# Capítulo 6 – Stack MEAN

## **Introducción**

En éste capítulo, se analizará el stack MEAN y sus componentes. El mismo está compuesto por un conjunto de tecnologías respetando el acrónimo como sigue: Mongo, Express, Angular y Node. Además, otros complementos, como Compodoc (documentador), bibliotecas y framework aplicados a la vista como Bootstrap y JQuery.

## **6.1 ¿Qué es MEAN?**

“Se denomina MEAN, o MEAN *stack*, a un conjunto de capas de software para el desarrollo de aplicaciones, dónde la característica predominante es el uso del lenguaje de programación popularizado como JavaScript”.



Ilustración 31 - Acrónimo MEAN

## **6.2 Componentes de MEAN**

### 6.2.1 MongoDB

Es un sistema de base de datos NoSQL, es decir, que almacena los datos en estructuras o “documentos”, los cuales están estructurados como objetos bajo la notación JSON (Notación simple de objeto tipo JavaScript), en vez de filas y agrupdos en colecciones en contraposición a las tablas de un RDBMS. Debido a la ausencia de comprobación de integridad referencial tine un alto desempeño.

### 6.2.2 Express

Es un paquete de NodeJS que ofrece una interface mínima para manejo de solicitudes o peticiones HTTP. Ofrece también un sistema simple de enrutamiento (Routing). Dentro del MEAN *stack* opera del lado del servidor, también conicido como *back-end*.

### 6.2.3 Angular

Es un framework orientado a crear aplicaciones web, basado en el sub-lenguaje TypeScript (JavaScript con verificación de tipos de dato *ahead of time*), mantenido por Google, enfocado en aplicaciones web de una sola página o SPA. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC), con el objetivo de que el desarrollo y las pruebas sean más sencillas.

### 6.2.4 Node

En MEAN, Node, es la plataforma encargada del funcionamiento del servidor. Se trata de un intérprete de JavaScript enfocado en la programación del lado del servior.

Utiliza el motor de ejecución de JavaScript de Google, denominado V8, y presenta una arquitectura orientada a eventos, en conjunto con una serie de APIs no-bloqueantes (asíncronas[[1]](#footnote-1)) que le proporcionan un rendimiento y una escalabilidad muy elevadas. Esta característica se debe a una librería en C, multiplataforma, que proporciona soporte de E/S asíncronas basada en bucles de eventos, se encuentra diseñada específicamente para el uso en NodeJS[[2]](#footnote-2) y es llamada LibUV (Unicornio Velocirraptors).

Ilustración 392 - Logo del motor V8

Si bien Node se puede utilizar para crear cualquier tipo de aplicación, dado que incorpora un módulo para poder actuar como un servidor web dentro de su biblioteca standard, es especialmente popular para crear aplicaciones web.

Actualmente lo emplean para sus aplicaciones multitud de empresas de todos los ámbitos, pero especialmente de Internet. Desde aplicaciones web, aplicaciones en línea de comandos, scripts para administración de sistemas, aplicaciones de red, etc. Es aconsejable su utilización en aplicaciones como: chats, APIs REST, entrada de datos concurrentes y en espera (escritura en DB bloqueante), procesamiento parcial de archivos, transmisión de datos, proxy, aplicaciones como corredores de bolsa (tiempo real), visualización de interacciones, etc.

Node es utilizado en la construcción rápida y escalable de aplicaciones de red, debido a que es capaz de manejar un gran número de conexiones simultáneas con alto rendimiento, lo que equivale a una alta escalabilidad. La idea principal de Node.js es el uso no-bloqueante, event-driven I/O, permanecer ligero y eficiente en la superficie del uso intensivo de datos en tiempo real de las aplicaciones que se ejecutan en dispositivos distribuidos (Como podemos apreciar en la comparativa entre los servidores tradicionales y Node Ilustración 40 Comparativa de servidores tradicionales y Nodejs).

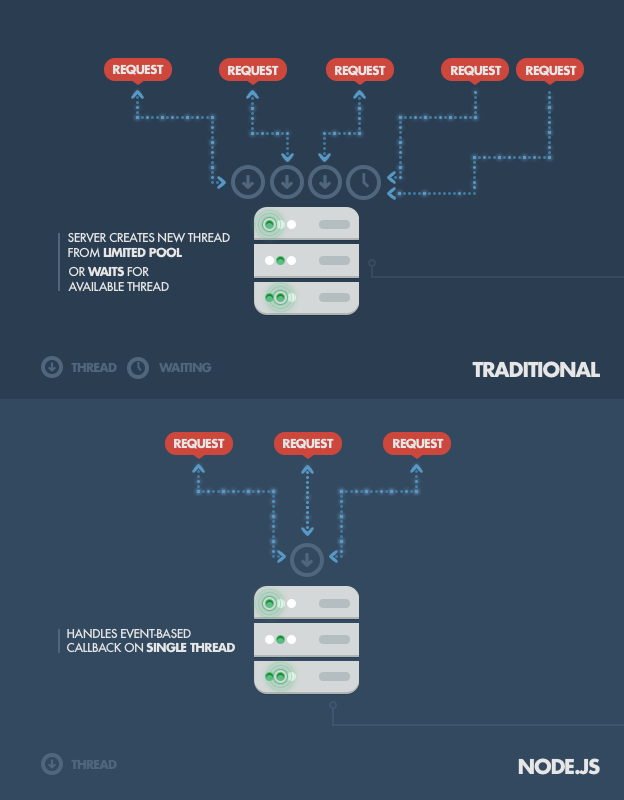


Ilustración 40 Comparativa de servidores tradicionales y Nodejs

## **6.3 Otros complementos**

### 6.3.1 Twitter Bootstrap

Es uno de los frameworks más populares que integra HTML, CSS, y JS para el desarrollo de proyectos adaptables a diferentes medios de reproducción (Responsive). Falta

### 6.3.2 Compodoc

Documentador JavaScript estático.

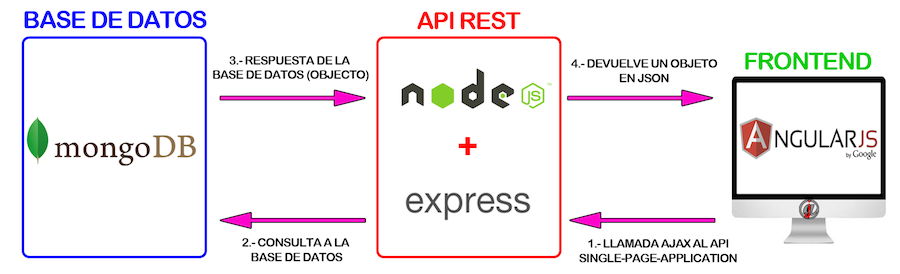


Ilustración 33 - Arquitectura de interacción MEAN

### 6.3.3 JSON

Es el acrónimo de JavaScript Simple Object Notation. Se trata de un mecanismo de *marshaling*, que permite transmitir en formato de cadenas de texto objetos (o estructuras complejas) que pueden ser luego des-*marshalizadas* para recuperar los objetos origniales.

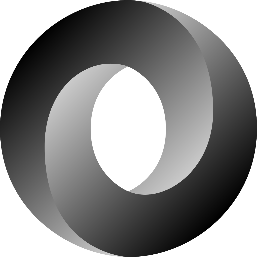
Una de las supuestas ventajas de JSON sobre XML como formato de intercambio de datos es que es mucho más sencillo escribir un analizador sintáctico (parser) de JSON. En JavaScript, un texto JSON se puede analizar fácilmente usando la función JSON.parse(), lo cual ha sido fundamental para que JSON haya sido aceptado por parte de la comunidad de desarrolladores AJAX, debido a la ubicuidad de JavaScript en casi cualquier navegador web.

Ilustración 414 - Logo de JSON

Podemos decir que en MEAN, JSON es el formato de transferencia de datos entre todas las capas: navegador, servidor web y servidor de datos”

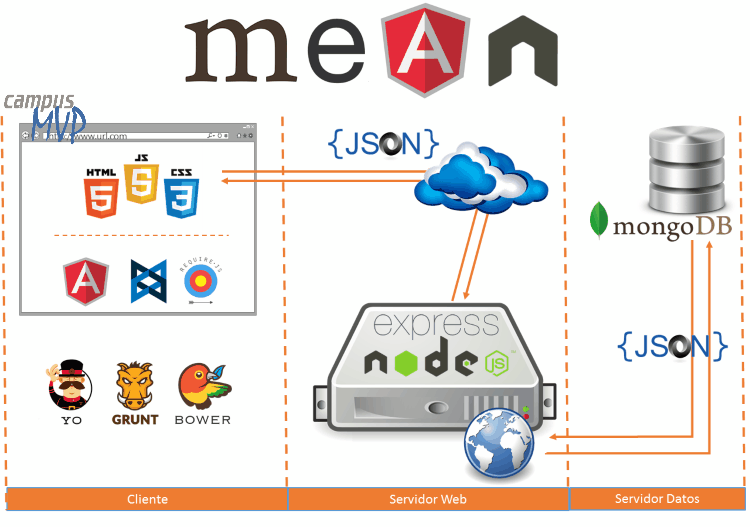


Ilustración 35 - Json pegamento de tecnologías

1. https://en.wikipedia.org/wiki/Libuv [↑](#footnote-ref-1)
2. https://github.com/libuv/libuv/wiki/Projects-that-use-libuv [↑](#footnote-ref-2)