena en equipos remotos y se puede acceder a ella a través de una red. La principal ventaja de utilizar bases de datos es que múltiples usuarios pueden acceder a ellas al mismo tiempo. 

**Administración de bases de datos**

Rápidamente surgió la necesidad de contar con un sistema de administración para controlar tanto los datos como los usuarios. La administración de bases de datos se realiza con un Sistema de Gestión de Bases de Datos (SGBD) también llamado **DBMS** (*Database Management System*). El DBMS es un conjunto de servicios (aplicaciones de *software*) que permite a los distintos usuarios un fácil acceso a la información y proporciona las herramientas para la manipulación de los datos encontrados en la base (insertar, eliminar, editar).



El DBMS puede dividirse en tres subsistemas: el sistema de administración de archivos, cuya función es almacenar la información en un medio físico; el DBMS interno, que sirve para ubicar la información en orden; y el DBMS externo, que representa a la interfaz de usuario.

## JasperReports (EcuRed, s.f.)

Es una biblioteca de creación de [informes](https://es.wikipedia.org/wiki/Informe) que tiene la habilidad de entregar contenido enriquecido al [monitor](https://es.wikipedia.org/wiki/Monitor_de_computadora), a la [impresora](https://es.wikipedia.org/wiki/Impresora) o a ficheros [PDF](https://es.wikipedia.org/wiki/PDF), [HTML](https://es.wikipedia.org/wiki/HTML), [XLS](https://es.wikipedia.org/wiki/XLS), [CSV](https://es.wikipedia.org/wiki/CSV) y [XML](https://es.wikipedia.org/wiki/XML).

Está escrito completamente en [Java](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_Java) y puede ser usado en gran variedad de aplicaciones de [Java](https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)), incluyendo [J2EE](https://es.wikipedia.org/wiki/J2EE) o [aplicaciones web](https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_web), para generar contenido dinámico. Se ha desarrollado un subproyecto que es un servidor integrado para informes: [JasperReports Server](http://community.jaspersoft.com/project/jasperreports-server).

Su propósito principal es ayudar a crear documentos de tipo páginas, preparados para imprimir en una forma simple y flexible.

JasperReports se usa comúnmente con [iReport](https://es.wikipedia.org/w/index.php?title=IReport&action=edit&redlink=1), un *[front-end](https://es.wikipedia.org/wiki/Front-end_y_back-end" \o "Front-end y back-end)* gráfico [de código abierto](https://es.wikipedia.org/wiki/Software_de_c%C3%B3digo_abierto) para la edición de informes, si bien a partir de la versión 5.5.0 iReport ha sido sustituido por [Jaspersoft Studio](http://community.jaspersoft.com/project/jaspersoft-studio), un [*front-end*](https://es.wikipedia.org/wiki/Front-end_y_back-end) gráfico [de código abierto](https://es.wikipedia.org/wiki/Software_de_c%C3%B3digo_abierto) basado en Eclipse.

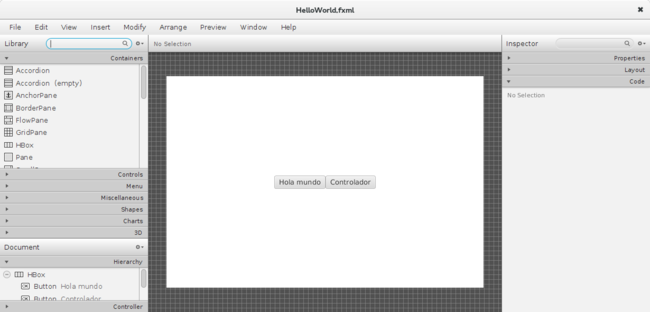
Se encuentra bajo licencia libre [GNU](https://es.wikipedia.org/wiki/GNU_Lesser_General_Public_License), por lo que es [Software libre](https://es.wikipedia.org/wiki/Software_libre). Forma parte de la iniciativa apilada [open source](https://es.wikipedia.org/wiki/Open_source) [Lisog](https://es.wikipedia.org/w/index.php?title=Lisog&action=edit&redlink=1" \o "Lisog (aún no redactado)).

## JavaFX Scene Builder (Blog Bitix, 2015)

JavaFX Scene Builder, editor para crear archivos FXML. Con la aplicación JavaFX Scene Builder podemos construir la interfaz gráfica de una aplicación de escritorio Java de forma más sencilla. JavaFX Scene Builder genera archivos descriptores FXML que podemos cargar en la aplicación evitando la tediosa y no sencilla tarea de construir la interfaz gráfica mediante código. En el artículo comento la aplicación JavaFX Scene Builder, como usar los archivos FXML en una aplicación de escritorio, como asociar manejadores de eventos y como cambiar las propiedades de los controles.

Por lo poco que he visto JavaFX es una gran evolución sobre AWT y Swing para desarrollar aplicaciones de escritorio en Java aportando varias mejoras. En el artículo [Introducción a JavaFX](https://picodotdev.github.io/blog-bitix/2015/10/introduccion-a-javafx-aplicaciones-de-escritorio-en-java/) comentaba que una de las cosas que me parecía más complicada usando tanto AWT y más tarde usando Swing era la construcción de los elementos visuales mediante código. El código para construir las ventanas, paneles, rejillas, botones, etiquetas, etc. era extenso y difícil de comprender la composición leyéndolo. Con JavaFX se ha introducido una nueva herramienta visual llamada [JavaFX Scene Builder](http://www.oracle.com/technetwork/java/javase/downloads/javafxscenebuilder-info-2157684.html) que permite generar un archivo en formato FXML (declarativo en XML) que contiene la descripción de las ventanas o como llama JavaFX escenas. Este descriptor es similar a la forma de construir interfaces gráficas en la plataforma de Microsoft con los archivos [XAML](https://msdn.microsoft.com/en-us/library/cc295302.aspx).

Con los archivos FXML que genera la aplicación JavaFX Scene Builder crear aplicaciones gráficas es mucho más sencillo y más fácilmente mantenible. Proporciona un editor que sigue el principio lo que ves es lo que obtienes (WYSIWYG) y que permite generar los archivos FXML que posteriormente se pueden utilizar en la aplicación Java de escritorio para crear la interfaz visual. Podemos [descargar la aplicación JavaFX Scene Builder](http://www.oracle.com/technetwork/java/javase/downloads/javafxscenebuilder-1x-archive-2199384.html) desde su página de descargas. En la siguiente imagen se puede ver la interfaz con el ejemplo Hola Mundo.

[](https://picodotdev.github.io/blog-bitix/assets/images/posts/2015/102/javafx-scene-builder.png)JavaFX Scene Builder con la aplicación ¡Hola Mundo!

Y el FXML que genera del mismo ejemplo.

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.effect.\*?>

<?import javafx.scene.control.\*?>

<?import java.lang.\*?>

<?import javafx.scene.layout.\*?>

<HBox alignment="CENTER" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1" fx:controller="io.github.picodotdev.javafx.HelloWorldFXMLController">

<children>

<Button fx:id="button" mnemonicParsing="false" text="Hola mundo" />

<Button fx:id="controllerButton" mnemonicParsing="false" onAction="#onAction" text="Controlador" />

</children>

</HBox>

Que podemos usar en una aplicación con el siguiente código. Comparándolo con el [código del ejemplo sin FXML](https://github.com/picodotdev/blog-ejemplos/blob/master/HolaMundoJavaFX/src/main/java/io/github/picodotdev/javafx/HelloWorld.java) vemos que el número de líneas necesarias son unas cuantas menos que se notará más cuanto más compleja sea la interfaz gráfica.

package io.github.picodotdev.javafx;

import java.io.IOException;

import javafx.application.Application;

import javafx.fxml.FXMLLoader;

import javafx.scene.Scene;

import javafx.scene.layout.HBox;

import javafx.stage.Stage;

public class HelloWorldFXML extends Application {

public static void main(String[] args) {

launch(args);

}

@Override

public void start(Stage stage) {

try {

HBox layout = (HBox) FXMLLoader.load(getClass().getResource("HelloWorld.fxml"));

stage.setTitle("¡Hola JavaFX!");

stage.setScene(new Scene(layout, 300, 250));

stage.show();

} catch (IOException e) {

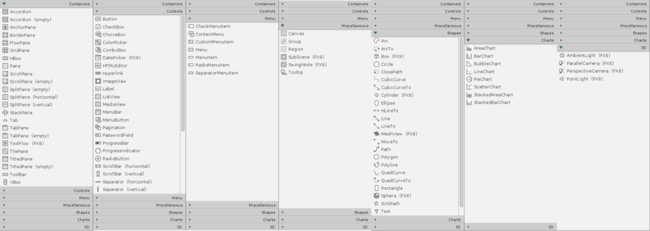
e.printStackTrace();

}

}

}

En la aplicación JavaFX Scene Builder disponemos de una amplia paleta de controles que podemos usar arrastrando y soltando para construir la interfaz, botones, checkbox, radio buttons, paneles, rejillas, menús, contenedores, miscelánea, formas, 3D, …

[](https://picodotdev.github.io/blog-bitix/assets/images/posts/2015/102/controles-javafx-scene-builder.png)Controles de JavaFX

Si con los archivos FXML construimos la interfaz queda como enlazar los elementos visuales con su comportamiento mediante eventos. Una forma es inyectar en propiedades con una anotación los elementos visuales y añadirles el código de comportamiento, otra es indicar en el propio archivo FXML la clase controlador que se encargará del manejo de los eventos del evento visual y otra es incluir el código manejador de evento en el propio archivo FXML.

package io.github.picodotdev.javafx;

import java.net.URL;

import java.util.ResourceBundle;

import javafx.event.ActionEvent;

import javafx.event.EventHandler;

import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.Button;

public class HelloWorldFXMLController implements Initializable {

@FXML

private Button button;

@FXML

private Button controllerButton;

@FXML

void onAction(ActionEvent event) {

System.out.println("Hello World! (controllerButton)");

}

@Override

public void initialize(URL location, ResourceBundle resources) {

button.setOnAction(new EventHandler<ActionEvent>() {

@Override

public void handle(ActionEvent event) {

System.out.println("Hello World! (button)");

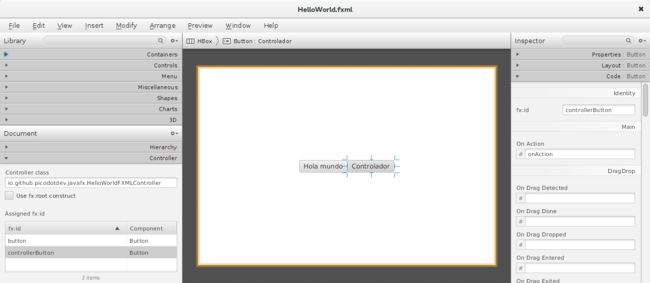
}

});

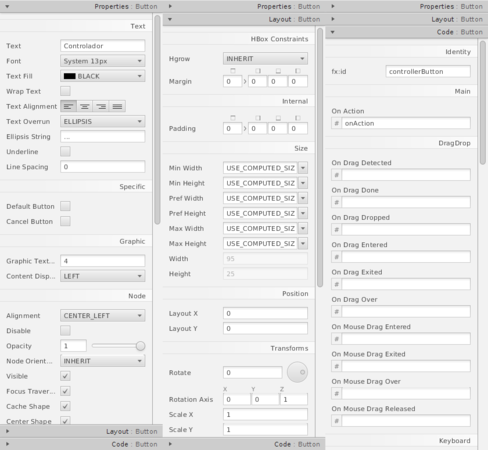
}

}

La clase manejador de eventos se indica en el panel Controller y es una clase con métodos que posteriormente se indican en el editor con una almohadilla (#) delante en el panel Code. Por ejemplo, si el manejador de un botón se llama onClicken la clase controlador HelloWorldFXMLController deberemos tener ese método con el parámetro adecuado en este caso un [ActionEvent](https://docs.oracle.com/javase/8/javafx/api/javafx/event/ActionEvent.html).

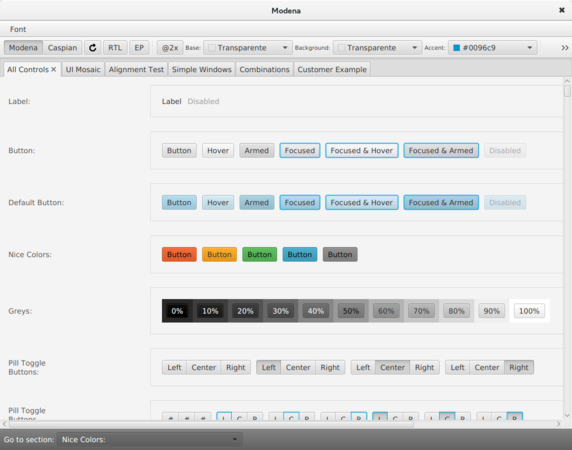
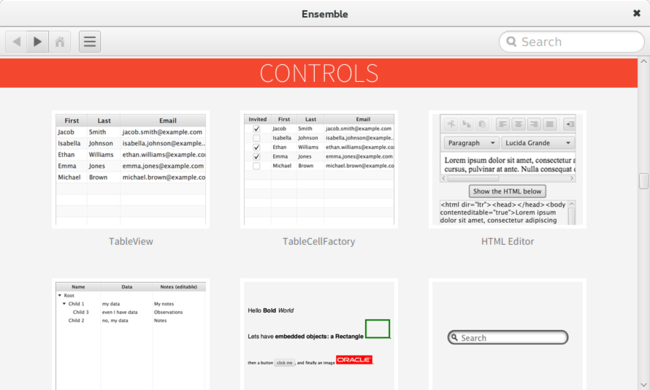
[](https://picodotdev.github.io/blog-bitix/assets/images/posts/2015/102/javafx-scene-builder-controller.png)Añadiendo comportamiento a aplicación JavaFX

Además de poder enlazar los componente visuales con el código para añadirles funcionalidad se pueden modificar las propiedades visuales como el texto, fuente y tamaño, alineación, opacidad, visibilidad, altura, anchura, margen, margen interior, rotación, escalado. Algunas propiedades son aplicables a varios componentes otras son específicas según el componente.

[](https://picodotdev.github.io/blog-bitix/assets/images/posts/2015/102/propiedades-javafx-scene-builder.png)

Propiedades de los controles JavaFX, en este caso de un botón

Entre las [demostraciones y ejemplos de Java 8](http://www.oracle.com/technetwork/java/javase/downloads/index.html) está la aplicación Modena con la que podemos ver el aspecto visual de los componentes, en la aplicación Ensemble hay más ejemplos junto con código de muchos controles.

[](https://picodotdev.github.io/blog-bitix/assets/images/posts/2015/102/modena.png)Aplicación Modena[](https://picodotdev.github.io/blog-bitix/assets/images/posts/2015/102/ensemble.png)Aplicación Ensemble

En la propia página web de JavaFX, en el libro [JavaFX For Dummies](http://www.amazon.es/gp/product/1118385349/ref=as_li_ss_tl?ie=UTF8&camp=3626&creative=24822&creativeASIN=1118385349&linkCode=as2&tag=blobit-21) y otra documentación se puede encontrar explicado de forma más detallada JavaFX.

En definitiva con la aplicación JavaFX Scene Builder y los archivos FXML construir la interfaces gráficas de una aplicación de escritorio Java es bastante más simple que lo era antiguamente. JavaFX además ofrece posibilidades que antes no eran posibles en Swing y AWT como rotar componentes, un componente visualizador de HTML, soporte para aceleración gráfica por hardware y más cosas. Después de haber conocido en algo más detalle JavaFX me quedo con una buena sensación. Como muestra de la flexibilidad de cambio de los estilos de los controles el proyecto [JFoenix](http://www.jfoenix.com/) les aplica el Material Desing de las aplicaciones [Android](https://www.android.com/) de Google.

## PosgreSQL (Ubuntu, s.f.)

**Características**

Algunas de sus principales características son, entre otras:

* **Alta concurrencia**

Mediante un sistema denominado MVCC (Acceso concurrente multiversión, por sus siglas en inglés) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo *commit*. Esta estrategia es superior al uso de bloqueos por tabla o por [filas](https://es.wikipedia.org/wiki/Fila_(base_de_datos)) común en otras bases, eliminando la necesidad del uso de bloqueos explícitos.

* **Amplia variedad de tipos nativos**

PostgreSQL provee nativamente soporte para:

* + Números de [precisión arbitraria](https://es.wikipedia.org/wiki/Precisi%C3%B3n_arbitraria).
  + Texto de largo ilimitado.
  + Figuras geométricas (con una variedad de funciones asociadas).
  + [Direcciones IP](https://es.wikipedia.org/wiki/Direcci%C3%B3n_IP) (IPv4 e IPv6).
  + Bloques de direcciones estilo [CIDR](https://es.wikipedia.org/wiki/CIDR).
  + [Direcciones MAC](https://es.wikipedia.org/wiki/Direcci%C3%B3n_MAC).
  + [Arrays](https://es.wikipedia.org/wiki/Vector_(programaci%C3%B3n)).

Adicionalmente los usuarios pueden crear sus propios tipos de datos, los que pueden ser por completo indexables gracias a la infraestructura GiST de PostgreSQL. Algunos ejemplos son los tipos de datos [GIS](https://es.wikipedia.org/wiki/GIS) creados por el proyecto [PostGIS](https://es.wikipedia.org/wiki/PostGIS" \o "PostGIS).

* **Otras características**
  + [Claves ajenas](https://es.wikipedia.org/wiki/Clave_ajena) también denominadas Llaves ajenas o Claves Foráneas (*foreign keys*).
  + [Disparadores](https://es.wikipedia.org/wiki/Disparador_(base_de_datos)) (*triggers*): Un disparador o *trigger* se define como una acción específica que se realiza de acuerdo a un evento, cuando éste ocurra dentro de la base de datos. En PostgreSQL esto significa la ejecución de un procedimiento almacenado basado en una determinada acción sobre una tabla específica. Ahora todos los disparadores se definen por seis características:
    - El nombre del disparador o *trigger*
    - El momento en que el disparador debe arrancar
    - El evento del disparador deberá activarse.
    - La tabla donde el disparador se activará
    - La frecuencia de la ejecución
    - La función que podría ser llamada
    - La función no es correcta

Entonces combinando estas seis características, PostgreSQL le permitirá crear una amplia funcionalidad a través de su sistema de activación de disparadores (*triggers*).

* [**Vistas**](https://es.wikipedia.org/wiki/Vista_(base_de_datos))**.**
  + Integridad transaccional.
  + Herencia de tablas.
  + Tipos de datos y operaciones geométricas.

Soporte para transacciones distribuidas. Permite a PostgreSQL integrarse en un sistema distribuido formado por varios recursos (p.ej, una base de datos PostgreSQL, otra Oracle, una cola de mensajes IBM MQ JMS y un ERP SAP) gestionado por un servidor de aplicaciones donde el éxito ("commit") de la transacción global es el resultado del éxito de las transacciones locales.

* **Funciones**
  + Bloques de código que se ejecutan en el servidor. Pueden ser escritos en varios lenguajes, con la potencia que cada uno de ellos da, desde las operaciones básicas de programación, tales como bifurcaciones y bucles, hasta las complejidades de la programación orientada a objetos o la programación funcional.
  + Los disparadores (*triggers* en inglés) son funciones enlazadas a operaciones sobre los datos.
  + Algunos de los lenguajes que se pueden usar son los siguientes:
    - Un lenguaje propio llamado [PL/PgSQL](https://es.wikipedia.org/wiki/PL/PgSQL) (similar al [PL/SQL](https://es.wikipedia.org/wiki/PL/SQL) de oracle).
    - [C](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_C).
    - [C++](https://es.wikipedia.org/wiki/C%2B%2B).
    - [Java](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_Java) [PL/Java web](http://web.archive.org/web/http:/gborg.postgresql.org/project/pljava/projdisplay.php).
    - [PL/Perl](https://es.wikipedia.org/wiki/Perl).
    - [plPHP](http://web.archive.org/web/http:/plphp.commandprompt.com/).
    - [PL/Python](http://www.postgresql.org/docs/current/interactive/plpython.html).
    - [PL/Ruby](http://raa.ruby-lang.org/project/pl-ruby).
    - [PL/sh](http://plsh.projects.postgresql.org/).
    - [PL/Tcl](http://www.postgresql.org/docs/current/interactive/pltcl.html).
    - [PL/Scheme](http://plscheme.projects.postgresql.org/).
    - Lenguaje para aplicaciones estadísticas [R](https://es.wikipedia.org/wiki/Lenguaje_R) por medio de [PL/R](http://www.joeconway.com/plr/).

PostgreSQL soporta funciones que retornan "filas", donde la salida puede tratarse como un conjunto de valores que pueden ser tratados igual a una fila retornada por una consulta (query en inglés).

Las funciones pueden ser definidas para ejecutarse con los derechos del usuario ejecutor o con los derechos de un usuario previamente definido. El concepto de funciones, en otros DBMS, son muchas veces referidas como "procedimientos almacenados" (stored procedures en inglés).

* **Ventajas**
  + Seguridad en términos generales
  + Integridad en BD: restricciones en el dominio
  + Integridad referencial
  + Afirmaciones (Assertions)
  + Disparadores (Triggers)
  + Autorizaciones
  + Conexión a DBMS
  + Transacciones y respaldos

# Bibliografía

(s.f.). Obtenido de http://ingsoftwarejefer.webcindario.com/unidad-3/estrategias-de-prueba-del-software/metricas-tecnicas-del-software.html

(s.f.). Obtenido de EcuRed: https://www.ecured.cu/JasperReport

*Blog Bitix*. (9 de OCTUBRE de 2015). Obtenido de https://picodotdev.github.io/blog-bitix/2015/10/javafx-scene-builder-editor-para-crear-archivos-fxml/

*Educando*. (s.f.). Obtenido de http://www.educando.edu.do/articulos/directivo/evaluacin-de-los-resultados/

Hidalgo, U. A. (s.f.). Obtenido de https://www.uaeh.edu.mx/docencia/P\_Lectura/Ciudad%20Sahagun/asignatura/introduccion.pdf

*Ingeniería de Software*. (s.f.). Obtenido de http://ingenieriadesoftware.mex.tl/52788\_Rup-Agil.html

León, J. (20017). *SisTel*. Obtenido de SisTel: https://www.sistel.es/equipos-desarrollo-software

Ubuntu. (s.f.). *Ubuntu*. Obtenido de http://www.guia-ubuntu.com/index.php?title=PgAdmin\_III