[CAPITULO I - INTRODUCCIÓN 1](#_Toc431546794)

[1.1 Antecedentes 1](#_Toc431546795)

[1.2 Análisis del Problema 1](#_Toc431546796)

[1.2.1 Definición del Problema 1](#_Toc431546797)

[1.3 Objetivo del Proyecto 2](#_Toc431546798)

[1.3.1 Objetivo General 2](#_Toc431546799)

[1.3.2 Objetivos Específicos 2](#_Toc431546800)

[1.4 Áreas Involucradas 2](#_Toc431546801)

[1.5 Justificación 2](#_Toc431546802)

[1.6 Alcance 3](#_Toc431546803)

[CAPITULO II - MARCO TEORICO 4](#_Toc431546804)

[2.1 Sistema Adminstrativo 4](#_Toc431546805)

[2.1.1 Definicion 4](#_Toc431546806)

[2.1.2 Sistema web 4](#_Toc431546807)

[2.1.3 Cloud Computing 4](#_Toc431546808)

[2.2 Metodologia del Proyecto 4](#_Toc431546809)

[2.2.1 Metodologia Scrum 4](#_Toc431546810)

[2.2.1.1 Definicion 4](#_Toc431546811)

[2.2.1.2 Caracteristicas 5](#_Toc431546812)

[2.2.1.3 Justificacion 5](#_Toc431546813)

[2.3 Plataforma de desarrollo 5](#_Toc431546814)

[2.3.1 Lenguaje de Programación Javascript 5](#_Toc431546815)

[2.3.1.1 Definición 5](#_Toc431546816)

[2.3.1.2 Caracteristicas 6](#_Toc431546817)

[2.3.1.3 Justificacion 6](#_Toc431546818)

[2.3.2 Framework Meteor 6](#_Toc431546819)

[2.3.2.1 Definicion 6](#_Toc431546820)

[2.3.2.2 Características 7](#_Toc431546821)

[2.3.2.3 Justificación 7](#_Toc431546822)

[2.3.3 Framework Semantic UI 8](#_Toc431546823)

[2.3.3.1 Definicion 8](#_Toc431546824)

[2.3.3.2 Caracteristicas 8](#_Toc431546825)

[2.3.3.3 Justificación 8](#_Toc431546826)

[2.4 Administrador de Base de Datos 8](#_Toc431546827)

[2.4.1 MongoDB 8](#_Toc431546828)

[2.4.1.1 Definición 8](#_Toc431546829)

[2.4.1.2 Terminología básica de MongoDB 9](#_Toc431546830)

[2.4.1.3 Arquitectura de MongoDB 9](#_Toc431546831)

[2.4.1.4 Características 9](#_Toc431546832)

[2.4.1.5 Justificación 10](#_Toc431546833)

# CAPITULO II - MARCO TEORICO

Este capítulo está dedicado a realizar una descripción completa de los elementos conceptuales que se usan en el proyecto. Define las características principales y justificación de cada elemento.

## 2.1 Sistema Adminstrativo

### 2.1.1 Definicion

Definimos a los sistemas administrativos como la red de procedimientos relacionados de acuerdo a un esquema integrado tendientes al logro de los finesde una organización. Esto quiere decir que un conjunto de procedimientos relacionados y dentro de determinadas condiciones, constituyen un sistema.

### 2.1.2 Sistema web

En la ingeniería de software se denomina aplicación web a aquellas herramientas que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores web en la que se confía la ejecución al navegador.

### 2.1.3 Cloud Computing

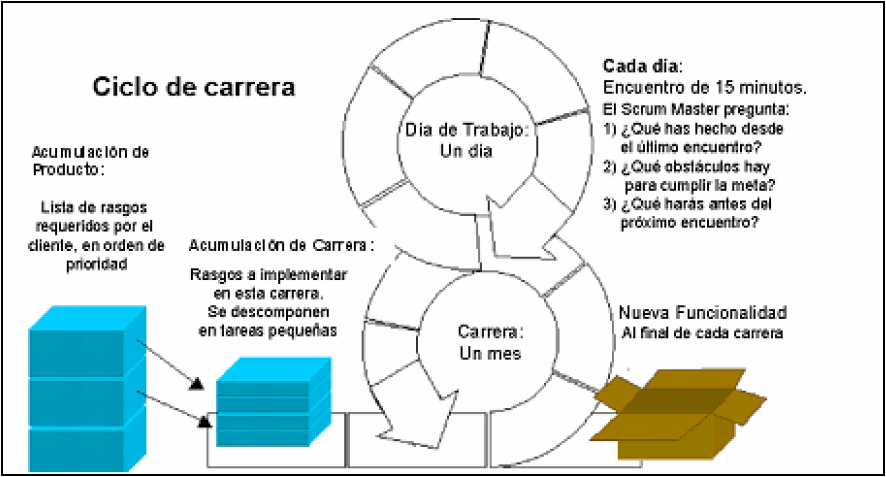
La computación en la nube, concepto conocido también bajo los términos servicios en la nube, informática en la nube, nube de cómputo o nube de conceptos, del inglés cloud computing, es un paradigma que permite ofrecer servicios de computación a través de Internet.

## 2.2 Metodologia del Proyecto

### 2.2.1 Metodologia Scrum

#### 2.2.1.1 Definicion

Scrum define un proceso empírico, iterativo e incremental de desarrollo que intenta obtener ventajas respecto a los procesos definidos (cascada, espiral, prototipos, etc.) mediante la aceptación de la naturaleza caótica del desarrollo de software, y la utilización de prácticas tendientes a manejar la impredictibilidad y el riesgo a niveles aceptables. El mismo surge e n 1 9 8 6 , de un artículo de la Harvard Business Review titulado “The New New Product Development Game” de Hirotaka Takeuchi e Ikujiro Nonaka, que introducía las mejores prácticas más utilizadas en 10 compañías japonesas altamente innovadoras. A partir de ahí y tomando referencias al juego de rugby, Ken Scwaber y Jeff Sutherland formalizan el proceso conocido como Scrum en el año 1995.



*Figura 1: Metodologia scrum*

El marco técnico de scrum está formado por:

Roles:

* El equipo scrum.
* El dueño del producto.
* El Scrum Master.

Artefactos:

* Pila del producto.
* Pila del sprint.
* Incremento.
* Sprint.

Eventos

* Reunión de planificación del sprint.
* Scrum diario.
* Revisión del sprint.
* Retrospectiva del sprint.

En donde lo mas importante es el sprint, que dura aproximadamente entre 2 a 4 semanas.

#### 2.2.1.2 Caracteristicas

* Enfatiza valores y prácticas de gestión, sin pronunciarse sobre requerimientos, prácticas de desarrollo, implementación y demás cuestiones técnicas
* Hace uso de Equipos auto-dirigidos y auto-organizados
* Puede ser aplicado teóricamente a cualquier contexto en donde un grupo de gente necesita trabajar junta para lograr una meta común.[2]
* Desarrollo de software iterativos incrementales basados en prácticas agiles
* Iteraciones de treinta días; aunque se pueden realizar con mas frecuencia, estas iteraciones, conocidas como Sprint
* Dentro de cada Sprint se denomina el Scrum Master al Líder de Proyecto quien llevará a cabo la gestión de la iteración
* Se convocan diariamente un “Scrum Daily Meeting” el cual representa una reunión de avance diaria de no más de 15 minutos con el propósito de tener realimentación sobre las tareas de los recursos y los obstáculos que se presentan. En la cual se responden preguntas como: ¿Qué has hecho desde el ultimo encunetro? ¿Qué obstaculos hay para cumplir la meta? ¿Qué haras antes del proximo encuentro?

#### 2.2.1.3 Justificacion

* + Capacidad de respuesta a cambios de requisitos a lo largo del desarrollo.
  + Entrega continua y en plazos breves de software funcional.
  + Trabajo conjunto entre el cliente y el equipo de desarrollo.
  + Importancia de la simplicidad, eliminado el trabajo innecesario.
  + Atención continua a la excelencia técnica y al buen diseño.
  + Mejora continua de los procesos y el equipo de desarrollo.

## 2.3 Plataforma de desarrollo

### 2.3.1 Lenguaje de Programación Javascript

#### 2.3.1.1 Definición

JavaScript(a veces abreviado como JS) es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web, pero también usado en muchos entornos sin navegador, tales como node.js o Apache CouchDB. Es un lenguaje script multi-paradigma, basado en prototipos, dinámico, soporta estilos de programación funcional, orientada a objetos e imperativa. Leer más sobre JavaScript.

JavaScript no debe ser confundido con el lenguaje de programación Java. Java es una marca registrada de Oracle en Estados Unidos y otros países.

Contrariamente a la falsa idea popular, JavaScript no es "Java interpretativo". En pocas palabras, JavaScript es un lenguaje de programación dinámico que soporta construcción de objetos basado en prototipos. La sintaxis básica es similar a Java y C++ con la intención de reducir el número de nuevos conceptos necesarios para aprender el lenguaje.[6]

#### 2.3.1.2 Caracteristicas

* Imperativo y estructurado.
* Tipado dinámico.
* Puede funcionar como lenguaje procedimental y como orientado a objetos.
* Evaluación del codigo en tiempo de ejecución.
* Funcional.
* Basado en prototipos.

#### 2.3.1.3 Justificacion

* Es el lenguaje de programación de los navegadores web (todos los más importantes lo soportan y lo tienen activado por defecto: Firefox, Chrome, IE, Opera, Safari…), lo que lo convierten en el lenguaje más popular en Internet.
* Hay una auténtica competición entre los navegadores para optimizar sus motores y dar mejor soporte a JavaScript y que su código se ejecute más rápidamente. JavaScript por tanto es cada vez más estable y tiene mejor rendimiento.
* Es muy potente y expresivo, con sintaxis que guarda similitudes con otros lenguajes muy populares, pero con características particulares.
* Puede interaccionar con otras tecnologías populares como Flash, Java, PHP, etc.
* Existen miles de bibliotecas para trabajar con javascript

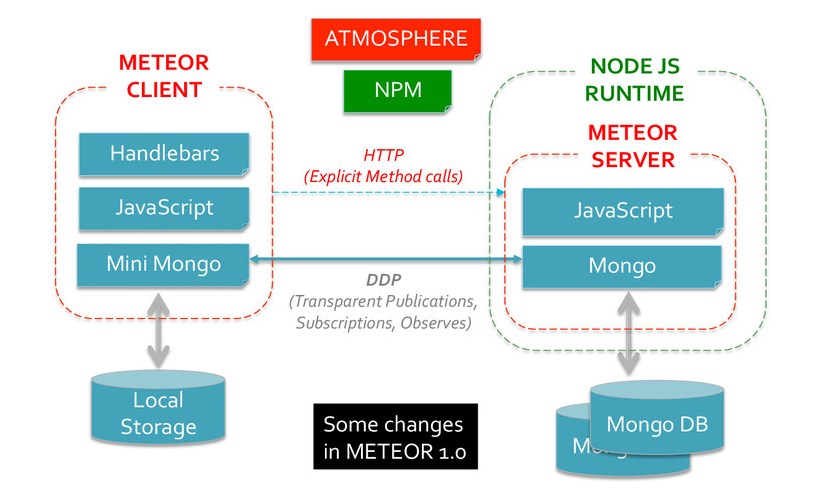
### 2.3.2 Framework Meteor

#### 2.3.2.1 Definicion

Meteor es un framework para desarrollo de aplicaciones web, el cual utiliza Node.js en su núcleo, permitiéndonos de esta manera programar en un mismo lenguaje (JavaScript) tanto del lado del cliente como del servidor. Lo que es más, Meteor también es capaz de compartir código entre ambos entornos. Entre sus principales características esta el poder crear aplicaciones "Realtime" de una manera realmente rápida y sencilla.

Meteor se encuentra entre la base de datos de la aplicación y la interfaz de usuario y se asegura de que ambos se mantengan en sincronía.[5]

La arquitectura que usa Meteor es la siguiente:



*Figura 2: Arquietectura de Meteor*

#### 2.3.2.2 Características

* **Transferencia de datos**. Meteor no envía HTML sobre la red. El servidor envía los datos y permite al cliente visualizarlo.
* **Un solo lenguaje**. Meteor permite escribir tanto el cliente como el servidor de tu aplicación en JavaScript.
* **Base de datos en todas partes**. Puedes usar los mismos métodos para acceder a la base de datos tanto desde el cliente, como desde el servidor.
* **Compensación de latencia**. En el cliente, Meteor presolicita los datos y simula modelos para hacer parecer que la llamada al servidor es instantánea.
* **Reactividad en Full Stack**. En Meteor, el tiempo real es por defecto. Todas las capas de la aplicación, desde la base de datos hasta la plantilla, se actualizan de forma automática cuando es necesario.
* **Adopta el ecosistema**. Meteor es código abierto, se integra con herramientas y frameworks de código abierto.
* **La simplicidad es igual a la productividad**. La mejor manera de hacer que algo parezca simple es hacerlo simple. La principal funcionalidad de Meteor es clara, crear bellas APIs.

#### 2.3.2.3 Justificación

* **Buena base tecnológica.** La parte de servidor de Meteor funciona sobre Node.js pero eliminando problemas como el famoso “callback hell” utilizando [Fibers](https://www.discovermeteor.com/blog/understanding-sync-async-javascript-node/) y actualmente utiliza MongoDB como base de datos utilizando [oplogtailing](http://blog.mongolab.com/2014/07/tutorial-scaling-meteor-with-mongodb-oplog-tailing/) para monitorizar los cambios de forma eficiente.
* **Facil de aprender** . Javascript en todas partes. Solo necesitas aprender un lenguaje.
* **Inspeccion de codigo.** En el lado del cliente, es posible utilizar cualquiera de las herramientas que tenemos a nuestor alcance ofrecidas por los distintos navegadores. Meteor tiene soporte para [debugging en](https://github.com/meteor/meteor/blob/devel/History.md#v094) el lado del servidor  y [WebStorm](http://blog.jetbrains.com/webstorm/2014/09/meteor-support-in-webstorm-9/) ha lanzado hace poco una versión totalmente compatible con Meteor, debugging incluido.
* **Simplicidad en el cliente.** A diferencia de otros frameworks como Angular, Meteor no mezcla la lógica de al aplicación con el HTML. El sistema utilizado por Meteor, [Blaze](https://www.meteor.com/blaze) no incluye en el HTML javascript,lógica de bindings ni atributos extraños como data- o ng-. No es necesario inventar tus propias etiquetas HTML.
* **Seguridad.** Meteor tiene medidas de seguridad de serie frente a muchos de los problemas de seguridad más frecuentes.
* **Pruebas.** El framework oficial de Testing de Meteor es [Velocity](http://velocity.meteor.com/) y nos permite integrar las funcionalidades propias del tiempo real con Jasmine o Mocha. Los tests se ejecutan automáticamente cuando guardamos los cambios y el estado de nuestros tests se indica en directo con un punto verde o rojo en la parte superior derecha de la aplicación.
* **Soporte Movil.** Las aplicaciones móviles se pueden generar usando el mismo código que la versión web usando PhoneGap/Cordova.

### 2.3.3 Framework Semantic UI

#### 2.3.3.1 Definicion

Semantic UI es un framework de aplicaciones front end que es compuesto por un conjunto de especificaciones para el intercambio de elementos de la interfaz entre los desarrolladores.

El vocabulario utilizado (para las clases e ID) es mucho más claro en comparación con las alternativas que reduce la curva de aprendizaje.

Semantic UI cuenta con muchos estilos para los elementos HTML, los elementos de la interfaz de usuario y escenarios (botones, formularios, tablas, acordeón, navegación, comentando, etc.).[4]

#### 2.3.3.2 Caracteristicas

* **Html conciso**. Las clases utilizan la sintaxis de lenguaje natural como sustantivo / relaciones modificadoras, orden de las palabras, y la pluralidad de vincular conceptos intuitivamente.
* **Javascript Intuitivo.** Semantic UI utiliza frases simples llamados comportamientos que desencadenan funcionalidad.
* **Diseño Responsive**.Semantic UI está diseñado completamente con medidas “em” haciendo sensible dimensionar componentes.
* **Socios.** Semantic UI tiene integraciones con React, Angular, Meteor, Ember y muchos otros Frameworks para ayudar a organizar la interfaz de usuario.

#### 2.3.3.3 Justificación

* Semantic cuenta con más de 3000 variables CSS.
* Compuesta por 50 definiciones de elementos UI prediseñados que puedes utilizar para aplicar en el diseño de una página o aplicación.
* Es un proyecto Open Source en el cual están involucrados múltiples diseñadores y desarrolladores quienes han puesto un granito de arena para hacer posible este proyecto que actualmente cuenta con más de 3800 Commits.
* Semantic UI al ser un proyecto Open Source es completamente gratuito.
* Semantic UI esta diseñado para ser completamente responsive.
* Cuenta con una buena documentacion.

## 2.4 Administrador de Base de Datos

### 2.4.1 MongoDB

#### 2.4.1.1 Definición

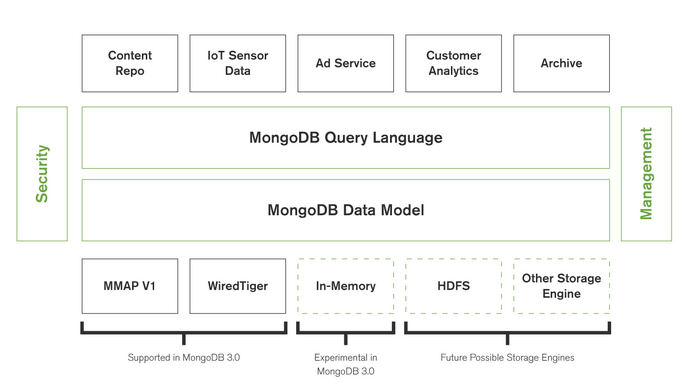
MongoDB es un sistema de base de datos **multiplataforma** orientado a **documentos**, de esquema libre. Como ya os expliqué, esto significa que cada entrada o registro puede tener un esquema de datos diferente, con atributos o “columnas” que no tienen por qué repetirse de un registro a otro. Está escrito en C++, lo que le confiere cierta cercanía al bare metal, o recursos de hardware de la máquina, de modo que es bastante rápido a la hora de ejecutar sus tareas. Además, está licenciado como GNU AGPL 3.0, de modo que se trata de un software de licencia libre. Funciona en sistemas operativos Windows, Linux, OS X y Solaris.

Los distintos documentos se almacenan en formato **BSON**, o Binary JSON, que es una versión modificada de JSON que permite búsquedas rápidas de datos. BSON guarda de forma explícita las longitudes de los campos, los índices de los vectores, y demás información útil para el escaneo de datos. Es por esto que, en algunos casos, el mismo documento en BSON ocupa un poco más de espacio de lo que ocuparía de estar almacenado directamente en formato JSON. Pero una de las ideas claves en los sistemas NoSQL es que el almacenamiento es barato, y es mejor aprovecharlo si así se introduce un considerable incremento en la velocidad de localización de información dentro de un documento.[3]

#### 2.4.1.2 Terminología básica de MongoDB

En MongoDB, cada registro o conjunto de datos se denomina **documento**. Los documentos se pueden agrupar en **colecciones**, las cuales se podría decir que son el equivalente a las tablas en una base de datos relacional (sólo que las colecciones pueden almacenar documentos con muy diferentes formatos, en lugar de estar sometidos a un esquema fijo). Se pueden crear **índices** para algunos atributos de los documentos, de modo que MongoDB mantendrá una estructura interna eficiente para el acceso a la información por los contenidos de estos atributos.

#### 2.4.1.3 Arquitectura de MongoDB



*Figura 3: Arquitectura de Mongo DB*

#### 2.4.1.4 Características

* **Modelo de datos flexible.** Bases de datos NoSQL surgió para hacer frente a los requisitos de los datos vemos que domina las aplicaciones modernas. Un modelo de datos flexible hace que sea fácil de almacenar y combinar los datos de cualquier estructura, y permita la dinamización del esquema sin tiempo de inactividad.
* **La escalabilidad elástica.** Bases de datos NoSQL fueron todas construidas con un enfoque en la escalabilidad, por lo que todos ellos incluyen algún tipo de sharding o partición, permitiendo que la base de datos a escala de salida en el hardware de los productos básicos, lo que permite un crecimiento casi ilimitado.
* **Alto Rendimiento.** Bases de datos NoSQL están diseñados para ofrecer un gran rendimiento, medido en términos de rendimiento y latencia en cualquier escala.

#### 2.4.1.5 Justificación

* Da respuesta a la necesidad de almacenamiento de todo tipo de datos: estructurados, semi estructurados y no estructurados.
* Tiene un gran rendimiento en cuanto a escalabilidad y procesado de la información.
* Puede procesar la gran cantidad de información que se genera hoy en día (millones de usuarios utilizando las mismas aplicaciones, redes sociales que crecen sin parar, internet de las cosas, cloud computing, etc).
* Se adapta a las necesidades actuales de las aplicaciones (millones de usuarios, miles de peticiones por segundo, etc).
* Permite a las empresas ser más ágiles y crecer más rápidamente, crear nuevos tipos de aplicaciones/productos, mejorar la experiencia del cliente y reducir el tiempo de manufacturado o desarrollo del producto, reduciendo así los costes.
* Está orientada a documentos. Lo que quiere decir que en un único documento es capaz de almacenar toda la información necesaria que define un producto, un cliente, etc, aceptando todo tipo de datos (incluidos arrays y otros subdocumentos). Todo ello sin tener que seguir un esquema predefinido.
* Permite adaptar el esquema de la base de datos a las necesidades de la aplicación rápidamente, disminuyendo el tiempo y coste de la puesta en producción de la misma. Esto es así porque permite modificar el esquema desde el código de la aplicación, sin tener que realizar labores de administración de la base de datos como estaríamos obligados a hacer con las bases de datos relacionales.
* Da a los desarrolladores todas las funcionalidades que tienen las bases de datos relacionales (como índices sobre campos secundarios, un completo lenguaje para realizar las consultas, etc).

# Bibliografía

[1] Jordi Torres Viñals. (2011). Empresas en la nube. Ventajas y retos del cloud computing. Barcelona: Libros de cabecera S.L..

[2] Jonathan Rasmusson. (2010). The Agile Samurai. Dallas, Texas: The Pragmatic Bookshelf.

[3] MongoDB Architecture. (2015, Octubre 05). Recuperado de <https://www.mongodb.com/mongodb-architecture>.

[4] Semantic UI User Interface is the language of the web. (2015, Octubre 05). Recuperado de <http://semantic-ui.com/>.

[5] Meteor Documentation. (2015, Octubre 05). Recuperado de <http://docs.meteor.com/>.

[6] JavaScript. (2015, Octubre 05). Recuperado de <https://developer.mozilla.org/es/docs/Web/JavaScript>.