How Uber Uses Node.js to Scale Their Business

The ride-sharing platform Uber has an ambitious goal: to make transportation as reliable as running water, everywhere and for everyone.

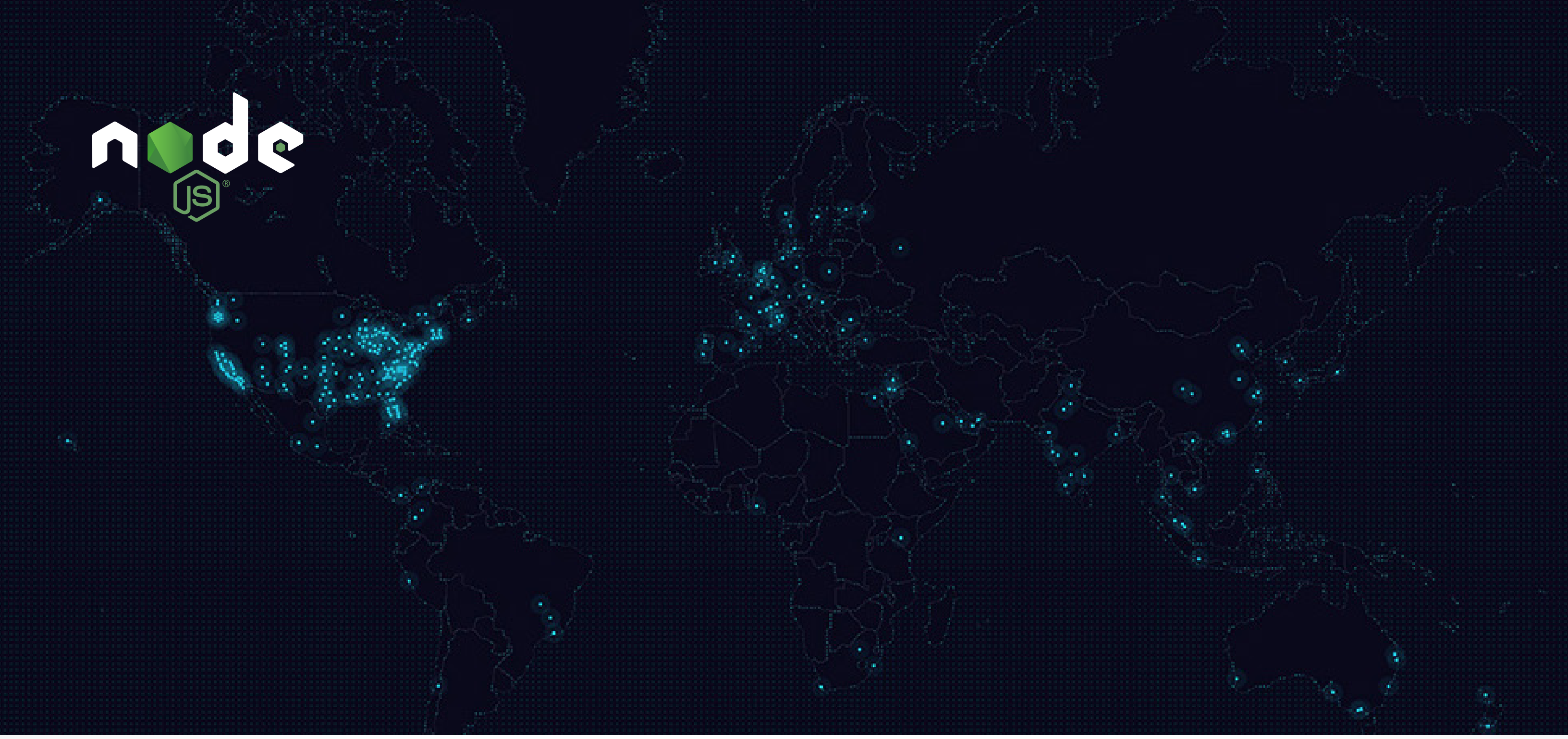
“You never think: ‘I’m going to have to plan out when to get a drink of water or shower,’” said Uber Senior Staff Engineer Matt Ranney. “You just go and get a drink of water or you take a shower when you feel like it, and that is how we want the Uber platform to work as well.”

> “You just go and get a drink of water or you take a shower when you feel like it, and that is how we want the Uber platform to work as well.”

It took humans a long time to get running water to be an on-demand service in most of the world; the Romans got off to a good start, but then there was an 800-year service interruption. That’s not the sort of timeline Uber is looking to follow.

To conquer the considerable challenge of ensuring a reliably excellent experience for its customers and drivers at a quickly-growing scale, Uber has built its massive matching system on Node.js. It’s one of the three companies that first put Node.js into full production (Voxer and Joyent being the other two) and credits the open source server-side technology with its ability to keep up with the pace of its massive business demands.

The company is doubling in size just about every six months; Uber now processes many of millions of rides every day, operates in 68 countries across six continents, and has drivers serving riders in more than 300 cities in just five and a half years of operation. Uber needs a system that will keep running no matter what, and that’s why its engineers have chosen Node.js.



Uber's operational markets as of April 2015, when they reached 300 cities.

Why Node.js?

There are three core strengths that make Node.js a particularly good fit for Uber.

1. Node.js handles asynchronous I/O requests with a non-blocking, single-threaded event loop. It is particularly well-suited to distributed systems that make a lot of network requests.

2. Node.js - and JavaScript in general - is excellent for quick iteration; programs can be inspected and errors can be addressed on the fly without requiring a restart, so developers can publish and deploy new code constantly.

3. The active open source community continuously optimizes the technology; it gets better, all the time, practically on its own.

Node.js processes lots of information, quickly

Uber's matching system creates an enormous amount of supply notifications (driver partners) and demand requests (riders). But a ride request doesn't just elect the eventual driver; it pings others on the network throughout the matchmaking process. (Nearest drivers are stored in a geospatial database which is continuously updated by every active driver on the network as they move about the city.) The requests come and go at varying speeds thanks to mobile internet connection speeds, but no request can be treated as a second-class citizen.

Remember, Uber's goal is to be just like running water: when you turn on the faucet, you get the water right away. People want to get where they're going. Matching riders with driver partners is *the* mission-critical system for Uber. If this breaks, Uber doesn't have a business.

Fortunately these are exactly the type of problems Node.js is built to address.

"Node.js is particularly well-suited to writing systems that have all their state in memory," said Kris Kowal, a Software Engineer at Uber. "They do not have to externalize the concerns of a distributed system. As a consequence, the systems can be more available, and they can respond more quickly to requests by eliminating the reading/writing and the serialization of state into a database."

> "Node.js is particularly well-suited to writing systems that have all their state in memory"

Where things get tricky is handling millions of these requests daily. This creates some significant engineering problems concerning distributed computing at a massive scale.

Promoting a creative, productive engineering culture

That's where the second strength Uber found in Node.js (quick iteration) comes into play: through an interactive testing environment called REPL - Read Eval Print Loop - JavaScript allows developers to deploy new code - and fix the errors that new code may create - without having to stop any processes.

"One of the things that makes Node.js uniquely suited to running in production is that you can inspect and change a program without restarting it," said Ranney. "So very few other languages offer that capability. Not a lot of people seem to know that ability exists, but indeed you can inspect and even change your program while it's running without restarting it."

> "One of the things that makes Node.js uniquely suited to running in production is that you can inspect and change a program without restarting it"

That allows on-the-fly problem solving in production, and that creates a special culture of both creativity and productivity inside Uber's fast-growing engineering team.

"Everyone is so motivated to solve problems and there's so much excitement and energy and enthusiasm about building software," said Ranney. "The team is great and super productive."

"There's a culture of 'publish often' which is fantastic here," said Kowal. "Get your work done. Deploy it. Start creating business value before it's even fully done."

Uber's engineering team has built three pieces of software to keep their matching system running all the time at the massive scale required: Ringpop, TChannel, and Hyberbahn.

Ringpop

Ringpop is scalable, fault-tolerant application layer sharding that allows Uber's supply/demand entities to exist entirely in process while rider requests and driver availability are matched. The requests run on whichever CPU is available in the moment and that CPU either handles the request, or forwards it on to be handled by one better suited to do so.

TChannel

TChannel is a networking multiplexing and framing protocol for RPC. It uses a request/response model with out-of-order responses, where slow requests at the head of the line will not block subsequent faster requests, and it creates a high-performance forwarding path for those requests.

Hyperbahn

Hyberbahn is a service-to-service discovery and routing system. It allows Uber to adapt in real time to ensure user requests get where the need to go to be resolved, no matter what's happening in the system. It's elastic, massively distributed, and fault tolerant.

Uber Engineering is steadily deploying each of these programs across their entire network; they're doing it with zero down-time so the riders looking for drivers never are left waiting. Uber has open-sourced each of these projects; all the minutiae are available on Github for all to see.

You’re never going it alone

That is the third major strength of Node.js: the open source community encounters problems, records them and tackles solutions with gusto. Node.js is not without its issues; the major and most prevalent problems are memory leaks that rapidly degrade performance.

“There is going to come a time, maybe not when you’re first getting started, but there’s going to come a time where your Node.js program is too slow, or you’re trying to track down a memory leak,” Ranney said. “That’s just the tax you’re going to have to pay at some point; if you’re really going to put this thing into production, you have to learn how to understand and diagnose memory leaks.”

The memory leak issue necessitates two things: a monitoring system so you can recognize a leak is happening quickly, and a way to find the Node.js process that’s using all the memory. The fix, Ranney and Kowal say, is either a heap dump or a core dump, or some variation of the two.

But it’s a known issue, so people have started to create tools to diagnose and solve these problems. Joyent, Node.js’ original open-source shepherd, has tools for this exact problem. NodeSource is another; they’ve created what they call an enterprise-grade Node.js version called N|Solid that comes with comprehensive tooling and services.

And the open source community—now shepherded by the Node.js Foundation—is always hard at work creating similar solutions. Node.js is built on top of Google V8, an open source project that compiles JavaScript to machine code. Google uses it to—among many other things—make Chrome and Gmail work, so they put a lot of resources toward making it better.

“The community is now producing tons and tons of software,” Ranney said. “Honestly the hardest part, I think, is not having to build your own stuff, it’s that you just have to find your way through the maze of different solutions someone else has made.

“By building on Node.js’s actively-developed, open-source system, we get the benefit of lots of people making the software better without us doing anything. We can sort of sit back and it just gets better on its own, which is pretty great.

> “By building on Node.js’s actively-developed, open-source system, we get the benefit of lots of people making the software better”

“It’s also interesting that the degree to which we are engaged in the community, we can just get our problems solved. Our problems become the community’s problems.”

Kowal also sees JavaScript’s growing ubiquity as a major benefit of adopting Node.js; the language is now being used from the very top interaction layer in browsers all the way deep into the server.

“I don’t think that we’ve even begun to fully realize the potential for JavaScript that’s being shared client-side and server-side and the many layers in between,” Kowal said. “The potential for modular, reusable code that runs literally anywhere still has a lot of potential that we haven’t begun to unlock.”

Transportation that’s always available

Uber needs its systems to scale quickly alongside the growth in demand for its service. “Uber is now doing over two million RPCs per second at peak across the Node.js fleet,” Ranney said. The company is aiming to add 1,000 engineers in 2016 to sustain that rapid growth; those engineers will be constantly expanding, improving and adjusting its matching system to handle the load. Because its core, mission-critical system uses an open-source system based on JavaScript, new Uber engineers can hit the ground running, deploy code early and often, and solve problems as they arise without sending the system offline.

> “Uber is now doing over two million RPCs per second at peak across the Node.js fleet”

Somewhere in the world, someone needs to get where they need to go, and without the flexibility and reliability that Node.js provides that someone might be left thirsty for another transportation option.