**Documento de Arquitectura de Software****Sistema de Transporte Urbano 1.0**

**Historial de Revisiones**

|  |  |  |  |
| --- | --- | --- | --- |
| ***Fecha*** | ***Versión*** | ***Descripción*** | ***Autor*** |
| 17/11/2015 | 1.0 | Arquitectura de la aplicación | Roberto Cuadros |
| 18/11/2015 | 1.1 | Arquitectura del servidor | Aaron Castillo |

# Arquitectura del Servidor

# Vista de Datos

# captura2.JPG

Fig. 1. Diagrama de Datos

# Vista de despliegue

El diagrama de despliegue esta considerado dentro de un IAAS , que soporta nuestras aplicaciones , llamese Amazon Web Services , los cual nos permite preocuparnos solo de la lógica de nuestra aplicación , como se puede ver el componente exterior es un Cliente móvil que se conecta mediante protocolo HTTPS a nuestro servidor y este mismo se comunica con una base de datos , cada uno dentro de su grupo de dominio.

diagrama-GSM.png

Fig. 2. Diagrama de Despliegue

# Vista de Componentes



Fig. 3. Diagrama de Componentes

Los componentes básicamente están separados en modulos , los cuales consisten en: Modulo de usuarios , encargado de manejar toda la lógica que se refiere a la gestión de los usuarios (creación,búsqueda de usuarios , etc ) , Modulo de Vehiculos , encargado de manejar la lógica de gestión de vehículos (obtener información según placa, comentar un vehiculo , hacer una denuncia , etc) ,y el modulo de acceso a datos ,donde las tablas de la base de datos se conectan a entidades dentro de la aplicación.

* 1. **Tecnologías usadas en el lado del servidor**

Detallamos las tecnologías con las que está construida el servidor y también las usadas por los desarrolladores en el proceso.

* **Nodejs**

Es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor (pero no limitándose a ello) basado en el lenguaje de programación ECMAScript, asíncrono, con I/O de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como por ejemplo, servidores web.Fue creado por Ryan Dahl en 2009 y su evolución está apadrinada por la empresa Joyent, que además tiene contratado a Dahl en plantilla.

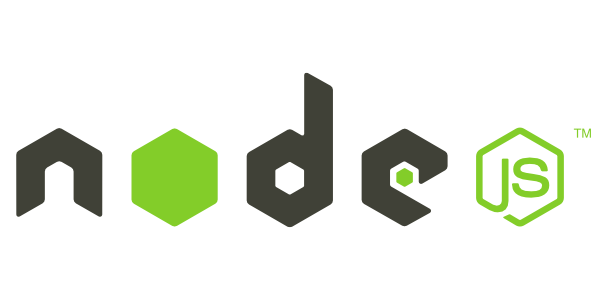


Fig. 4. Node Js

* **Expressjs**

Es un framework del lado del servidor para aplicación web basados en  Node.js, diseñado para construir single-page ,multipage y aplicaciones web híbridas.]Es el framework estándar para la plataforma Nodejs. El autor original, TJ Holowaychuk, lo describe como un framework del servidor inspirado en Sinatra ( otro framework web usado en los entornos basados en el lenguaje de programación Ruby), lo que significa que es relativamente mínimo con muchas características disponibles como plugins.



Fig. 5. Express Js

* **Sequelize Js**

Sequelize es un ORM basado en promises para Node.js y io.js. Es compatible con los dialectos de PostgreSQL, MySQL, MariaDB, SQLite y MSSQL y cuenta con soporte de transacciones sólida, las relaciones, leer la replicación y más.



Fig. 6. Sequelize Js

# Arquitectura de la Aplicación

Aplicación movil desarrollada con tecnologías web usando el framework Ionic para el release multiplataforma.

## Diagrama de la arquitectura

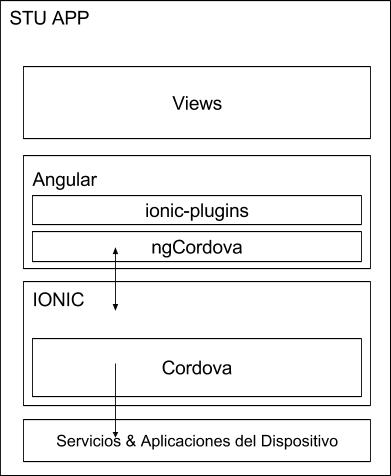


Fig. 7. Diagrama de la aplicación

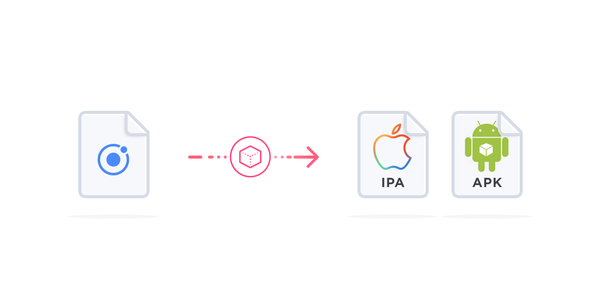


Fig. 8. Proceso de tranformación en app con Inonic

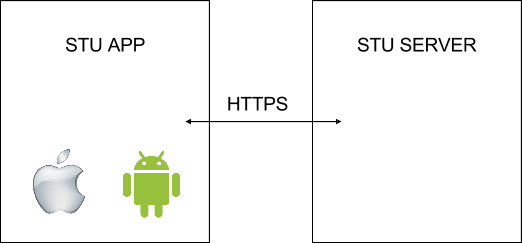


Fig. 9. Relación de modulos

## Tecnologías usadas

Detallamos las tecnologías con las que está construida la aplicación y también las usadas por los desarrolladores en el proceso.

* **Ionic Framework**

Gratuito y de código abierto, Ionic ofrece una biblioteca de HTML optimizado para móviles, CSS y componentes JS, los gestores y herramientas para la construcción de aplicaciones altamente interactivas. Construido con Sass y optimizado para angularjs.



Fig. 10. Ionic framework

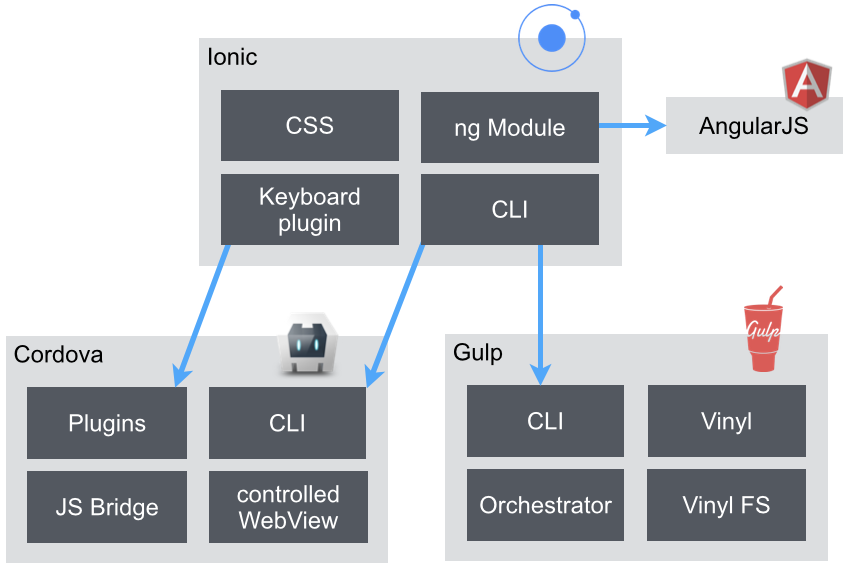


Fig. 11. Arquitectura de Ionic

* **Apache Cordova**

Es un framework de desarrollo móvil de código abierto. Permite el uso de las tecnologías web estándar, como HTML5, CSS3 y JavaScript para el desarrollo multiplataforma, evitando el desarrollo del lenguaje nativo de cada plataformas móviles. Las aplicaciones se ejecutan dentro de envoltorios específicos para cada plataforma, y ​​se basan en estándares compatibles con fijaciones API para acceder a los sensores, los datos y el estado de la red de cada dispositivo.



Fig. 12. Apache Cordova

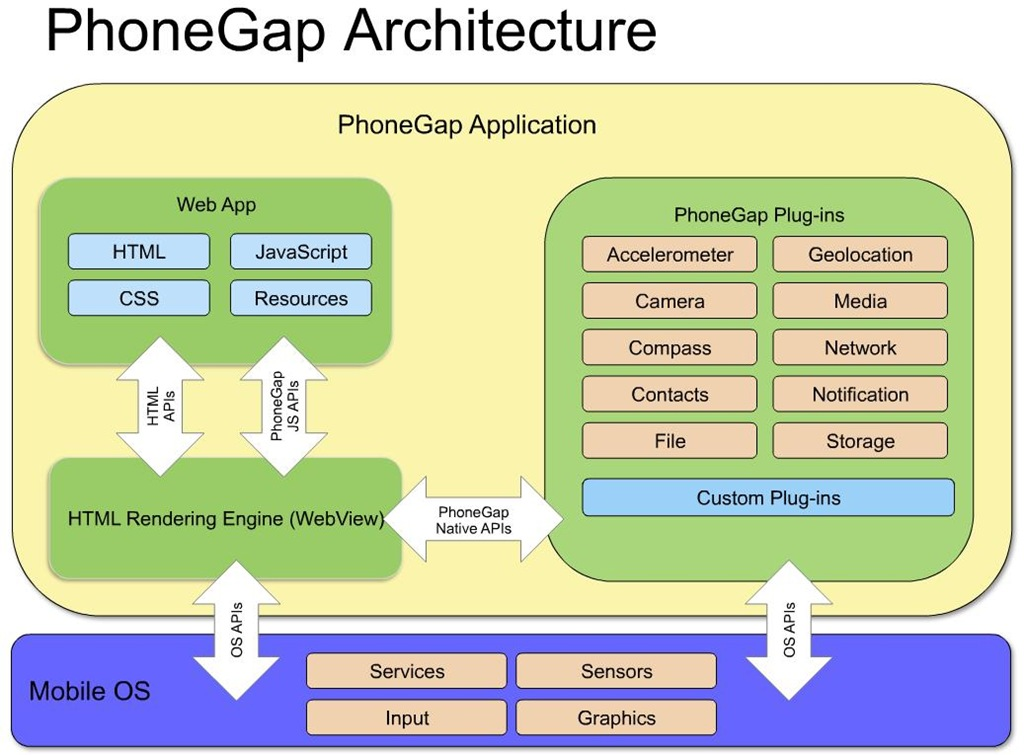


Fig. 13. Arquitectura de Cordova

* **Angular JS**

Angularjs es un conjunto de herramientas para construir el framework más adecuado para el desarrollo de su aplicación. Es completamente extensible y funciona bien con otras bibliotecas. Permite ampliar el vocabulario HTML para su aplicación. El ambiente resultante es extraordinariamente expresiva, legible y rápida de desarrollar.



Fig. 14. Angular Js

* **ngCordova**

Es una colección de más de 70 extensiones angularjs en la parte superior del API de Córdoba que hacen que sea fácil de construir, probar y desplegar aplicaciones móviles Córdova con angularjs.



Fig. 15. Plugin ngCordova

* **Html + Css + Js**

Las tecnologías web usadas para la construcción del app por el framework de Ionic.



Fig. 16. Html + Css + Js

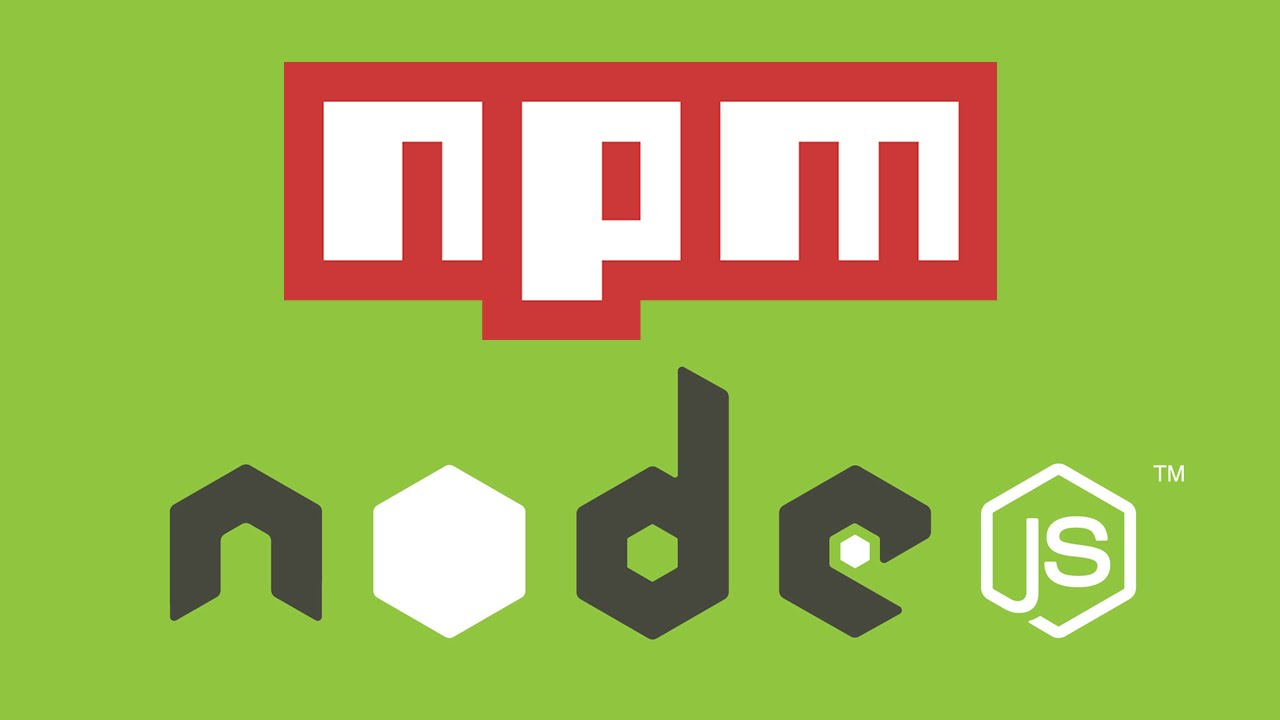
* **Gulp**

Gulp.js es un build system (sistema de construcción) que permite automatizar tareas comunes de desarrollo, tales como la minificación de código JavaScript, recarga del navegador, compresión de imágenes, validación de sintaxis de código y un sin fin de tareas más.

 Fig. 17. Gulp Js

* **Nodejs + Npm**

Npm es el package manager para muchas aplicaciones como Gulp, Bower, Angular, Cordova.

 Fig. 18. Node Js + Npm

* **Bower**

Bower está optimizado para el front-end. Bower utiliza un árbol de dependencias plana, requiriendo solamente una versión para cada paquete, reduciendo de carga de página a un mínimo.

 Fig. 19. Bower