Web программирование

Web servers

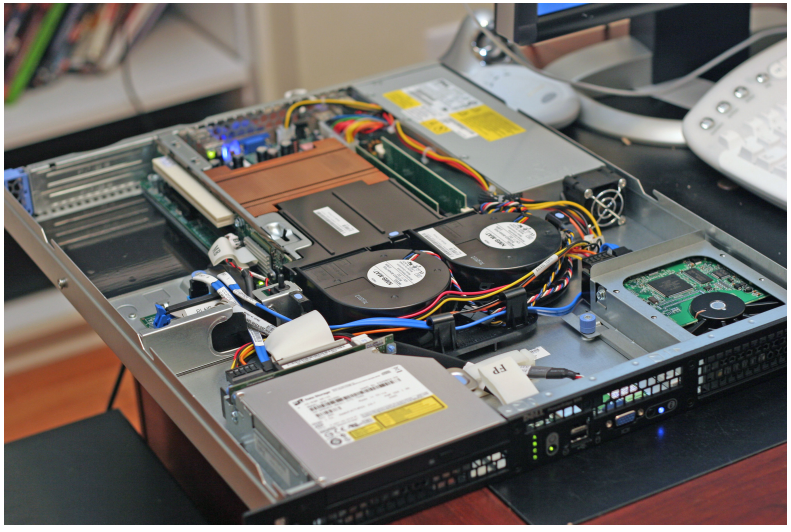
Игорь Родионов

Омский Государственный Технический Университет

кафедра Информатики и вычислительной техники

ОмГТУ, 2014.

Web Server

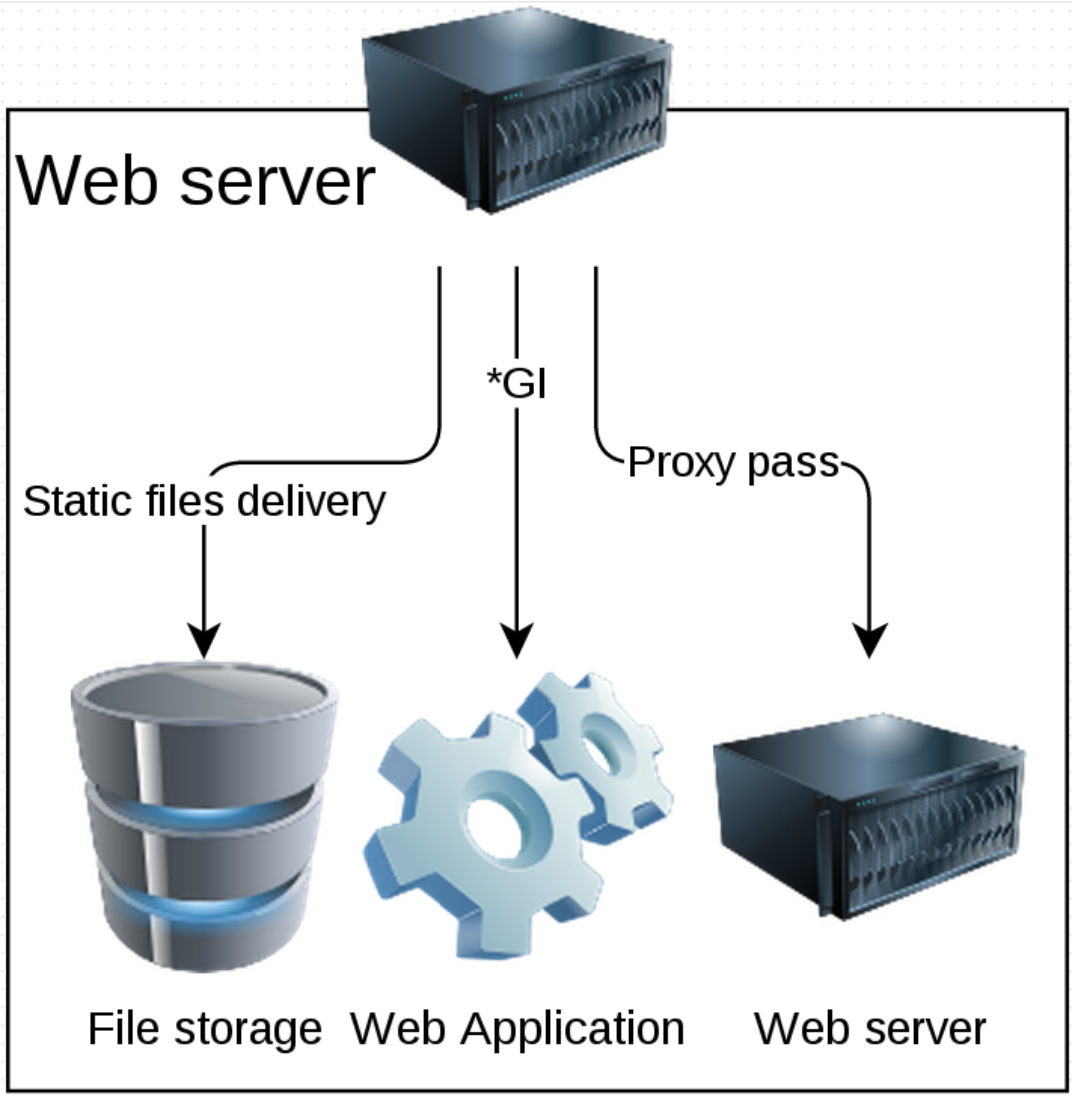


Web Server - компьютер используемый для генерации и\или доставки web содержимого.

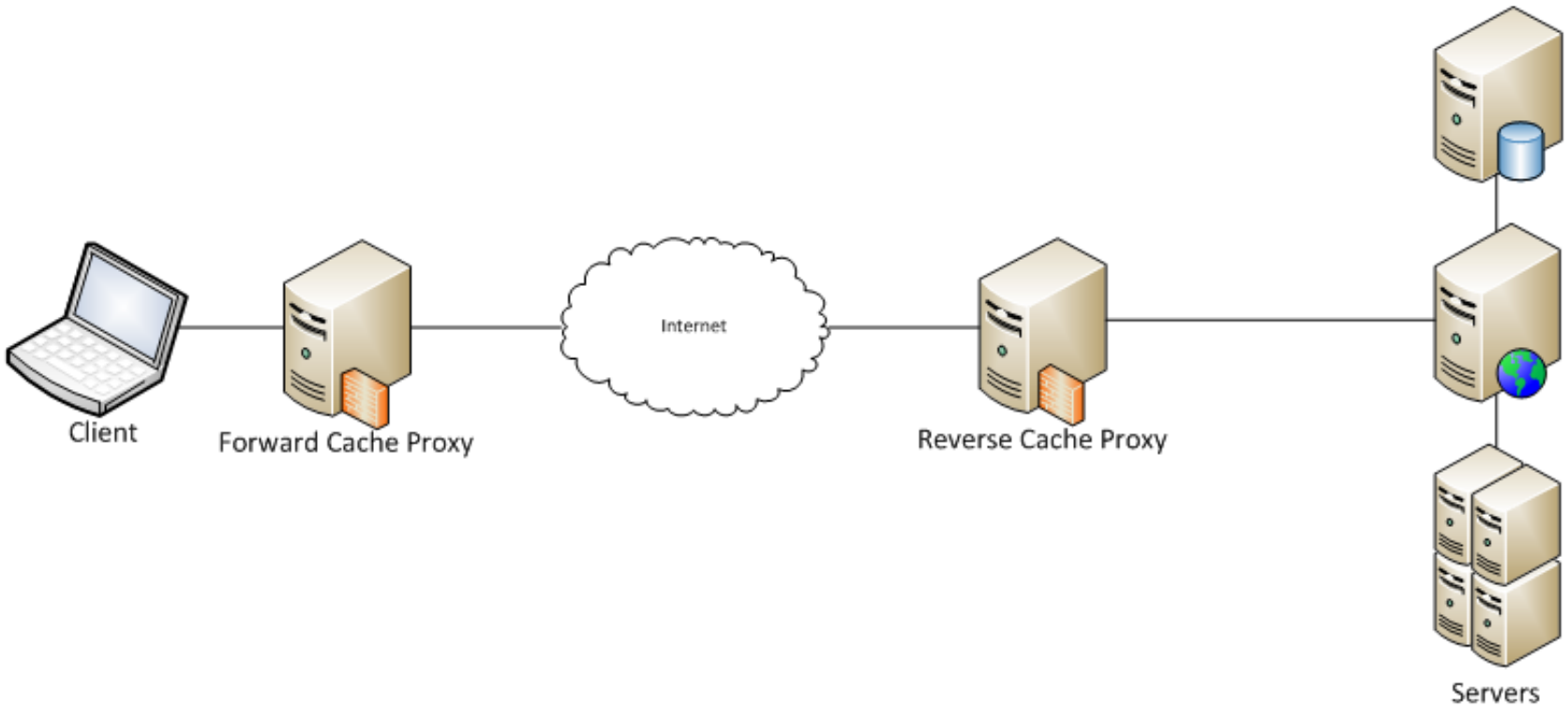
Web Server

Web Server - программное обеспечение используемое для генерации и\или доставки web содержимого.

Web Server - Архитектура



Web Server - Proxy



Web Server - CGI

Common Gateway Interface — стандарт интерфейса, используемого для связи внешней программы с веб-сервером. Интерфейс разработан чтобы работать со стандартными устройствами ввода-вывода.

Web Server - CGI

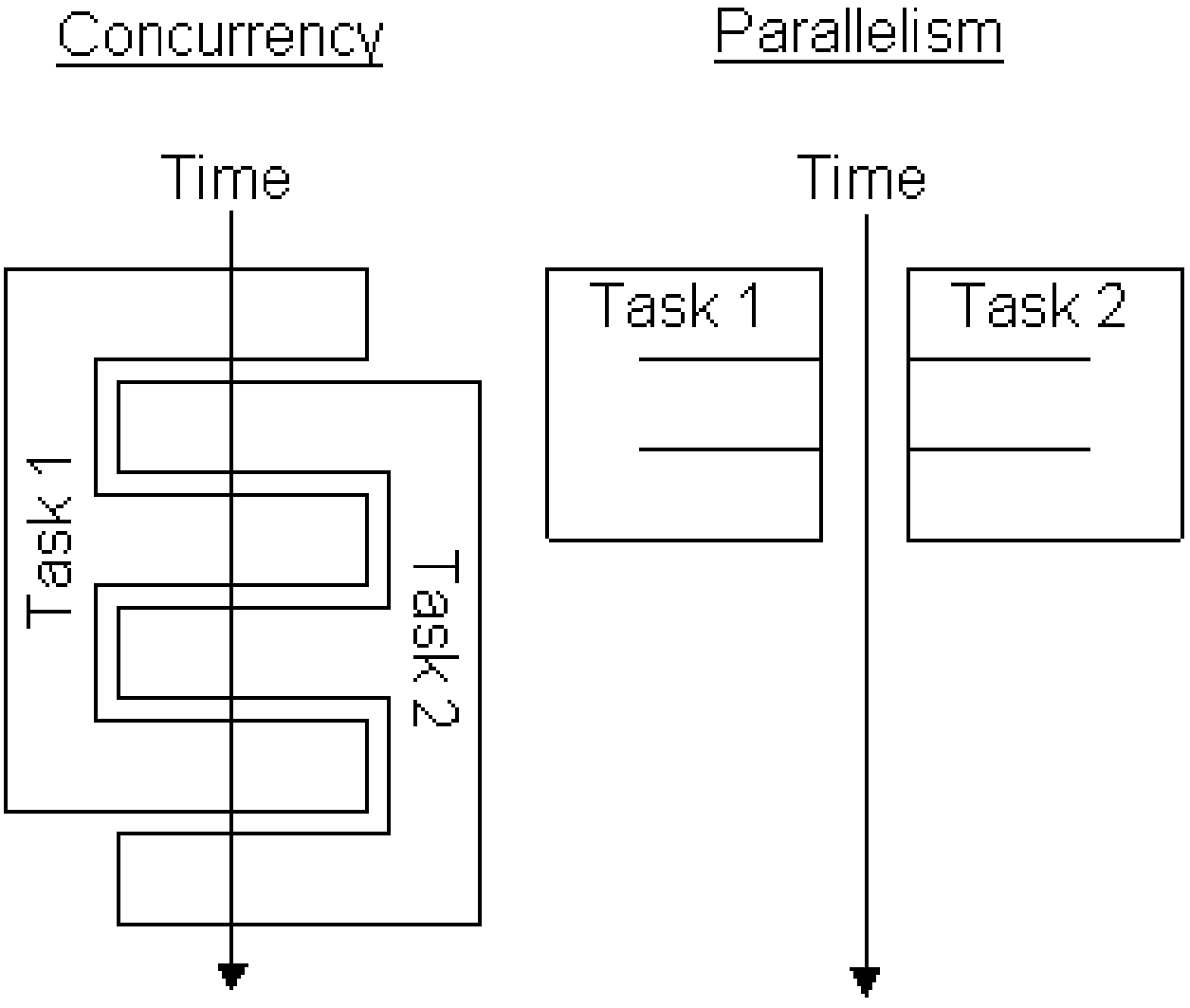
```
1 Program first;
2 Begin
3   Writeln('Content-Type: text/plain');
4   Writeln;
5   Writeln('Hello, world!');
6 End.
```

Questions

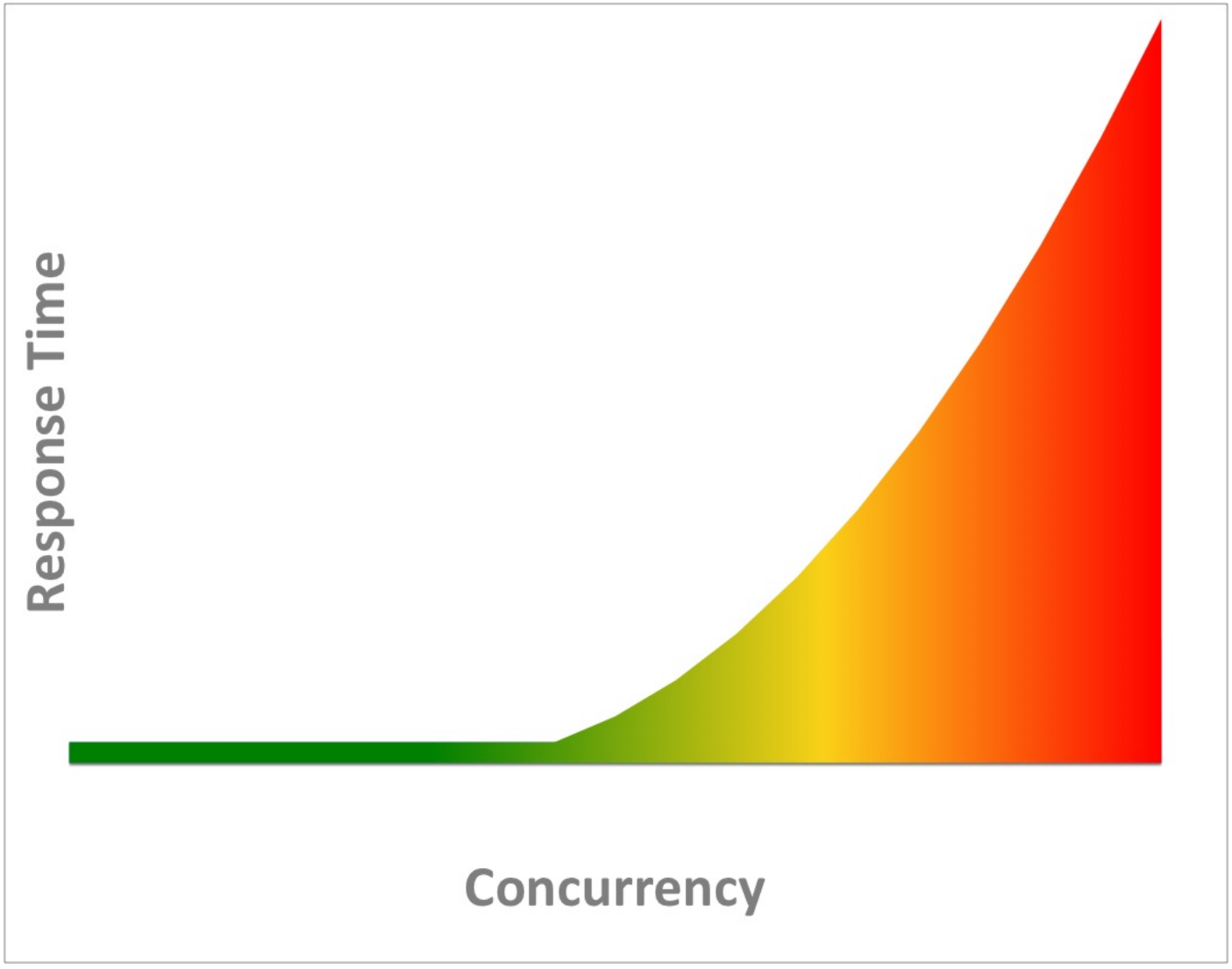
Вопросы?

Concurrency

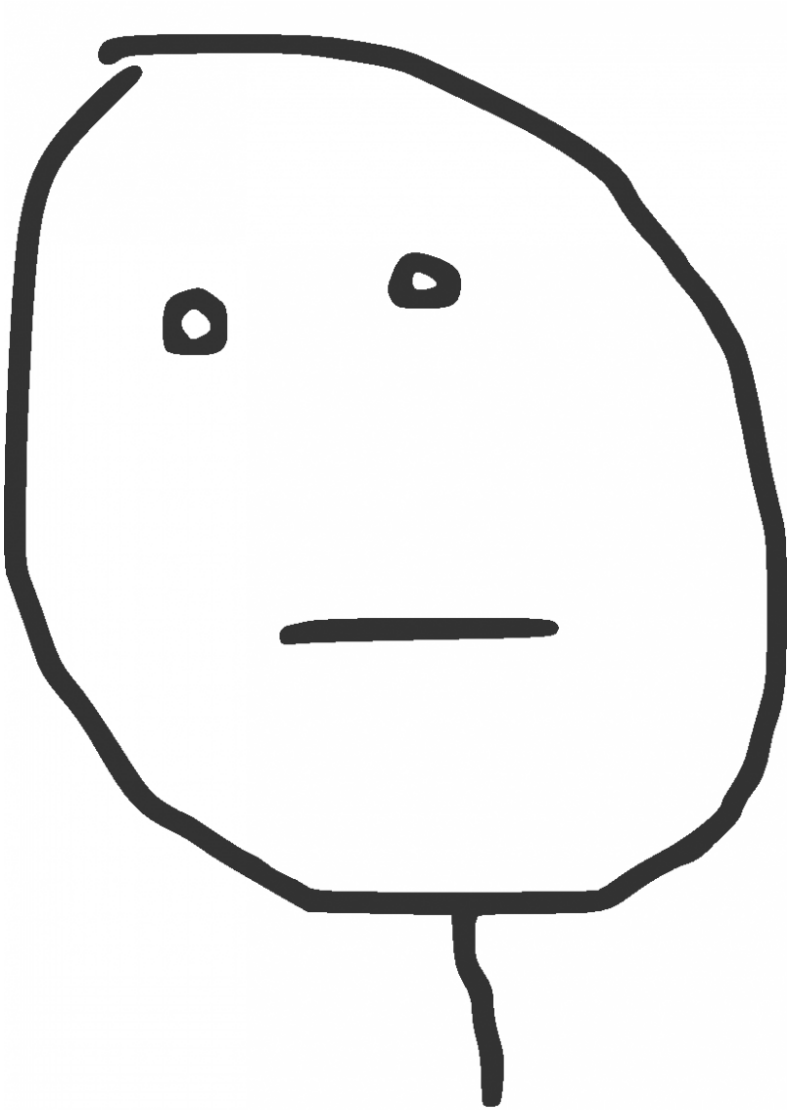
Concurrency



Concurrency

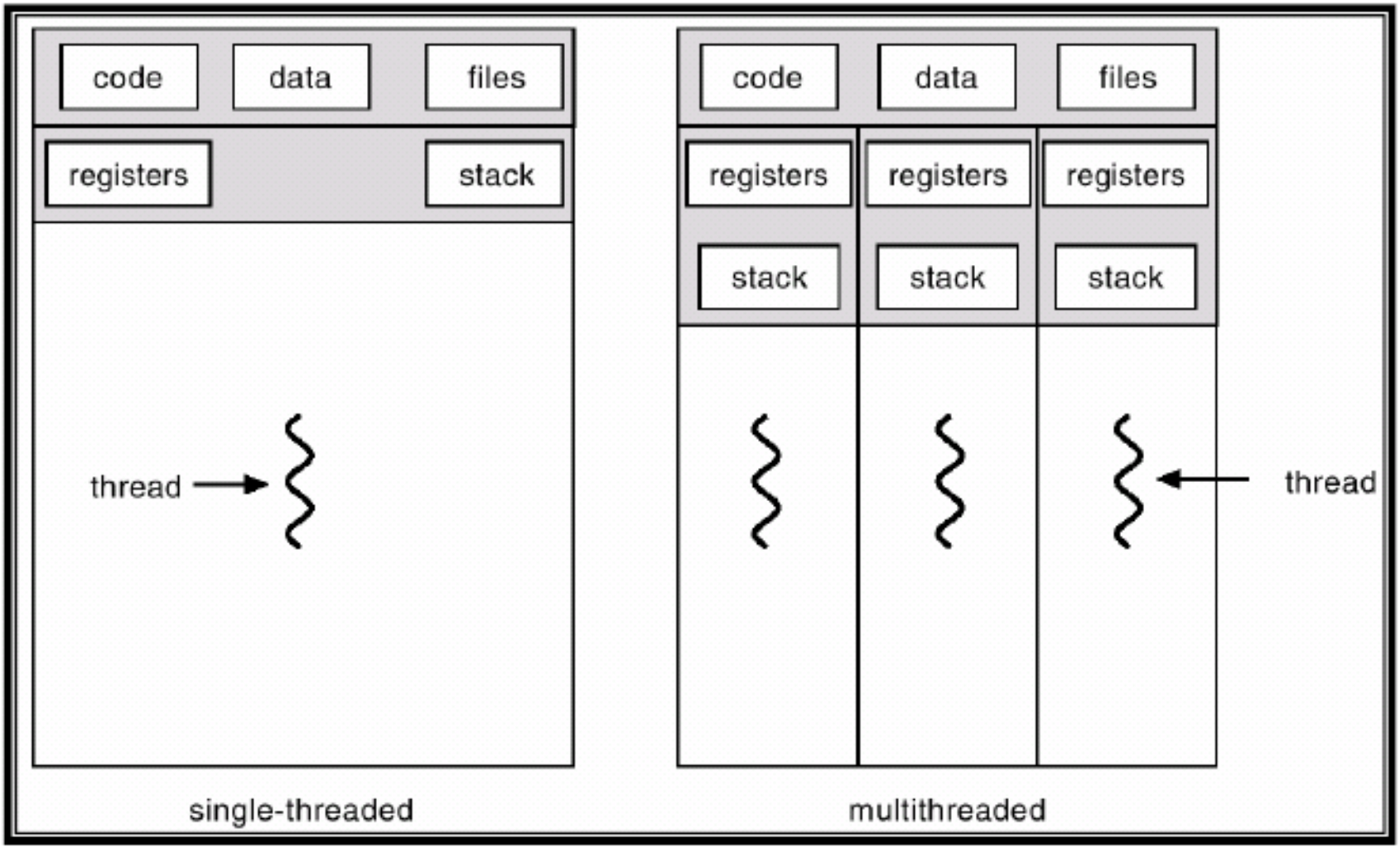


Concurrency



Этот неловкий момент, когда для объяснения материала курса, надо рассказать другой курс.

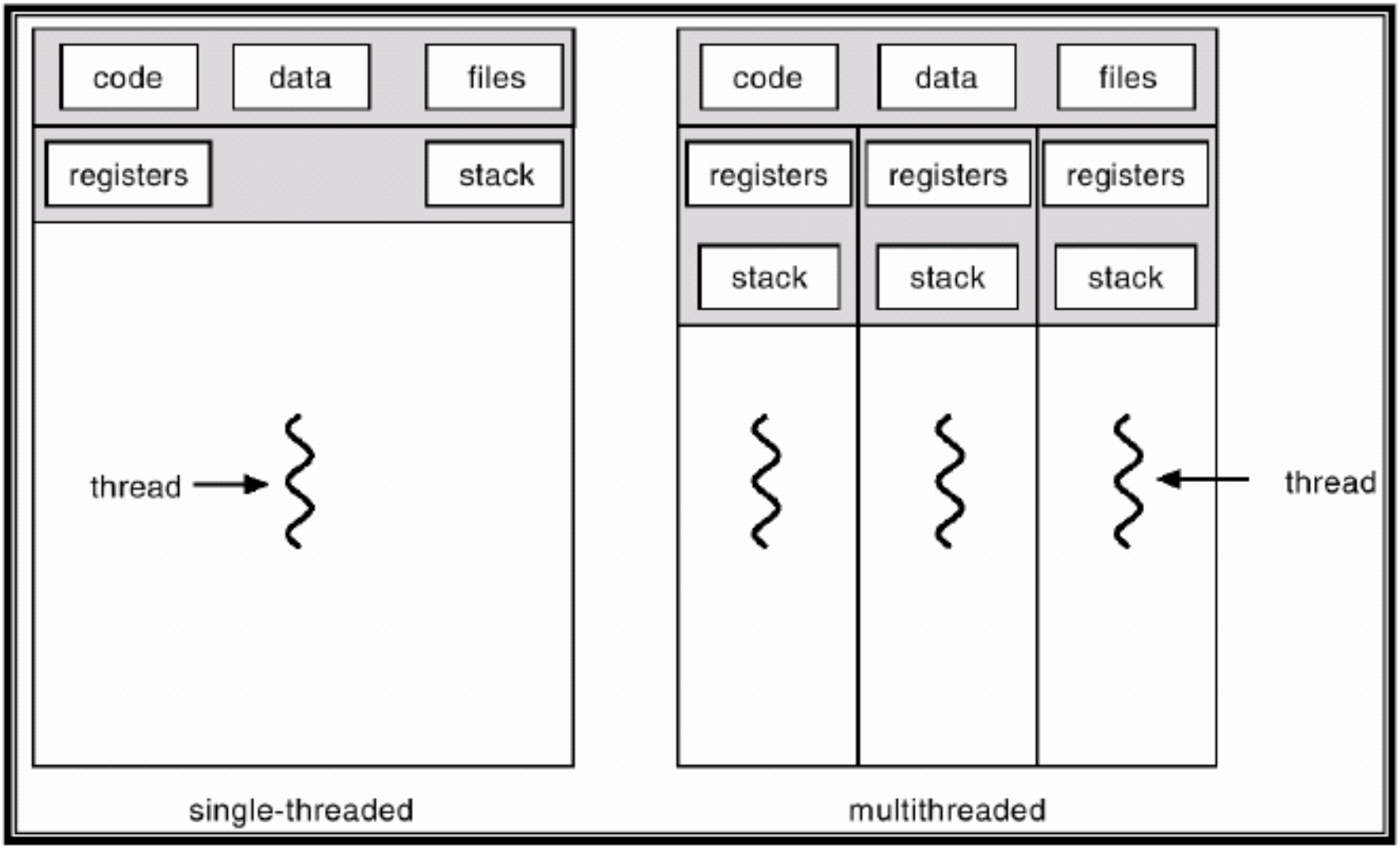
Process VS Thread



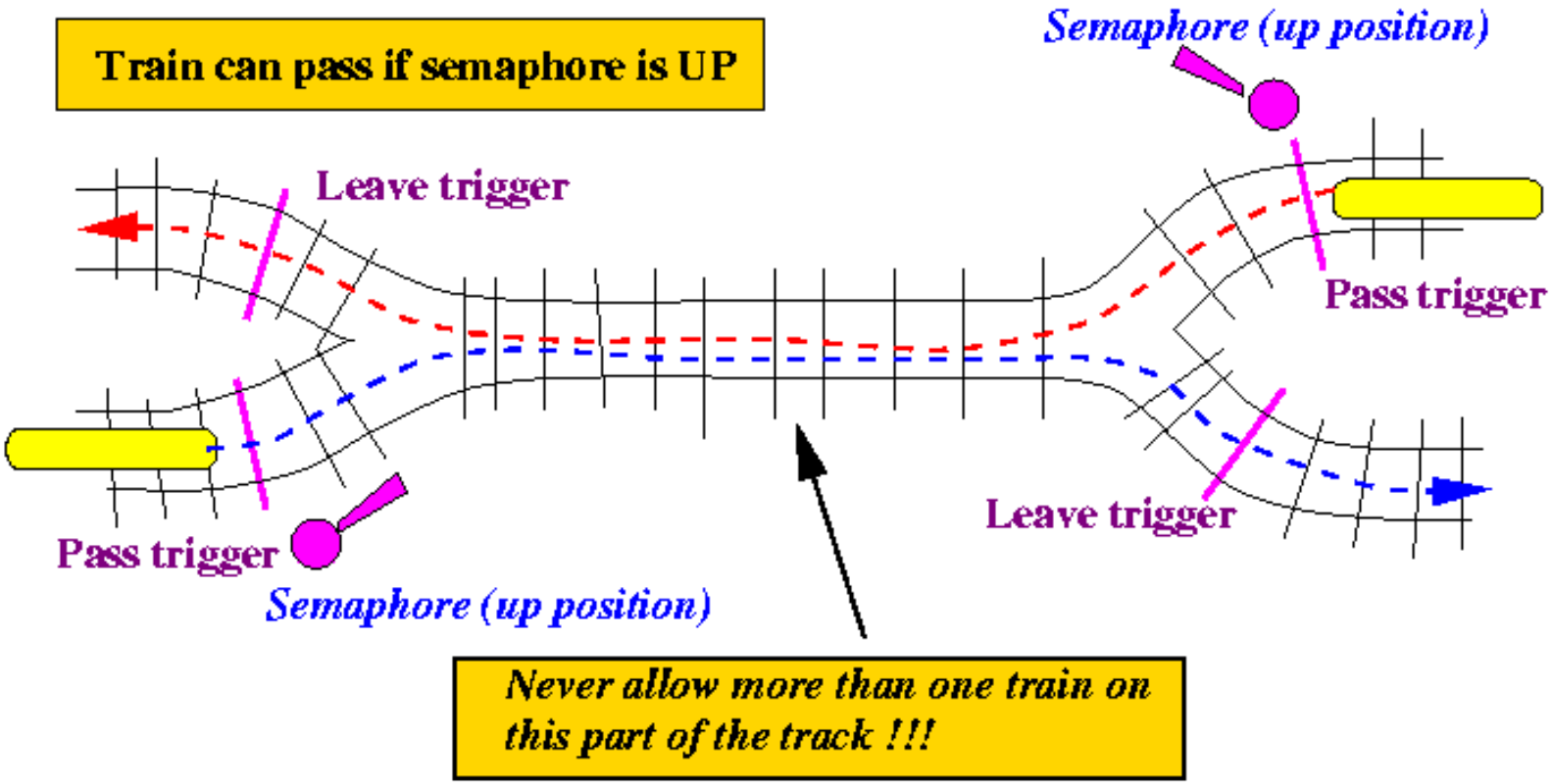
Process

- Parent
- Child
- Zombie

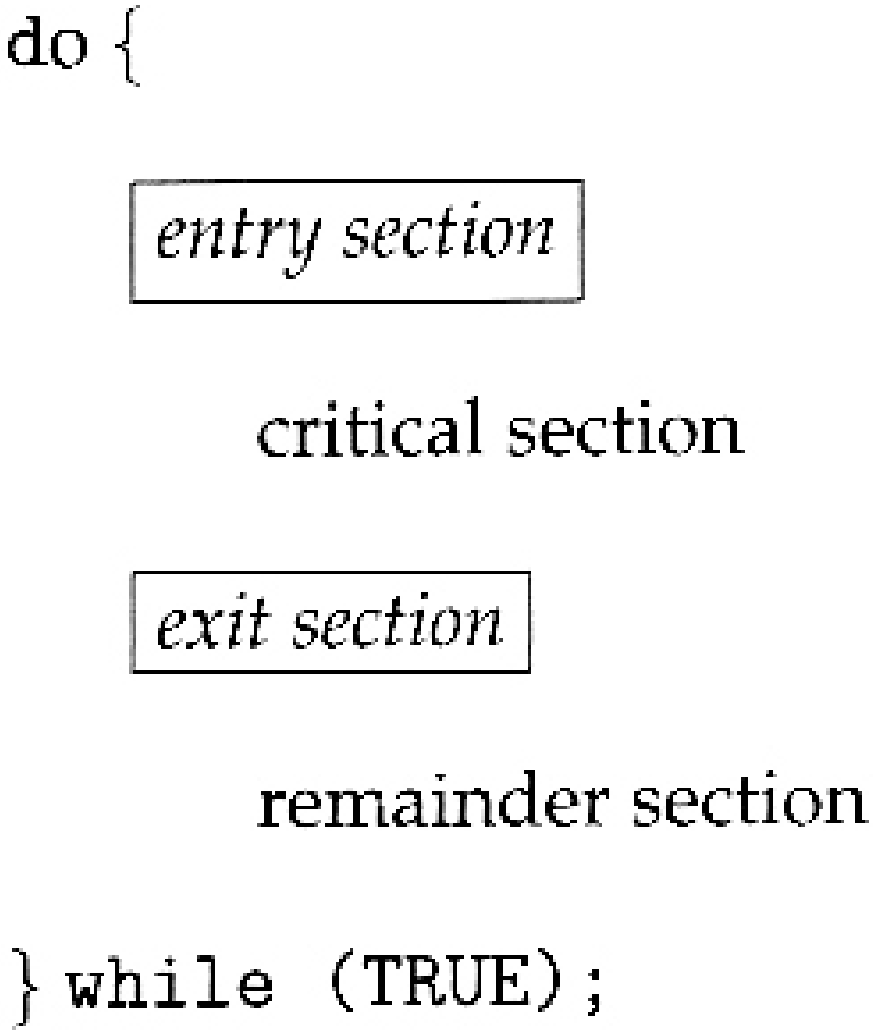
Process VS Thread



Syncronization



Syncronization



Questions

Вопросы?

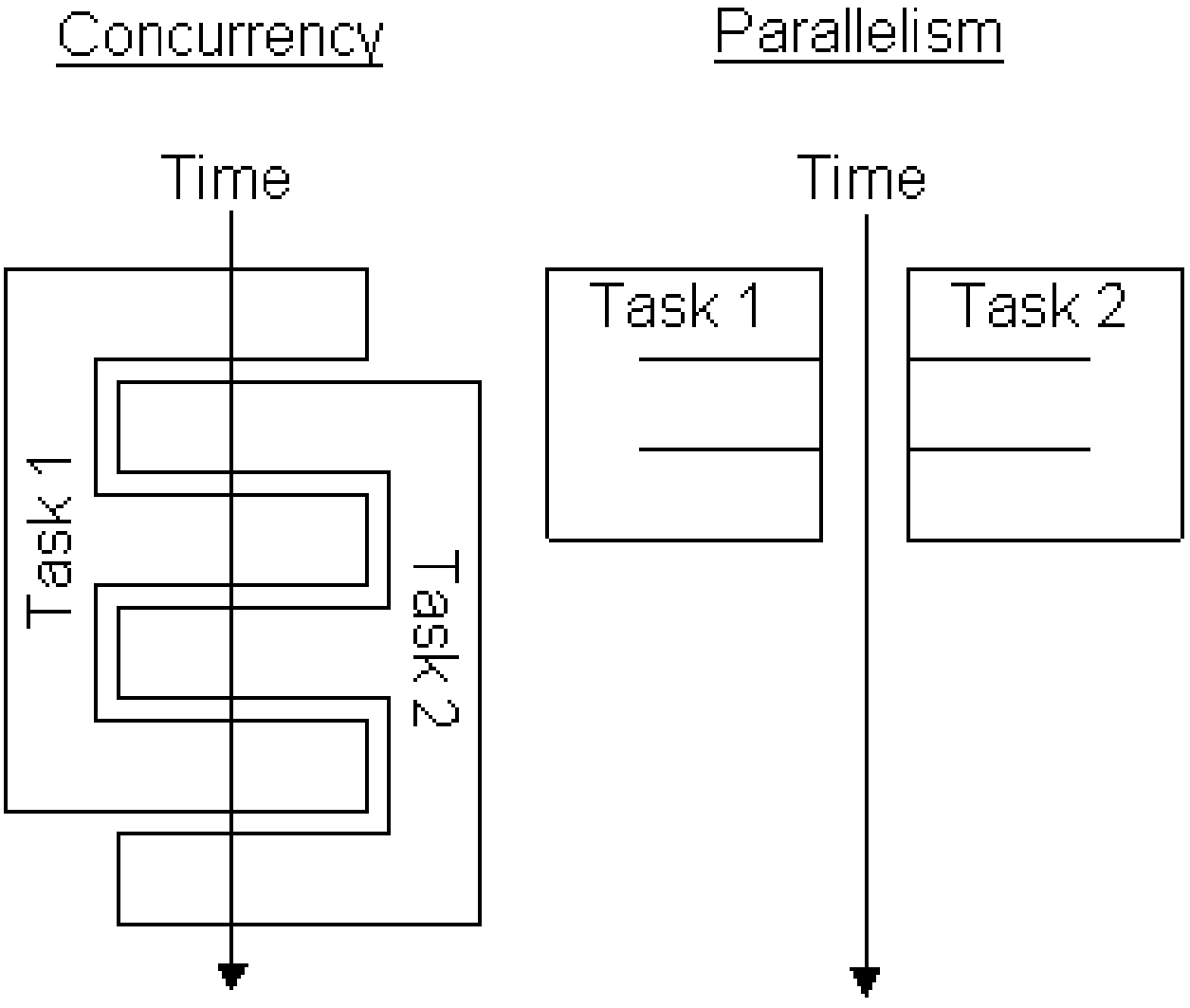
Process communication and sharing

- File
- Signals
- Shared memory
- Pipe
- Unix socket
- Socket

Questions

Вопросы?

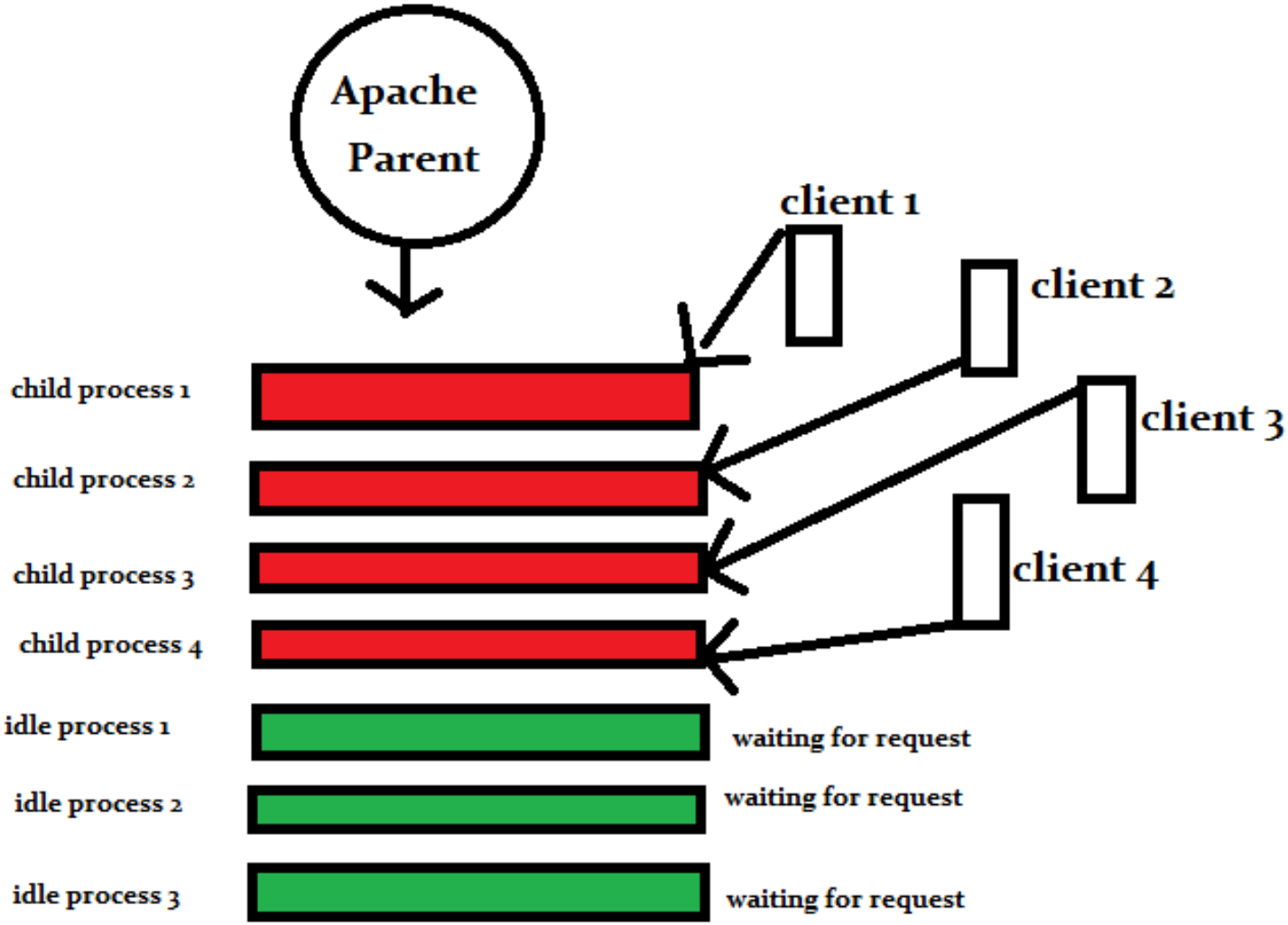
Concurrency



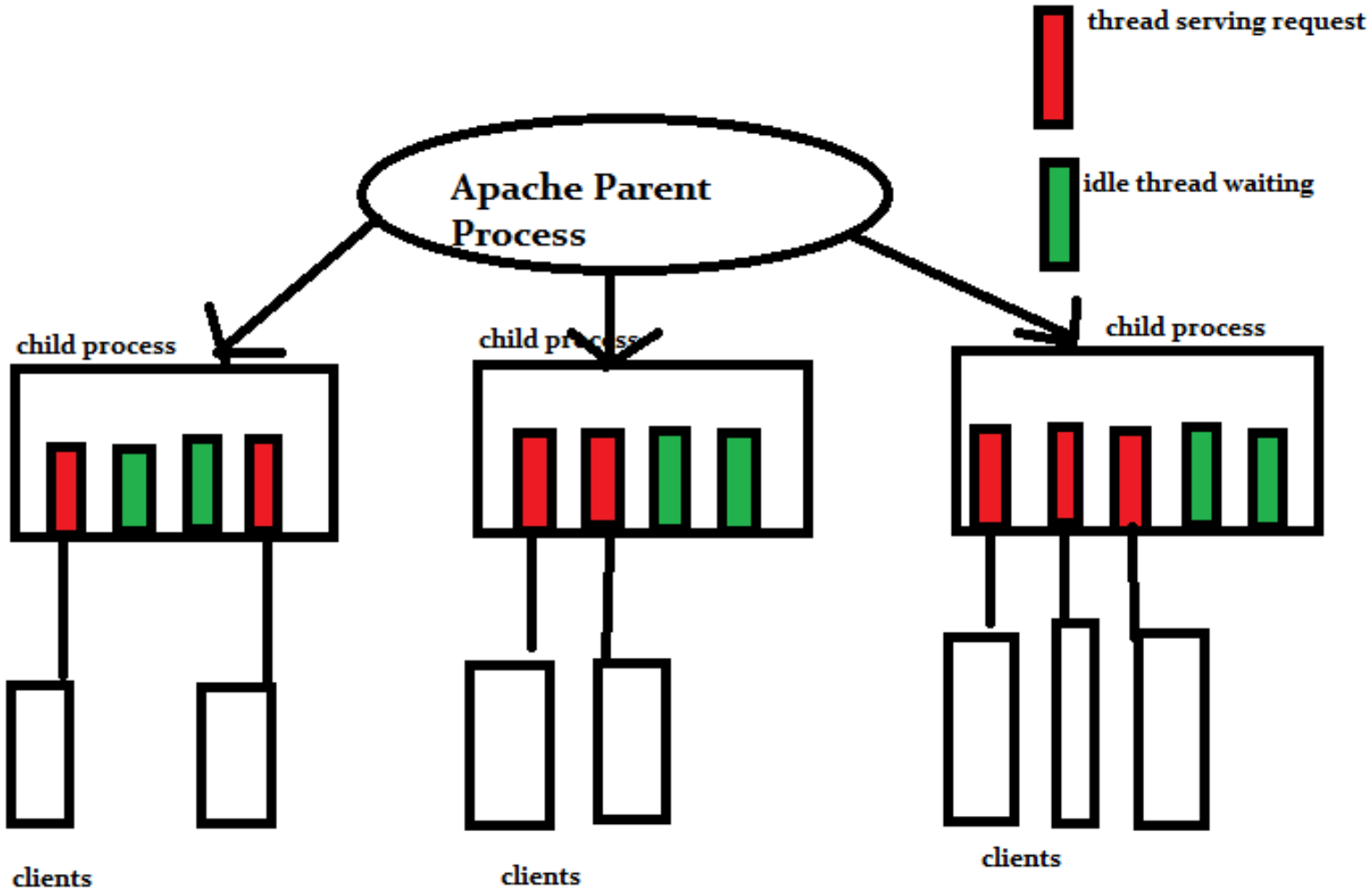
Process based architecture

Process\Thread per Request

Apache architecture



Apache architecture



Questions

Вопросы?

Process based

```
// Start a thread for each connection
while (1) {
  fd = accept();
  pthread_create(t2, start, fd);
}

start(int fd) {
  while (1) {
    char *data = do_netread(fd);   // NET_READING
    do_netwrite(fd, data);         // NET_WRITING
  }
}

char *do_netread(int fd) {
  return read(fd);
}

void do_netwrite(int fd, char *data) {
  write(fd, data);
}
```

Non blocking IO

```
state    s[N];              // clients' state field
int      fd[N], readfd[N];  // clients' file descriptors
char *data[N], *fdata[N]; // buffers holding clients' data

while (1) {
  if (fd = nb_accept())
    create state for new client, initialized to NET_READING;

  for (int i = 0; i < N; i++) {
    if (s[i] == NET_READING) {
      if (nb_read(fd[i], data[i]))
        s[i] = NET_WRITING;
    }

    if (s[i] == NET_WRITING) {
      if (nb_write(fd[i], fdata[i])
        s[i] = NET_READING;
    }
  }
}
```

Event based architecture

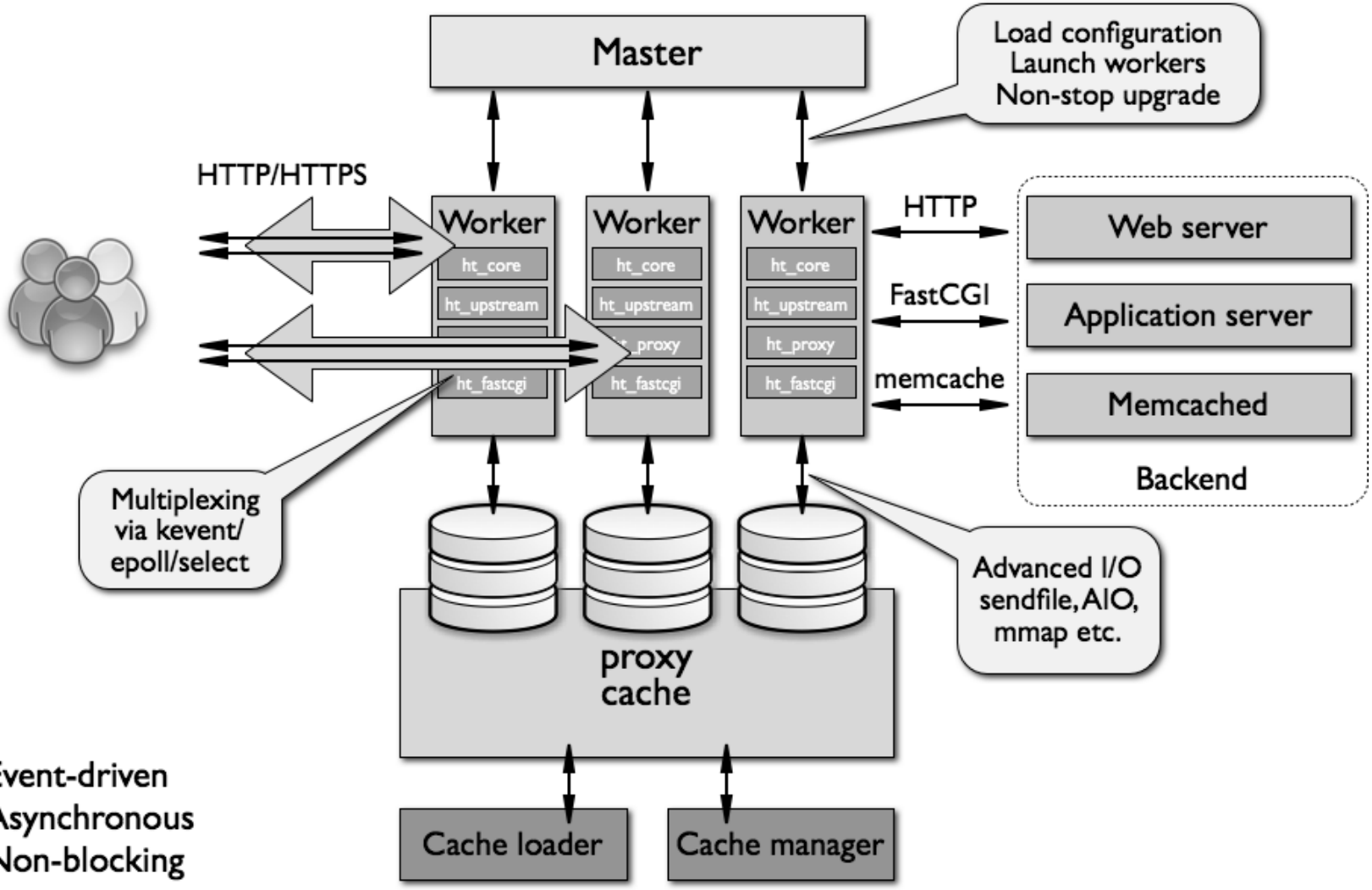
```
while (1) {
    (fd, event) = wait_for_next_event( fd_array );

    switch (event) {
        NET_ACCEPTABLE:
            (lookup_state, new_fd) = do_accept(fd);
            break;
        NET_WRITEABLE:
            do_netwrite(fd, lookup_state(fd));
            break;
        NET_READABLE:
            do_netread(fd, lookup_state(fd));
            break;
        NET_CLOSED:
            close(fd);
            break;
    }
}
```

Event based architecture

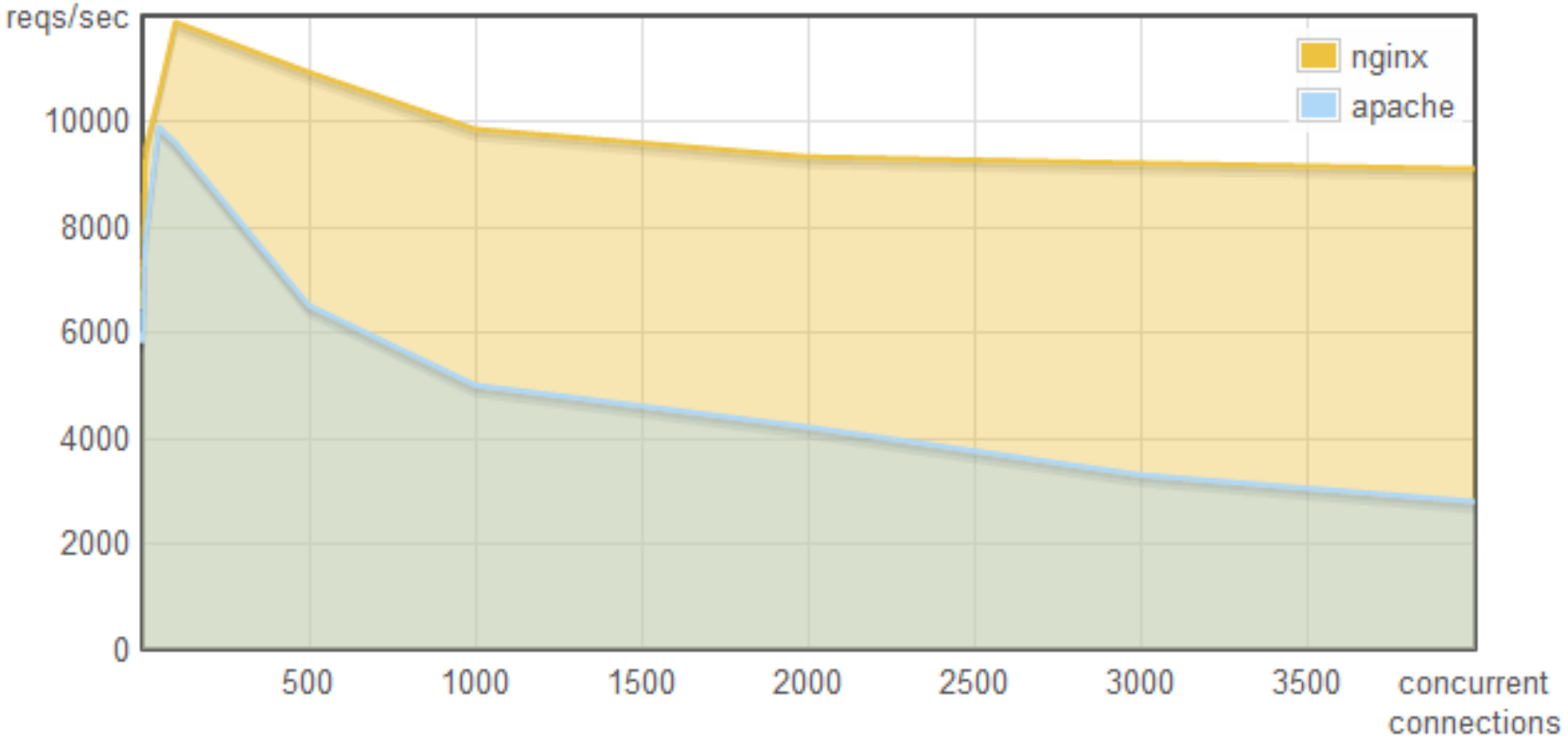
- Select/Poll - O(n)
- Epoll/Kqueue - O(1)

Nginx architecture



- Event-driven
- Asynchronous
- Non-blocking

Apache VS Nginx



Questions

Вопросы?

Web Server - CGI

```
1 Program first;
2 Begin
3   Writeln('Content-Type: text/plain');
4   Writeln;
5   Writeln('Hello, world!');
6 End.
```

Web Server - FastCGI

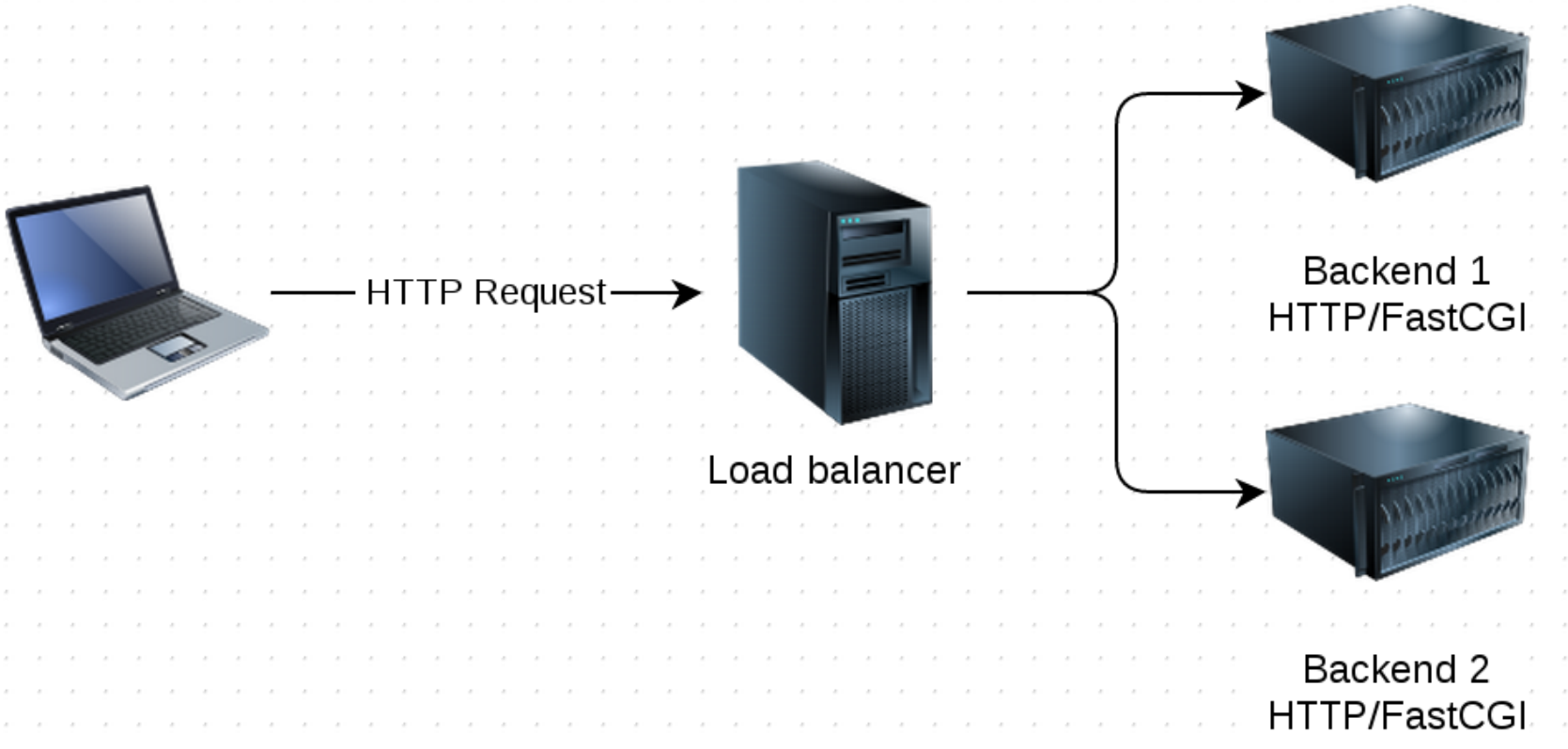
Fast Common Gateway Interface — дальнейшее развитие технологии CGI. По сравнению с CGI является более производительным и безопасным.

- Deamon
- TCP\Unix Socket
- Separeted user
- Separeted server

Questions

Вопросы?

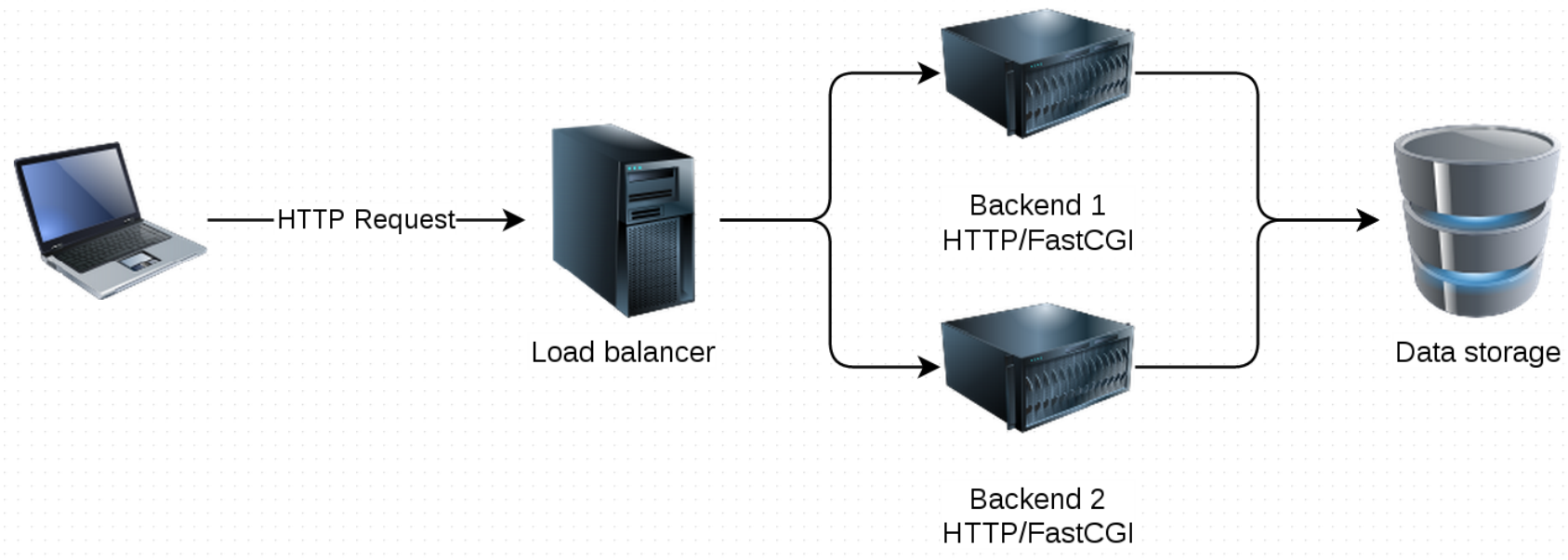
Load balancing



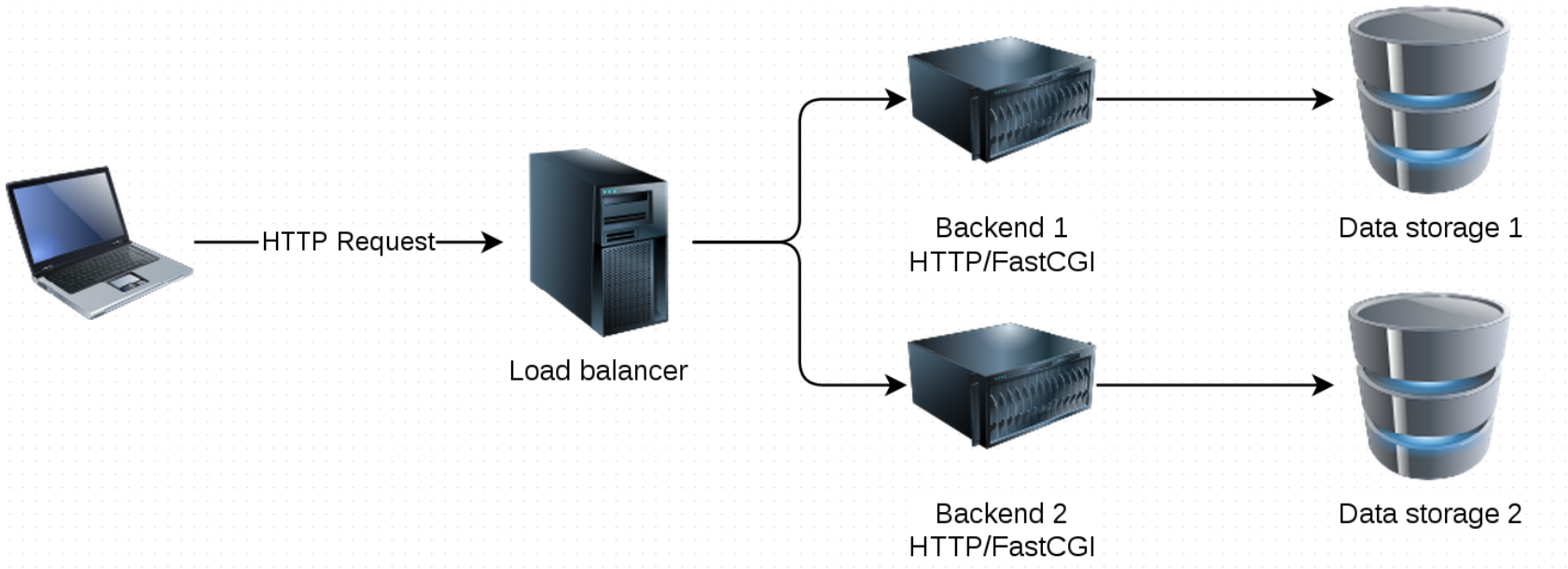
Load balancing

- Round-robin
- Weight round-robin
- IP sticky
- Sesion sticky
- Region sticky

Session share



Session sticky



Questions

Вопросы?