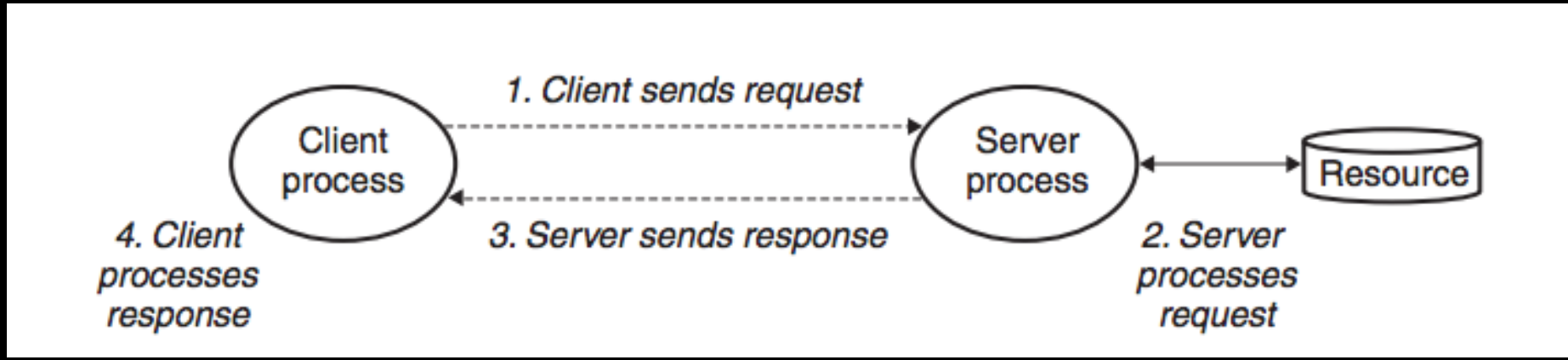
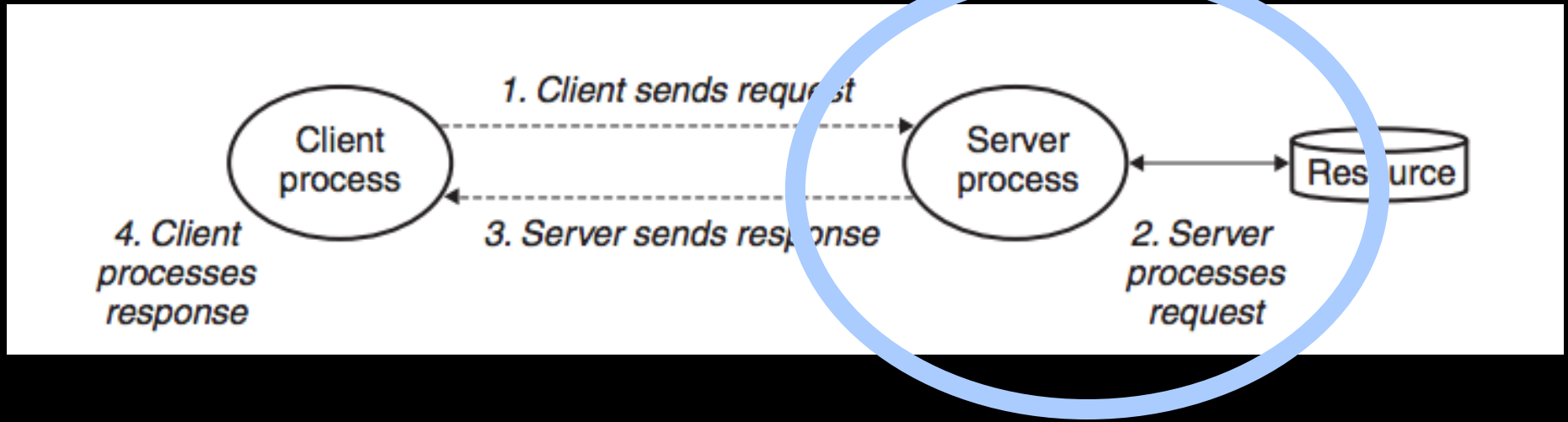
node.js

kevin schmid



Computer Systems: A Programmer's Perspective, Second Edition



Computer Systems: A Programmer's Perspective, Second Edition

Web Servers

- Receives HTTP **request**, issues HTTP **response**
- Reminder about HTTP
- In the CS50 Appliance, **Apache** is set up to work with PHP
  - PHP: scripting language
  - C$50 Finance

Web Servers

- When a request is received, must determine appropriate action to take:
  - Serve a **static** file (index.html, picture.jpg, thisisnotavideo_i_promise.mp4)
  - Run a script, like a PHP program (**dynamic**)
  - And more…
- Then issue response…

How can a web server handle lots of users at same time?

Apache's Solution

- Depending on configuration, Apache handles requests in separate **threads** or **processes**
  - For more, take courses in systems (CS61)
  - Allows for concurrent handling of requests
- Are there any issues with this approach?
  - Threads/processes are expensive: memory, cost of context switch

Observation

- For certain web applications, much of the time spent handling a request is time… waiting.
  - Getting data from a database (e.g., MySQL)
  - Reading a file from the computer's disk
  - I/O tasks
- And not that much time is spent doing *computational* work.
- Chat servers, for one

So...

- *Recap*: Apache / similar web servers fork threads/processes for each request. Can be wasteful, since most time spent is time waiting.
- This begs the question: do we even need multiple threads or processes?

Idea

- Let’s only use **one thread** that constantly checks to see if new things have happened
  - Somebody made a request to the server!
  - Something came back from the database!
  - That file we were looking for is ready for reading!
- When these things happen, run the **(non-computationally-intensive)** code that does the useful stuff
- Implications?

Idea (continued)

- Called an **event loop**, and is the basic concept behind node.js and similar event-driven systems
- But… how do we make this happen?
- Consider this C code:

```
FILE *f = fopen(“new_katy_perry_song.mp3”, “r”);
// work with `f`
```

```
void code(FILE *f) {
    // do stuff with f
}

fopen("new_katy_perry_song.mp3", "r", code);
// do other stuff, but can't assume file is
open here
```

node.js

- Offers the kind of framework you need to build event-driven servers like this
  - Asynchronous I/O libraries
  - Event loop
- … in JavaScript!
  - Built on top of Google’s V8 JavaScript
  - Why is this striking?

github.com/kevinschmid/nodecode