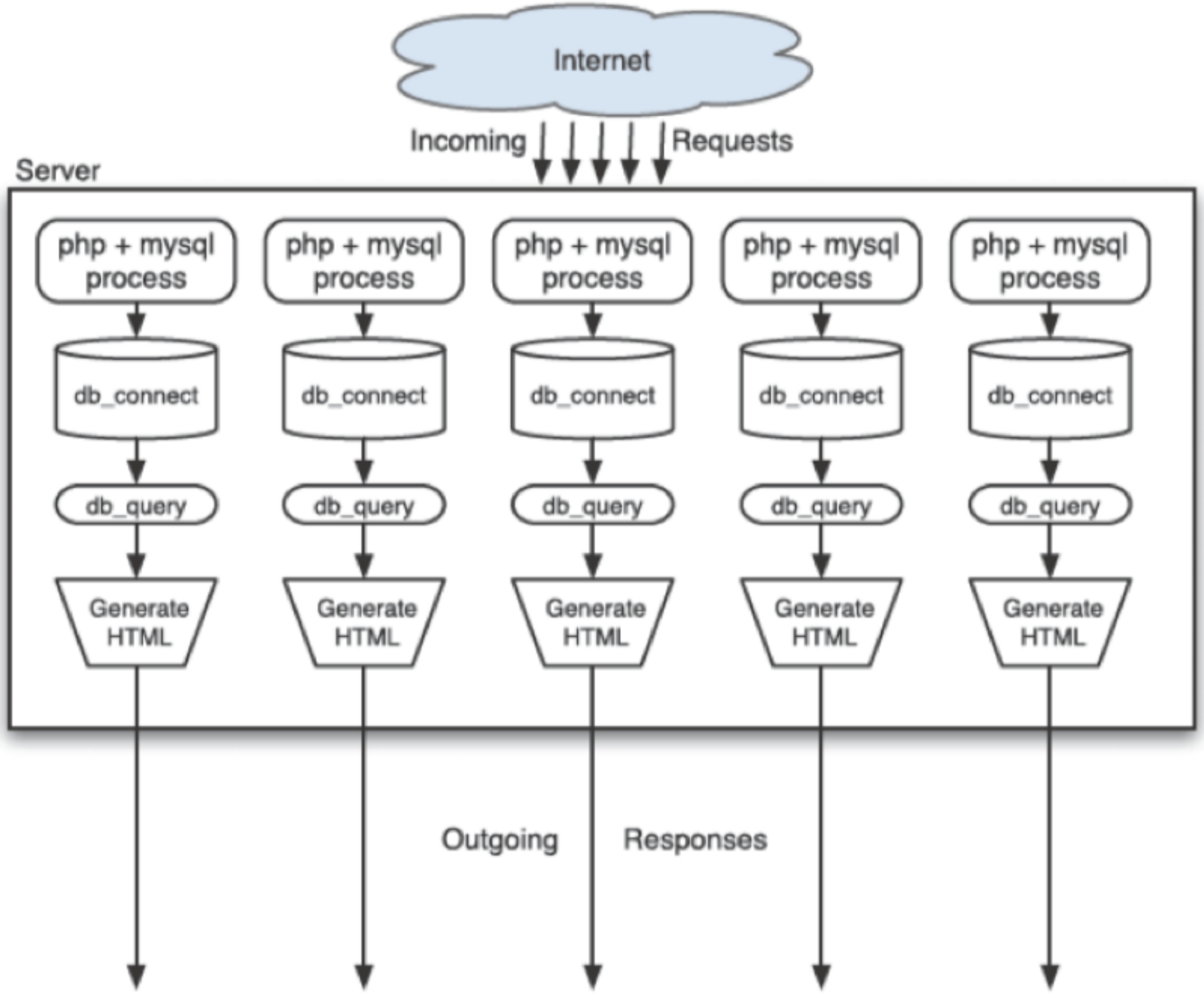
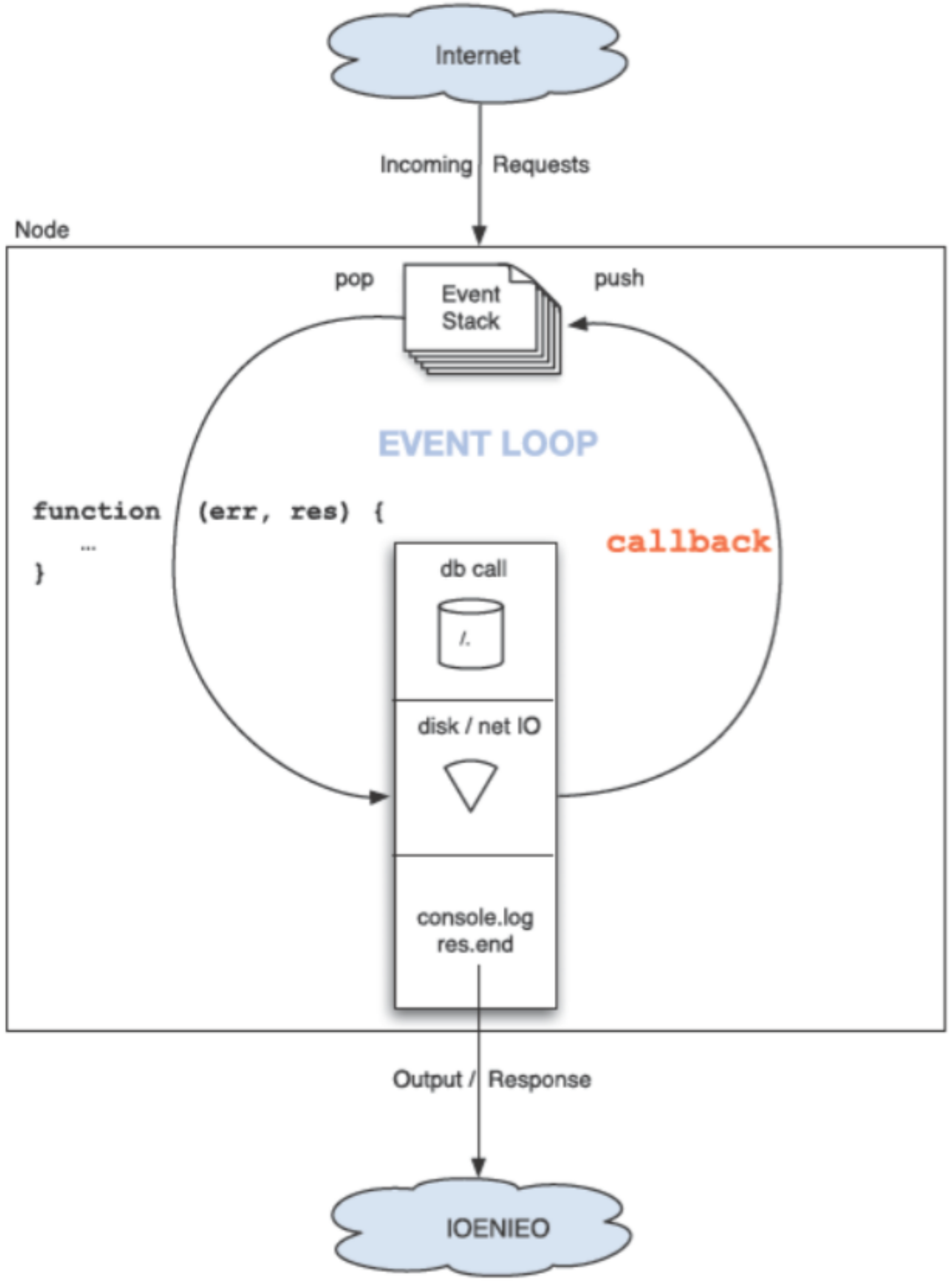
Node Components

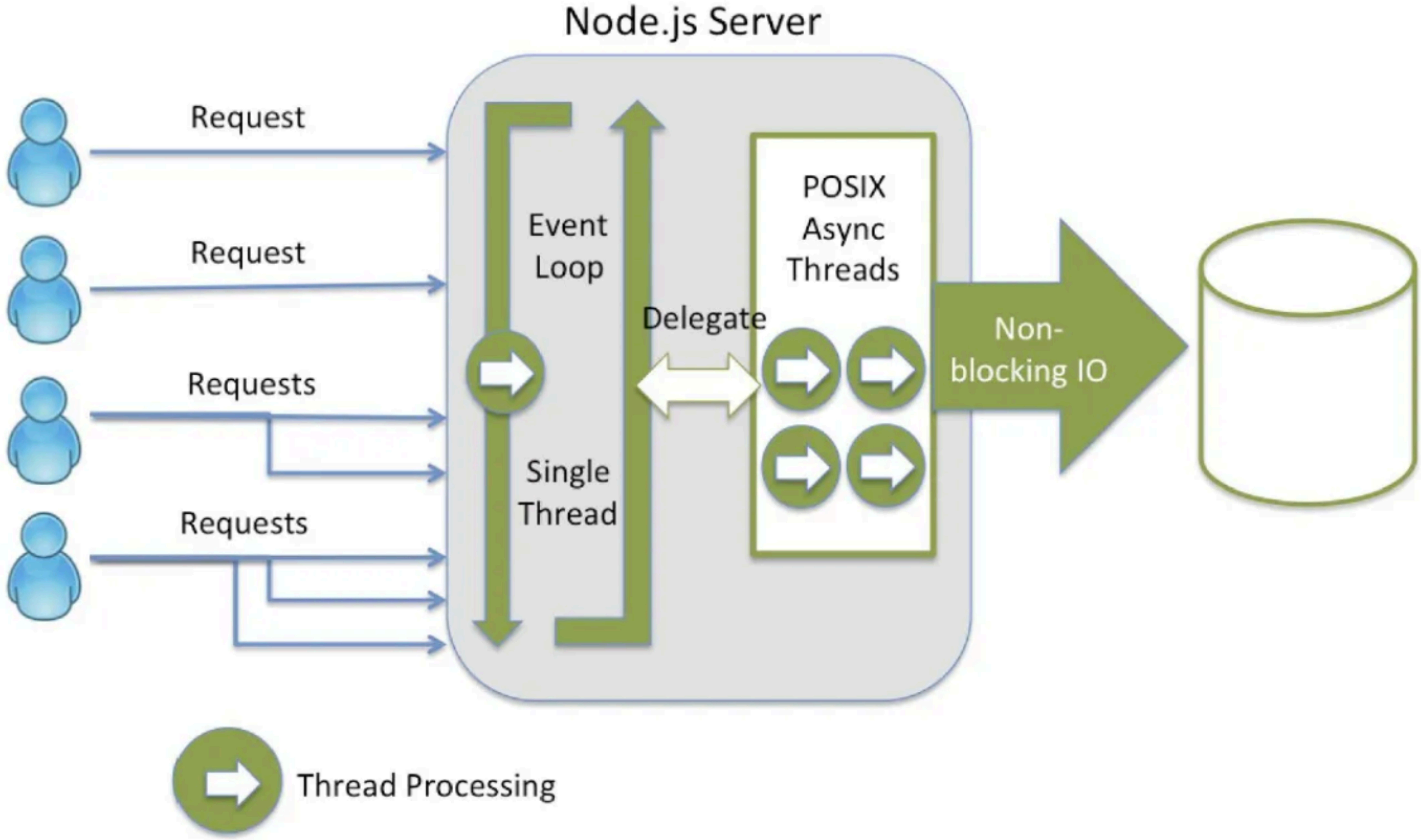
Full Stack Web Development

Traditional Web Application Structure



Node Structure





Node.js Server

Request

Request

Requests

Requests

Event Loop

Single Thread

Delegate

POSIX Async Threads

Non-blocking IO

Thread Processing

What is Node.js

- a complete software platform for scalable server-side and networking applications
- open-source under the MIT license
- comes bundled with a JavaScript interpreter
- runs on Linux, Windows, Mac OS & most other major operating systems

2009

- An early preview of npm was released just a few months after Node

- Ryan Dahl had the first presentation on Node.js at JSConf

2010

- The initial release of npm (in January)

- The express framework was released

- Socked.io came to life

- Ryan Dahl had the first Google talk about Node

2011

- Npm 1.0 was released

- One of the first books for Node.js beginners was released: The Node.js Beginner book by Manuel Kiessling.

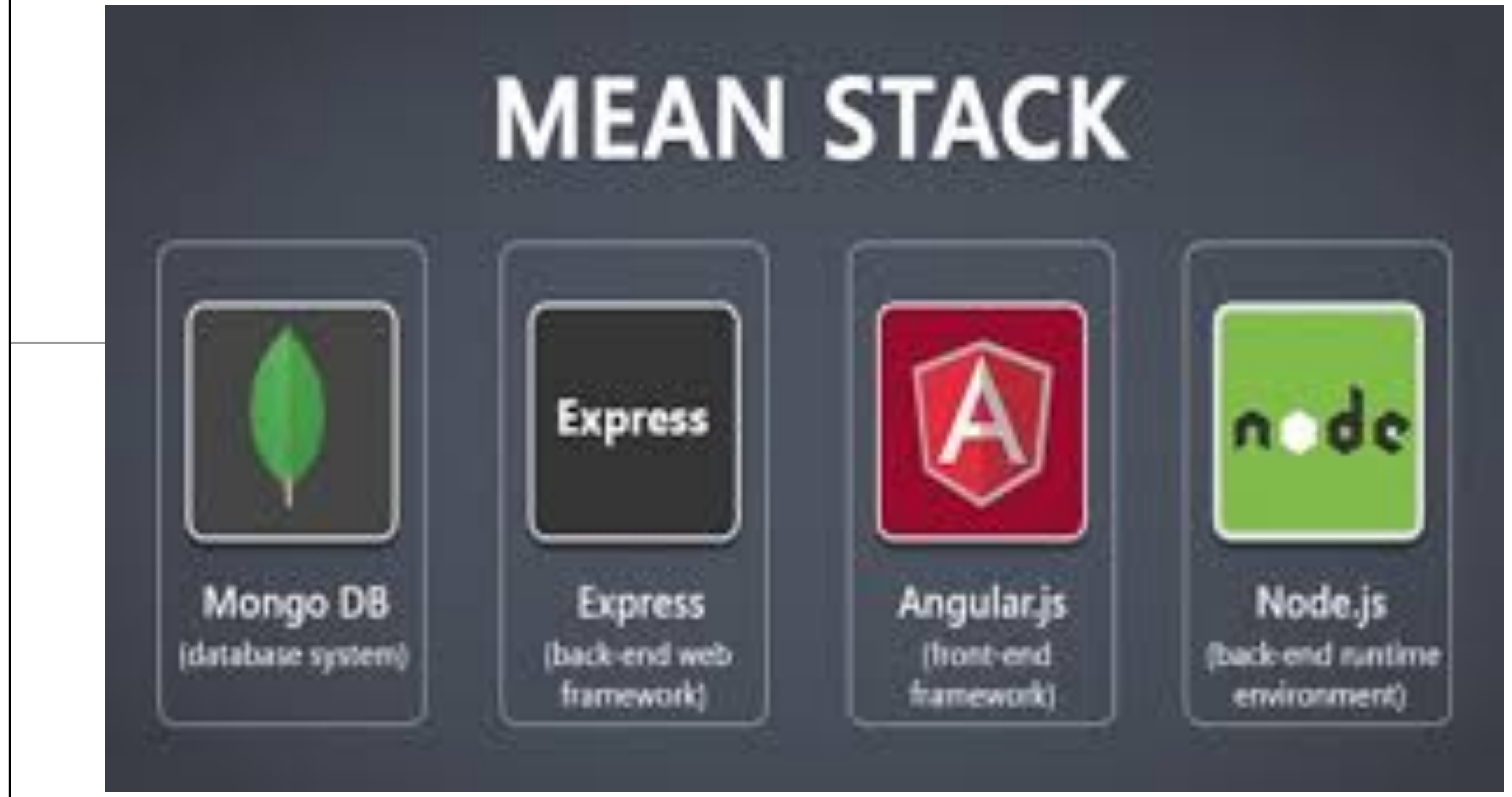
- 2 major companies started using Node.js: Linkedin and Uber

2013

- The initial concept of the MEAN stack was introduced by Valeri Karpov, a MongoDB developer.
- Ghost, a node.js open source publishing platform was released
- Paypal adopted Node, so did ebay (they both moved from Java)

2014

- Node leadership changes hands again as TJ Fontaine takes over
- Netflix takes the Node way
- This was the year then the io.js project forked, the year Node divided. Io.js released several version in tandem with the original Node.js team. Opinions were divided and people went in different directions.



PayPal

ghost

A proud non-profit organisation building technology for the future of journalism

NETFLIX

2015

- A house divided cannot stand on itself, so in June of 2015 the Node.js Foundation was created and the io.js project reunited with the original node project. It still remains and independent entity that works for the promotion and development of Node.js
- 3 months after the Node Foundation was launched TJ Fontaine stepped down as the lead of the project
- In September 2015 the Node.js and io.js combined in a single codebase with the release of Node 4.0. This was the original version 1.0 planned before the merger.
- In 2015 a plan was outlined for constant and regular Node.js version updates.

2016

- The leftpad incident
- Npm got to more than 200k users
- The first Node Interactive Europe conference happened
- Yarn was released
- Node.js 6 LTS version

2017

- npm focuses more on security
- HTTP/2
- V8 introduces Node in its testing suite, officially making Node a target for the JS engine, in addition to Chrome
- 3 billion npm downloads every week

2018

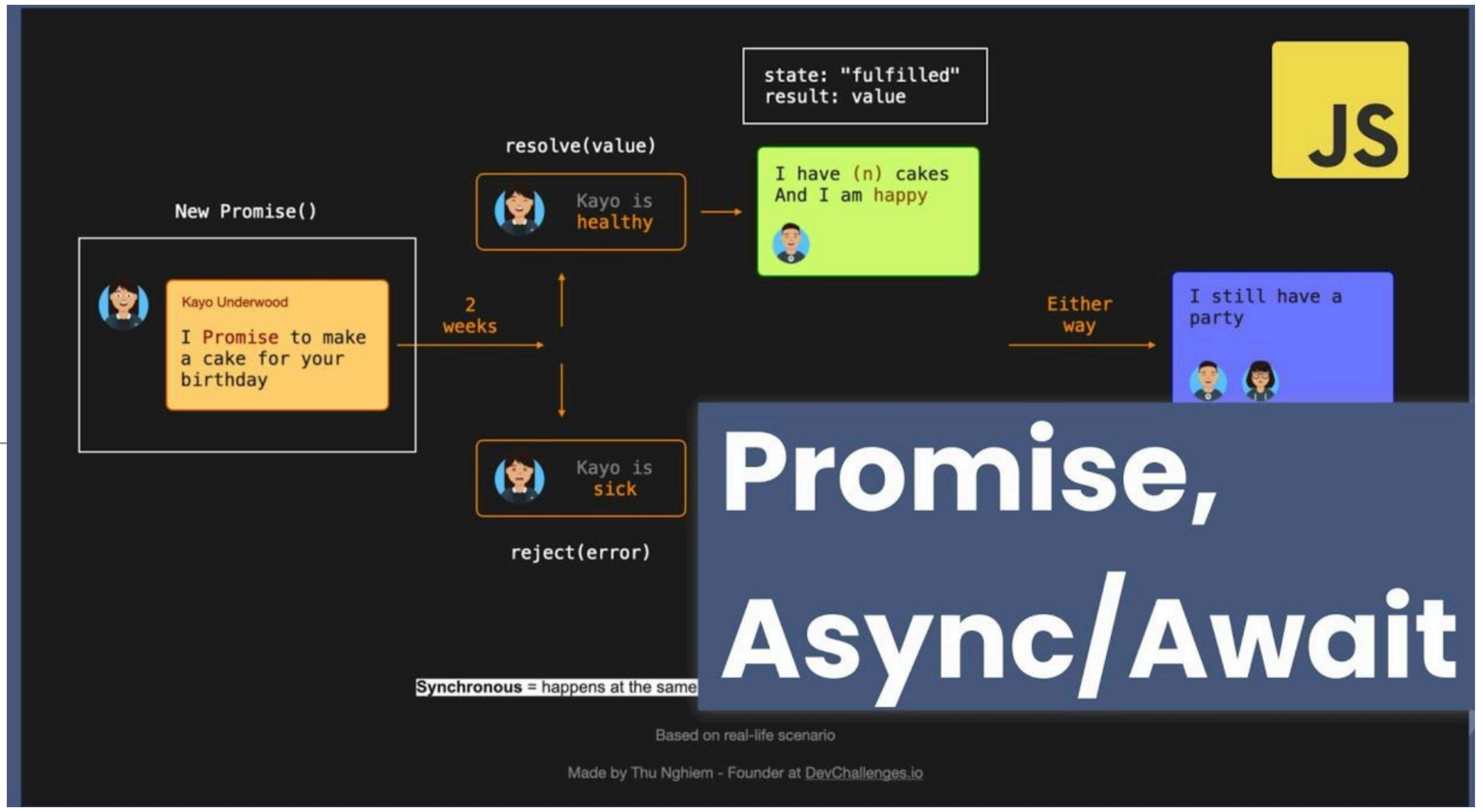
- ES modules .mjs experimental support

2019

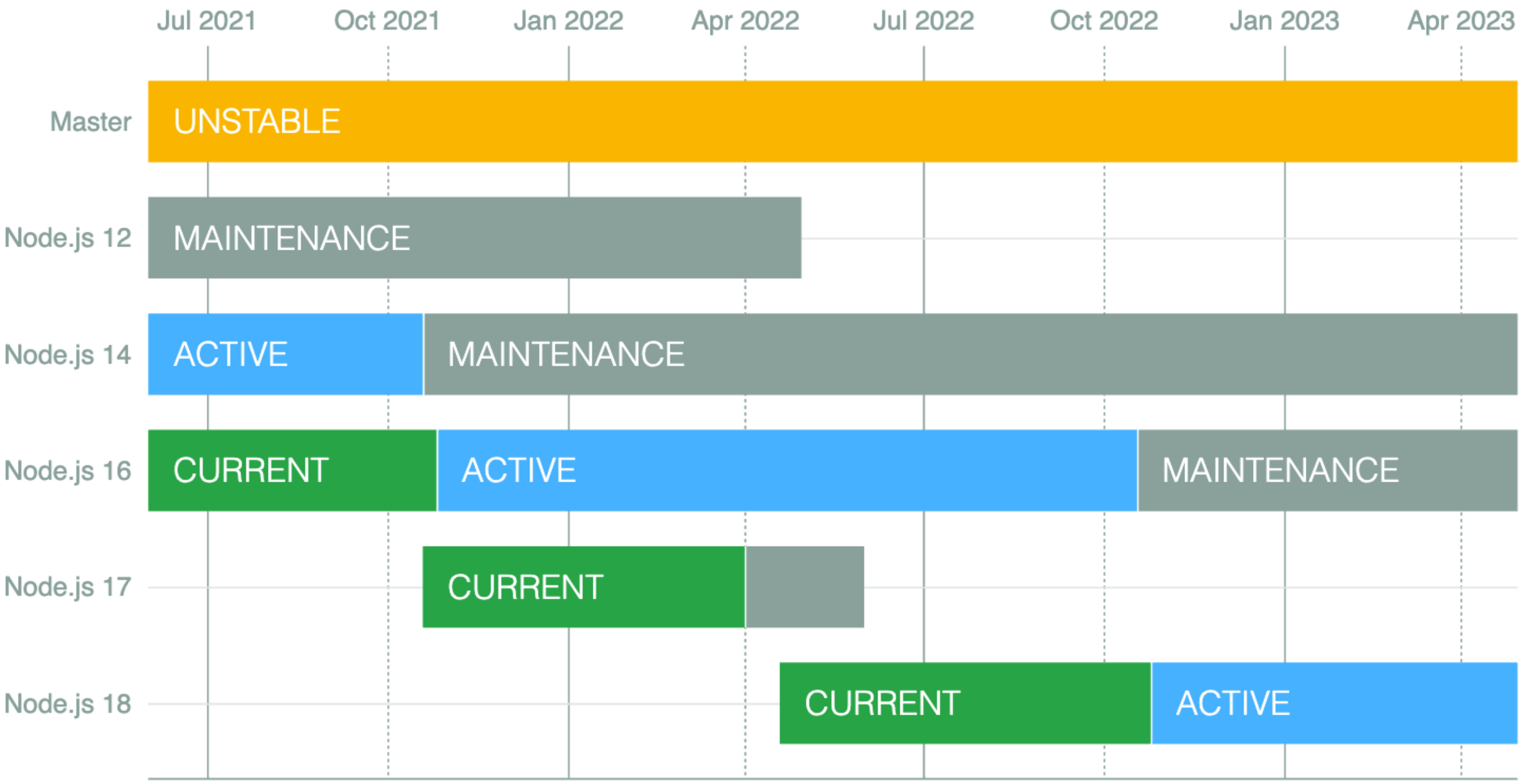
- Faster async/await implementation
- Node.js Foundation and the JS Foundation merged to form the OpenJS Foundation.
- New JavaScript language features

2020

- ECMAScript modules
- WebAssembly support
- ECMAScript modules finalised



- Major Node.js versions enter Current release status for six months, which gives library authors time to add support for them.
- After six months, odd-numbered releases (9, 11, etc.) become unsupported, and even-numbered releases (10, 12, etc.) move to Active LTS status and are ready for general use.
- LTS release status is "long-term support", which typically guarantees that critical bugs will be fixed for a total of 30 months.
- Production applications should only use Active LTS or Maintenance LTS releases.



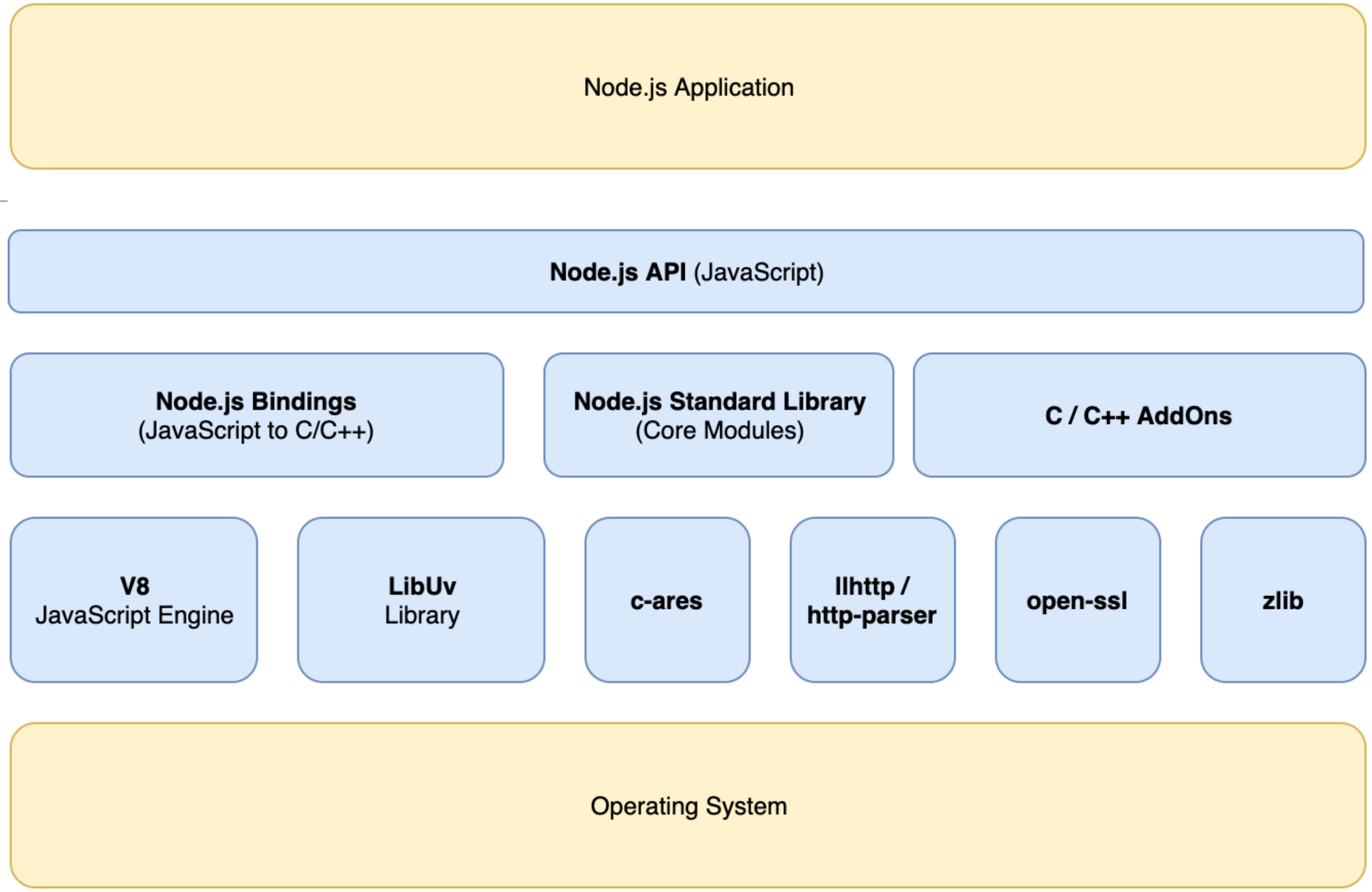
how it works

- Built on Chrome's V8 JavaScript runtime for easily building fast, scalable network applications

- Uses an event-driven, non-blocking I/O model that makes it lightweight and efficent, perfect for data- intensive real-time applications that run across distributed devices

Overall Structure

- All requests handled by the Main Single Thread
- API in JavaScript
- Node bindings allow for server operations
- Relies on Google’s V8 runtime engine
- Lib UV responsible for both asynchronous I/O & event loop



- Two major components:
  - V8 Engine written in C and C++
  - Modules, such as Http, FS, Crypto etc.. also written in C++

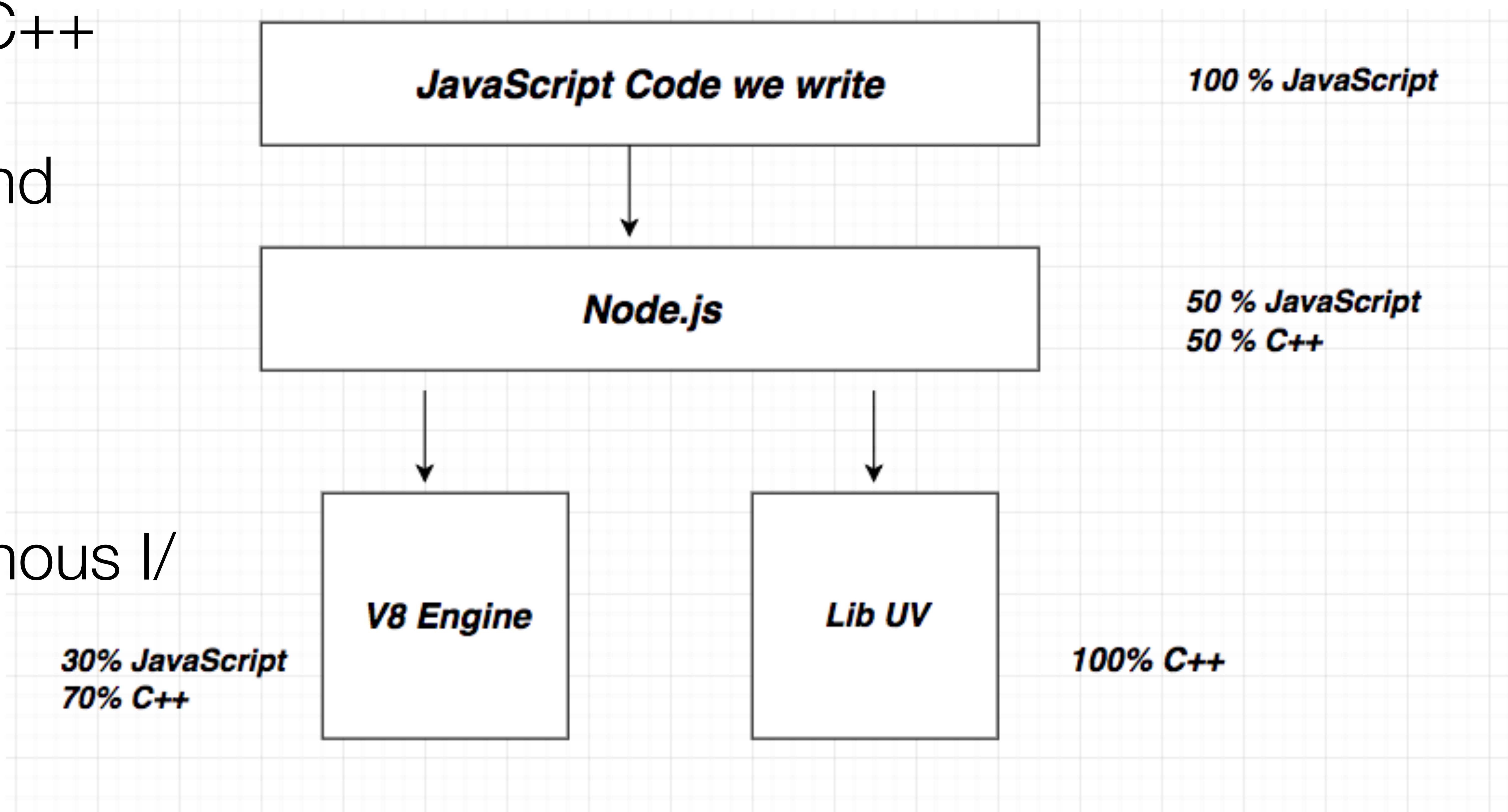
Main Components

v8 runtime engine

- Just in Time compiler, written in C++
- Consists of compiler, optimizer and garbage collector

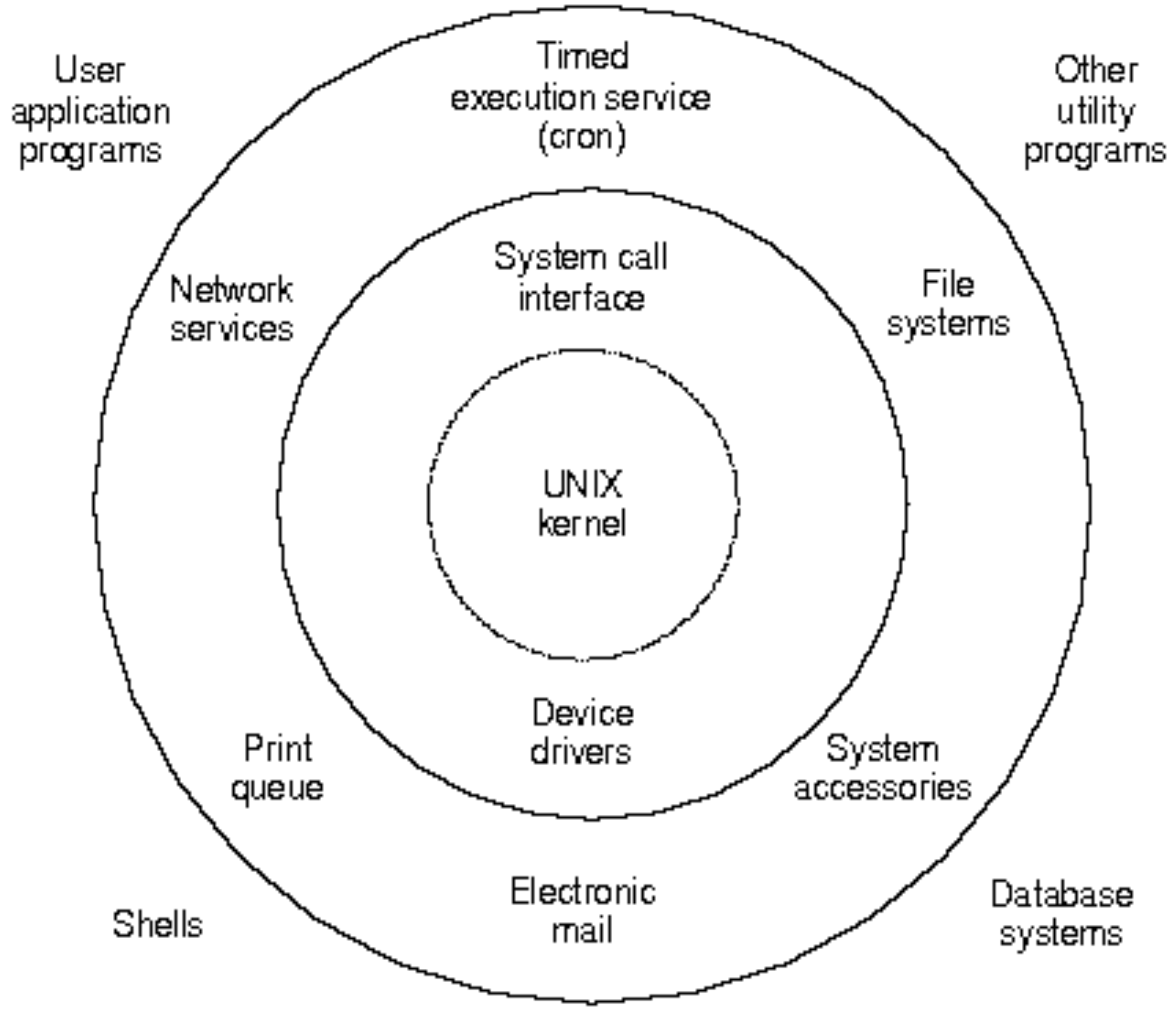
Lib UV

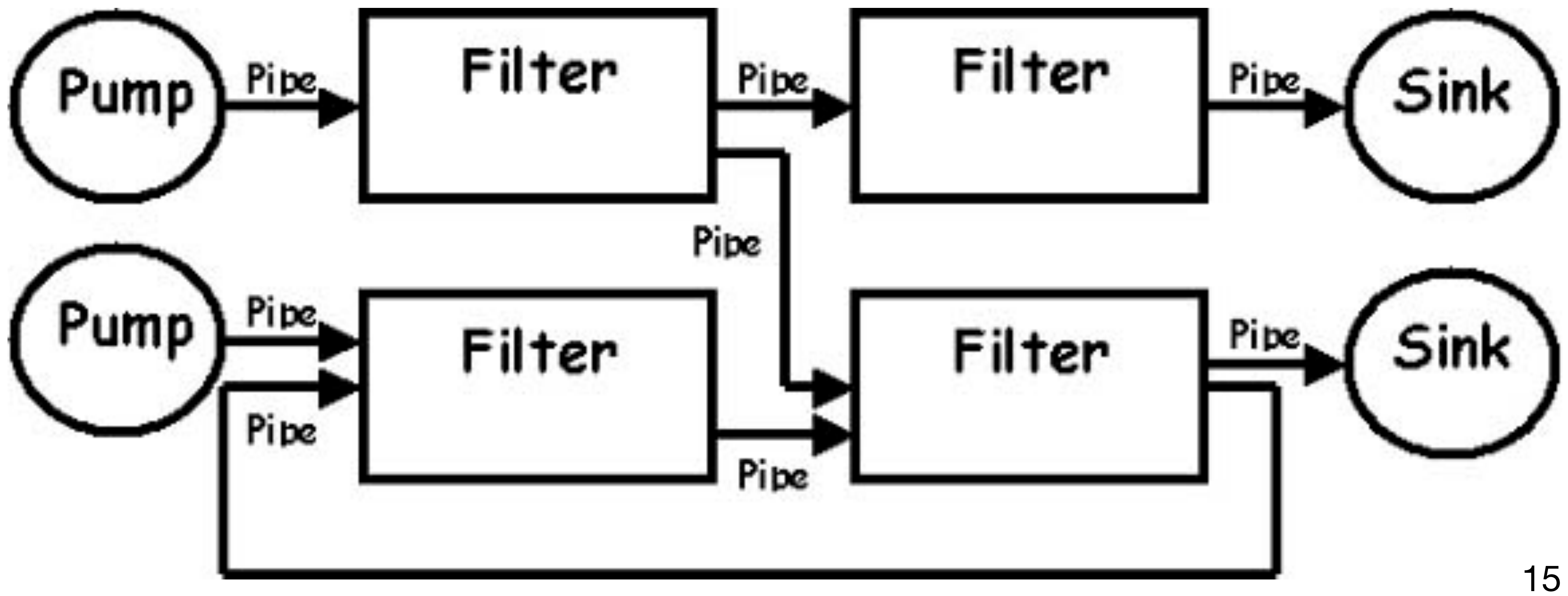
- Responsible for Node's asynchronous I/O operations
- Contains fixed-size thread pool



Major Influences

- Heavily influenced by architecture of Unix operating system
- Relies on a small core and layers of libraries and other modules to facilitate I/O operations
- Built-in package manager contributes to the modularity of Node





Single Threaded

- Most other similar web platforms are multi-threaded
- With each new request, heap allocation generated
- Each request handled sequentially

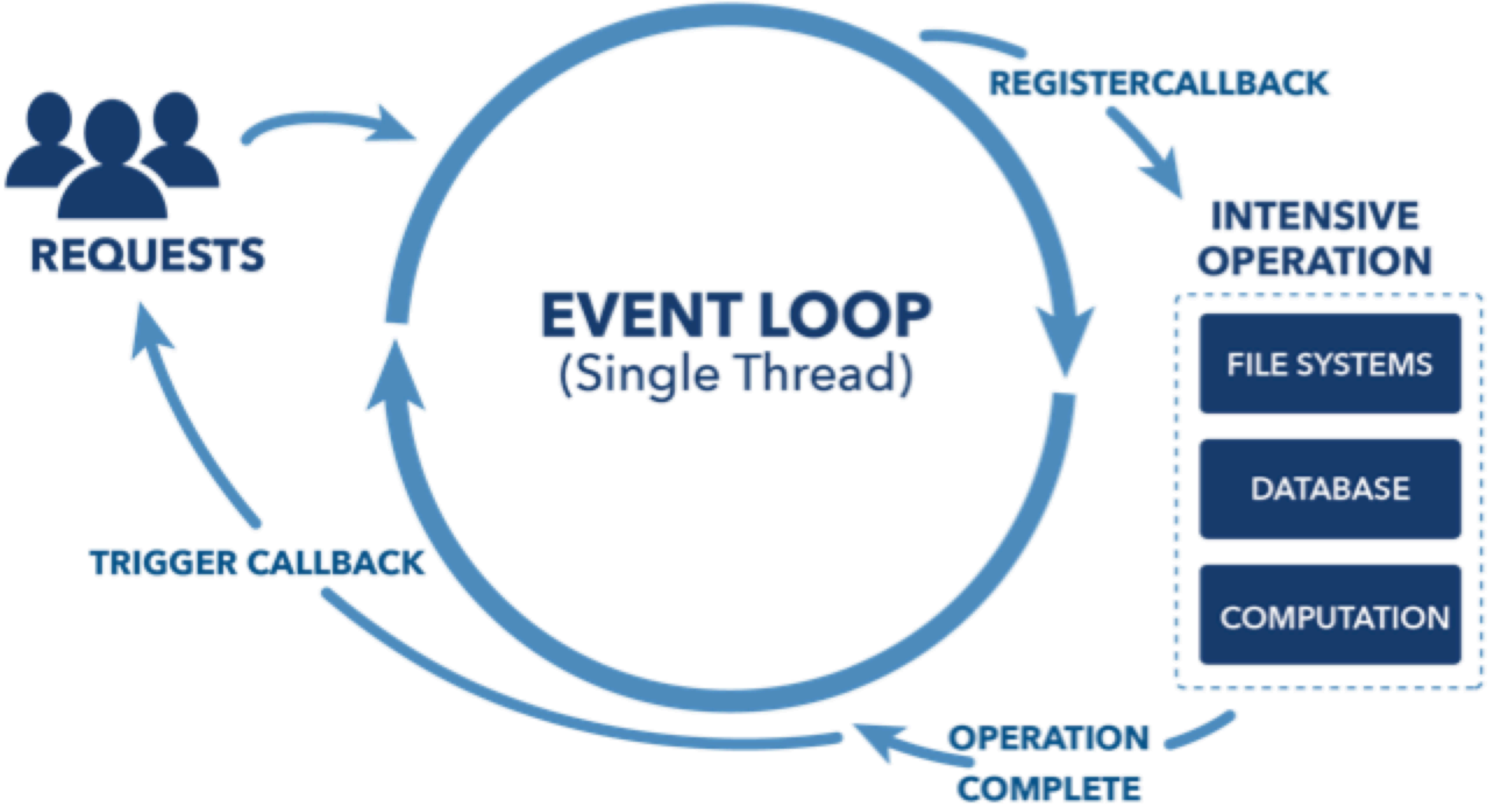
Event Loop

- Typically implemented using library via a blocking call, but Node is non-blocking throughout
- Implemented using language construct & Automatically terminated
- Tightly coupled to V8 engine

Non-blocking I/O

- All requests temporarily saved on heap
- Requests handled sequentially
- Can support nearly 1 million concurrent connections

how it works


REQUESTS
EVENT LOOP
(Single Thread)
REGISTERCALLBACK
INTENSIVE OPERATION
FILE SYSTEMS
DATABASE
COMPUTATION
OPERATION COMPLETE
TRIGGER CALLBACK

Advantages

- Because of its single-threaded, non-blocking scheme, Node can support nearly 1 million concurrent connections

- Asynchronous, event-based scheme allows for scalability, lower memory usage & CPU overhead

- Can be used to implement an entire JavaScript-based web application.

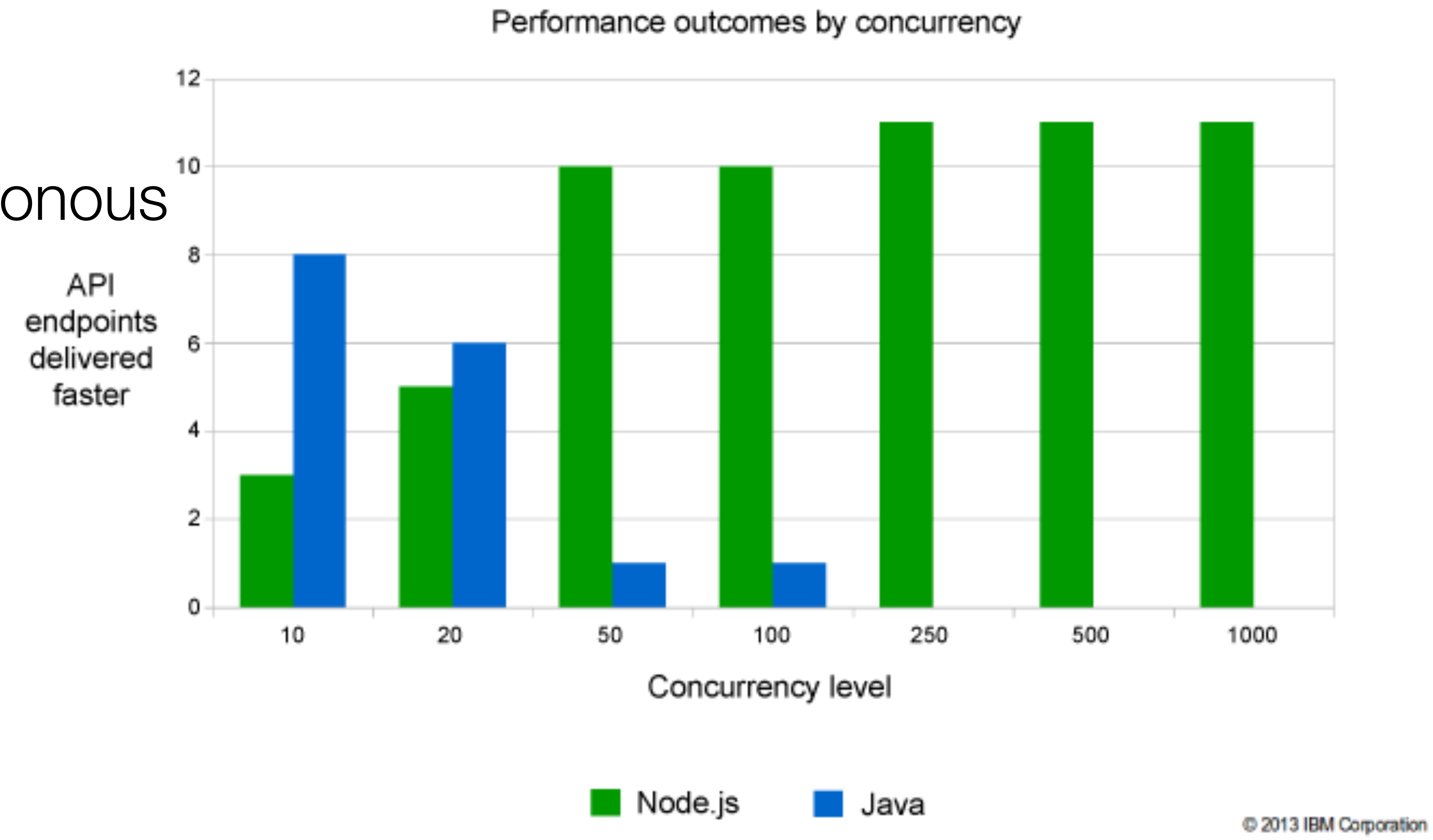
- Requests are acknowledged quickly due to asynchronous nature

- Native JSON handling

- Easy RESTful services

- Speedy native bindings in C

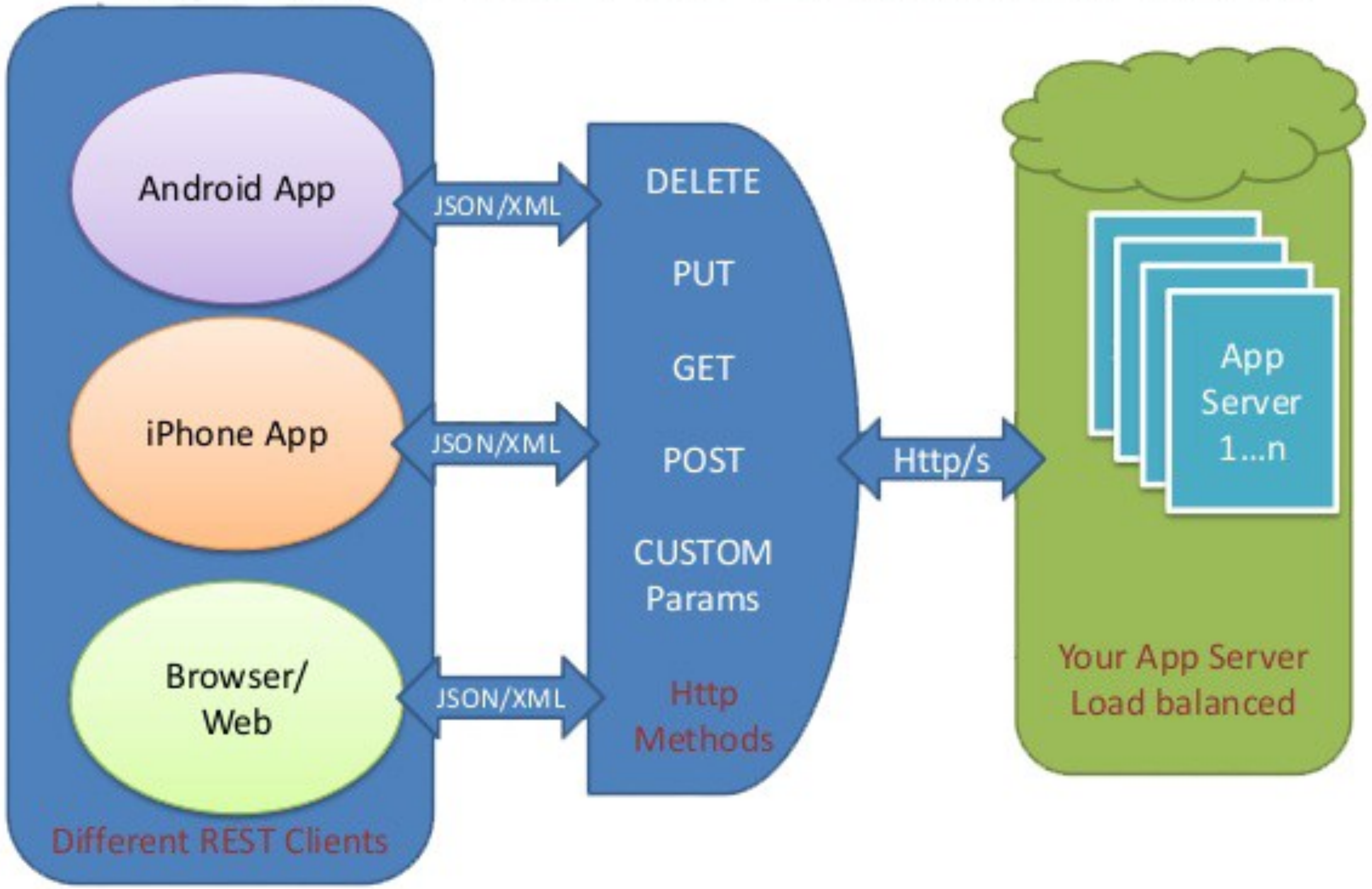
- Due to its real-time nature, it's possible to process files while they are being uploaded



© 2013 IBM Corporation

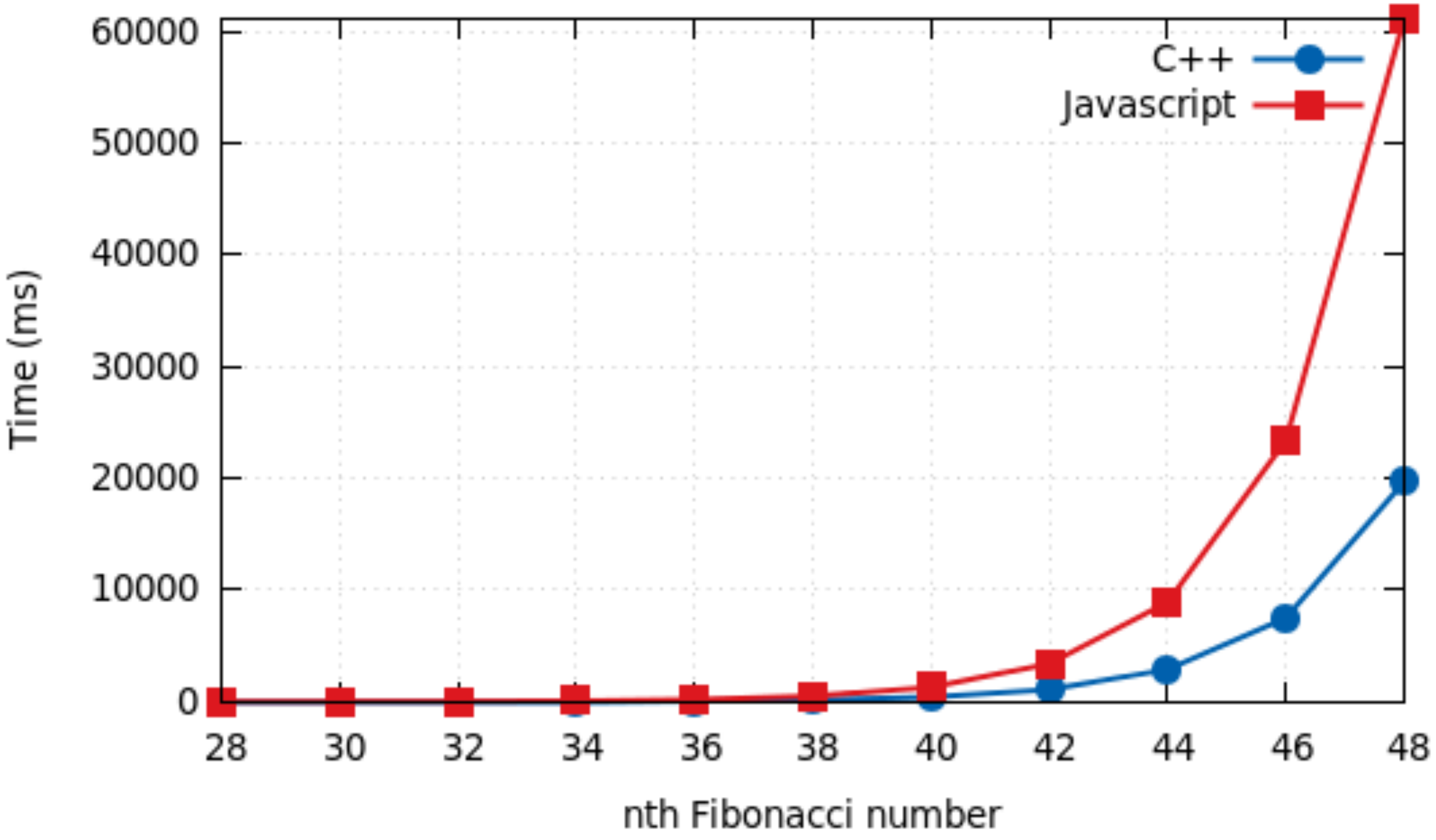
Best Suited For…

- REST + JSON APIs
- Backend for single-page web apps with same language for client and server
- Quick prototyping
- Rapidly evolving applications: media sites, marketing, etc.
- Chat applications
- Computing and orchestration tasks divided using worker processes



Limitations

- Node & V8 runtime engine are tightly coupled
- Because it is single-threaded, it has a single point of failure for all requests (low fault-tolerance)
- Not suited for CPU-bound tasks
- Not suited for Applications needing to process large amounts of data in parallel, unless using worker processes



Node Components

Full Stack Web Development