Многопоточность

Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность

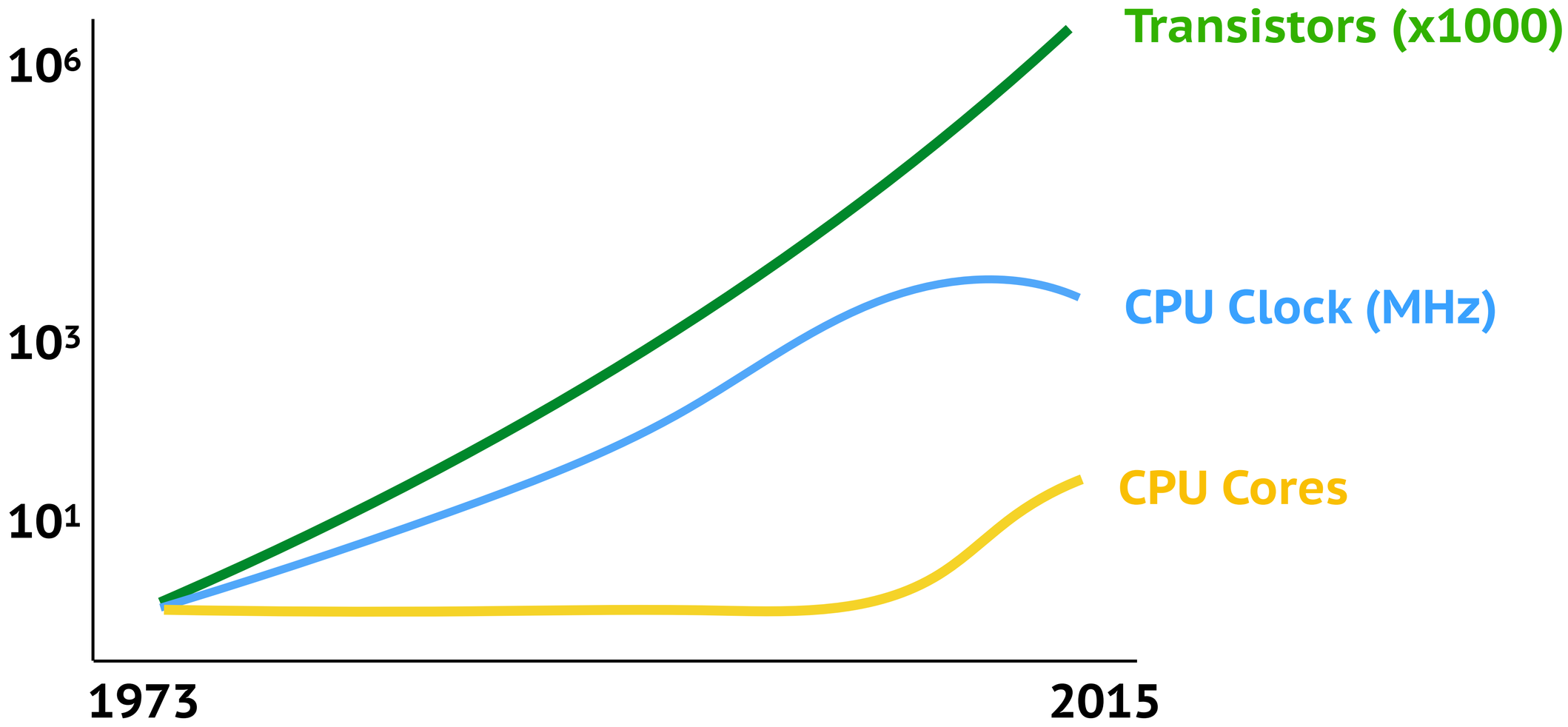
dev (nsk)

N1.RU
НЕДВИЖИМОСТЬ

Олег Федосеев

olegfedoseev
oleg.fedoseev@me.com

Закон Мура


Transistors (x1000)
CPU Clock (MHz)
CPU Cores
10⁶
10³
10¹
1973
2015

Современный мир параллелен

- 46, а то и **96 ядер** в ARM процессорах
- Многопользовательские системы и IoT
- Облака
- Распределённые системы

План

- Виды и подходы к “многопоточности”
- Проблемы “многопоточных” систем
- Особенности “многопоточности” в разных языках

Многопоточность
Многопоточность
Многопоточность
Многопоточность
Виды и подходы
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность

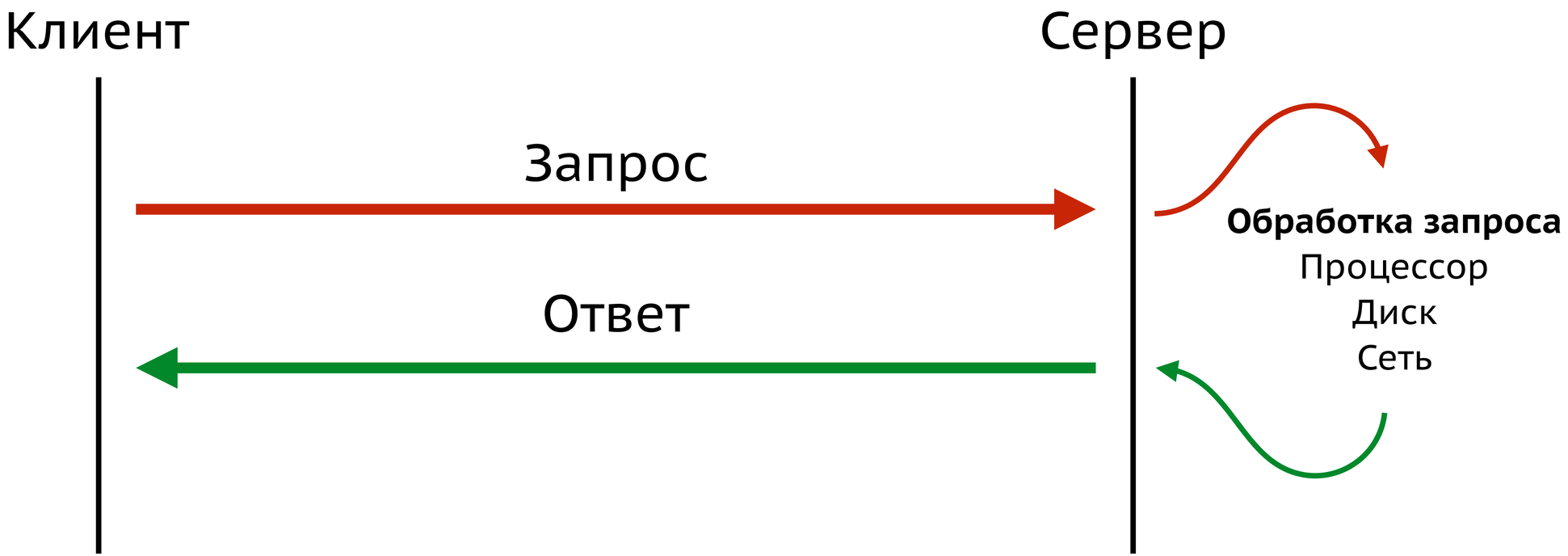
Виды “многопоточности”

- **Synchronous** (Синхронное)
- **Asynchronous** (Асинхронное)
- **Parallelism** (Параллельное)
- **Concurrency** (Конкурентное/одновременное)
- **Multithreading** (Многопоточное)

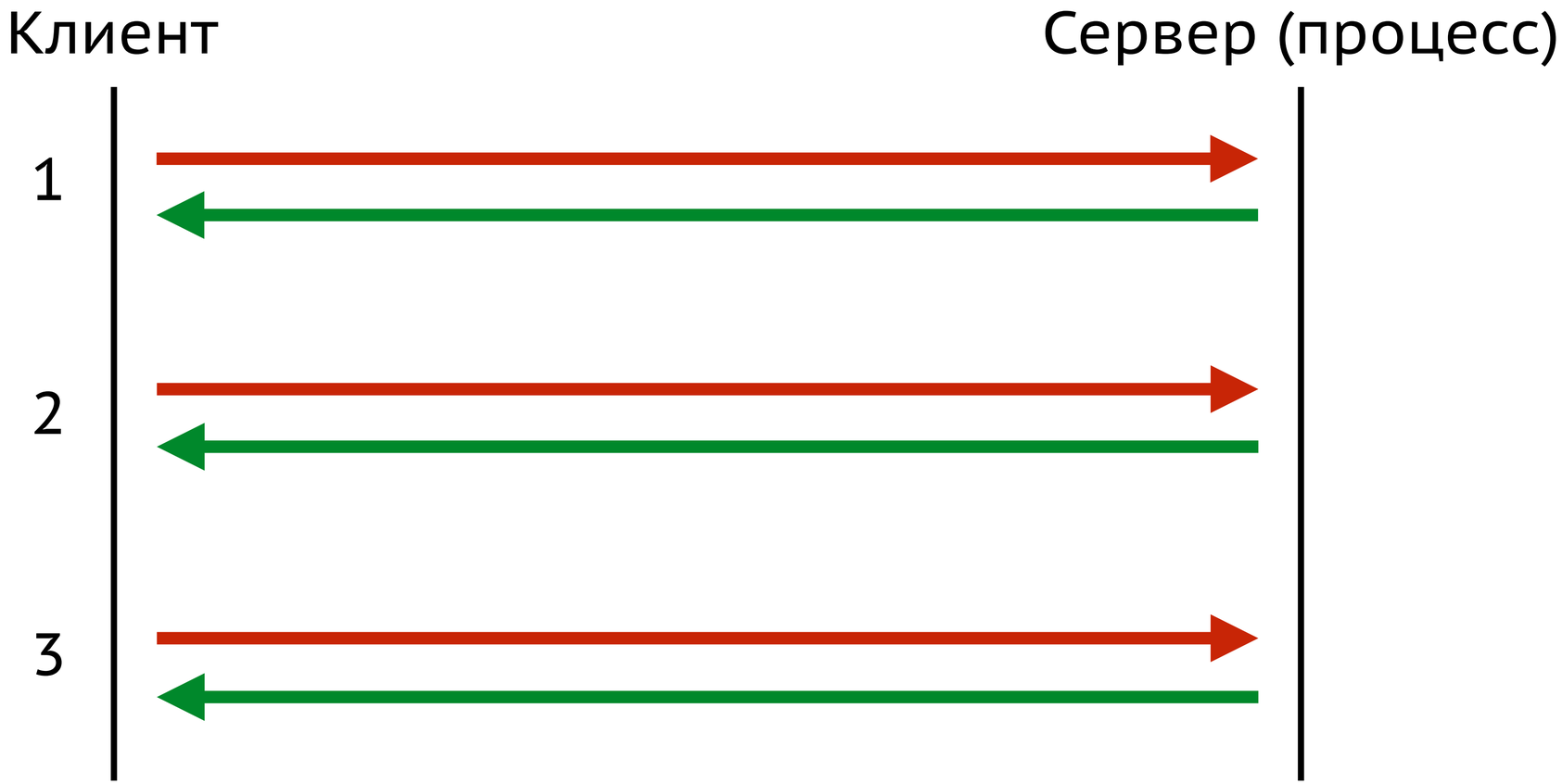
Виды “многопоточности”

- **Synchronous** (Синхронное)
- **Asynchronous** (Асинхронное)
- **Parallelism** (Параллельное)
- **Concurrency** (Конкурентное/одновременное)
- **Multithreading** (Многопоточное)

Пример - сетевой-сервис



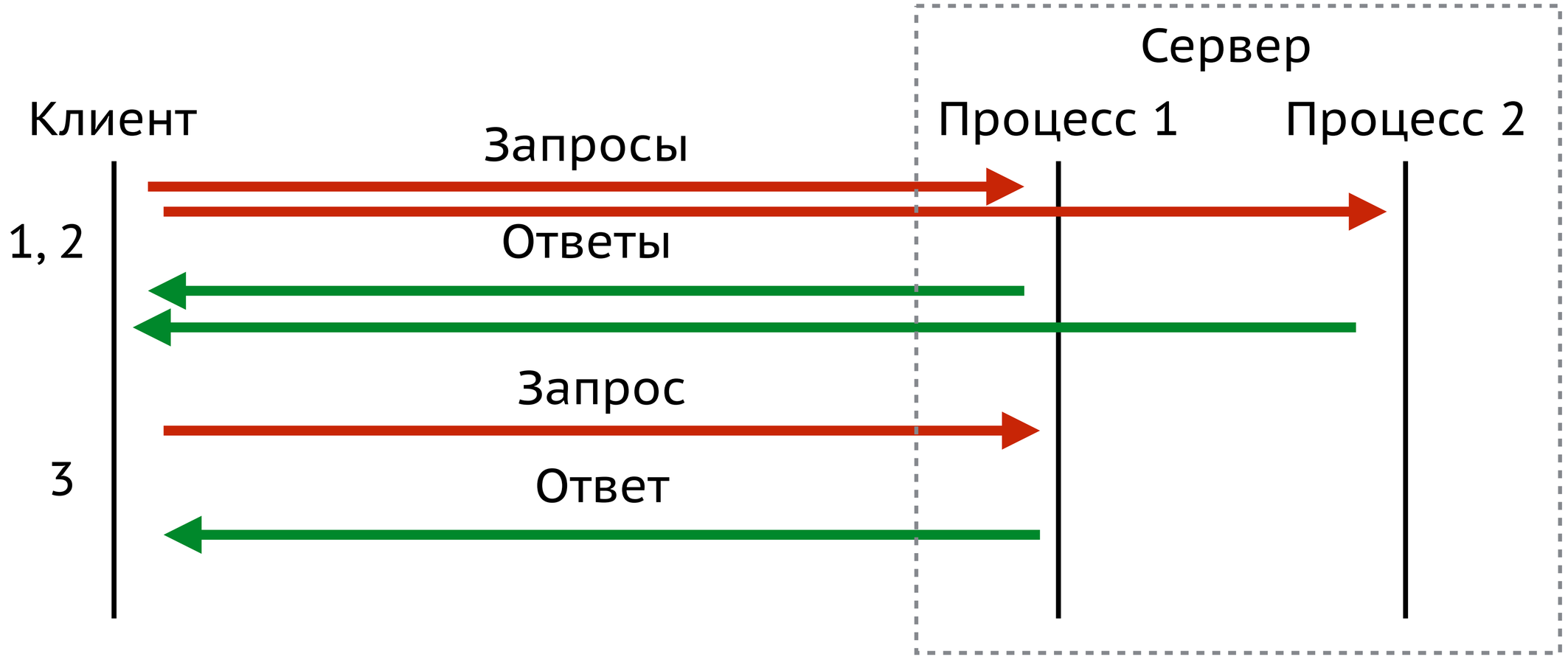
Синхронная обработка запросов



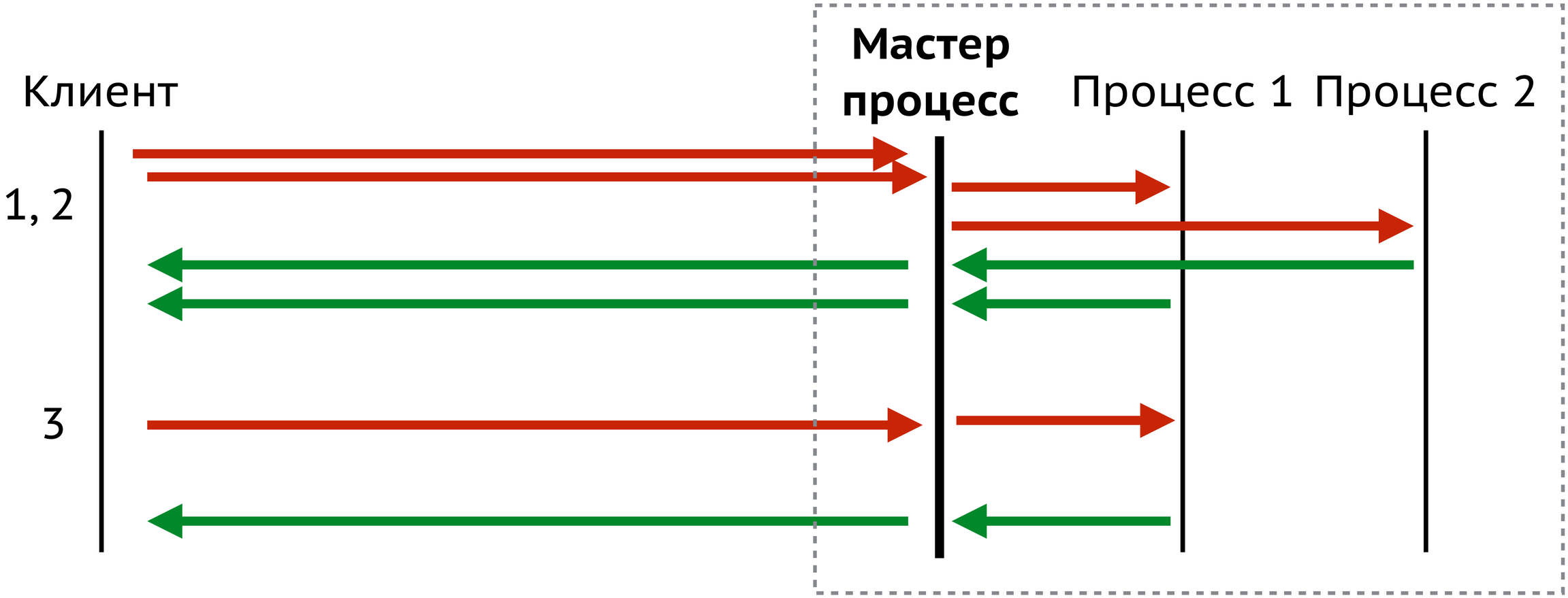
Синхронная обработка запросов

- Самый простой вариант
- Один запрос в единицу времени

1+1 = 3


Сервер
Клиент
Процесс 1
Процесс 2
Запросы
1, 2
Ответы
Запрос
3
Ответ

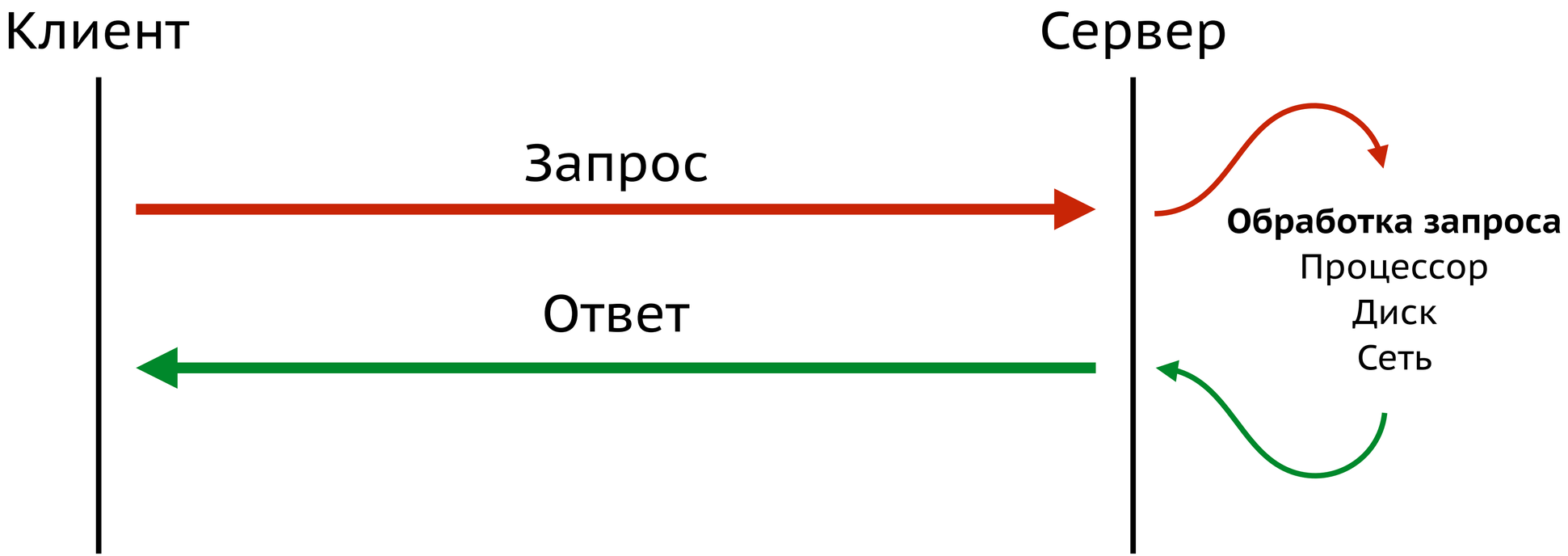
Многопроцессорная обработка



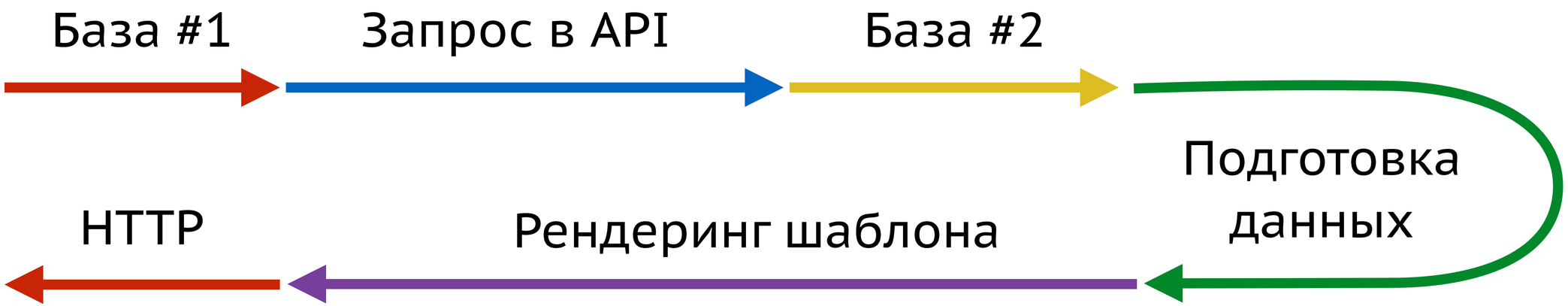
Многопроцессорная обработка

- Несколько синхронных процессов
- Независимые относительно друг друга
- Полностью параллельное выполнение на разных ядрах/процессорах
- Но нужен “мастер”

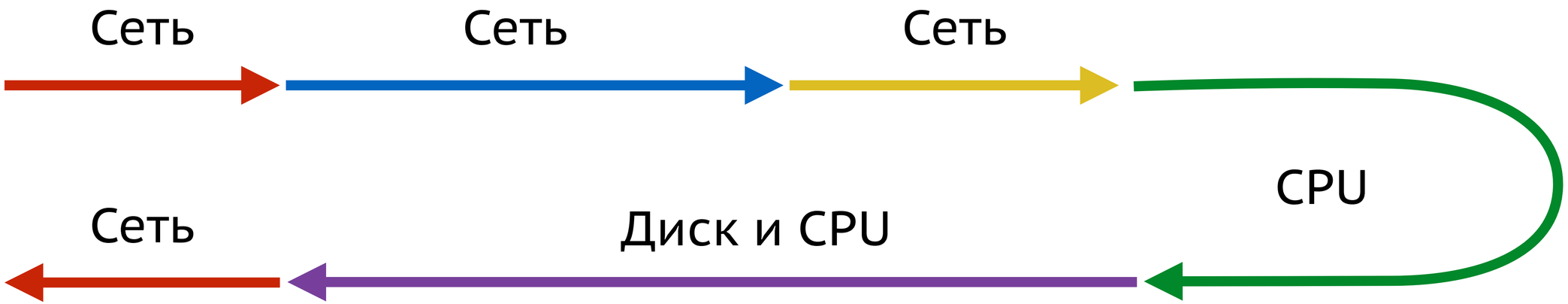
Конкурентная обработка запросов



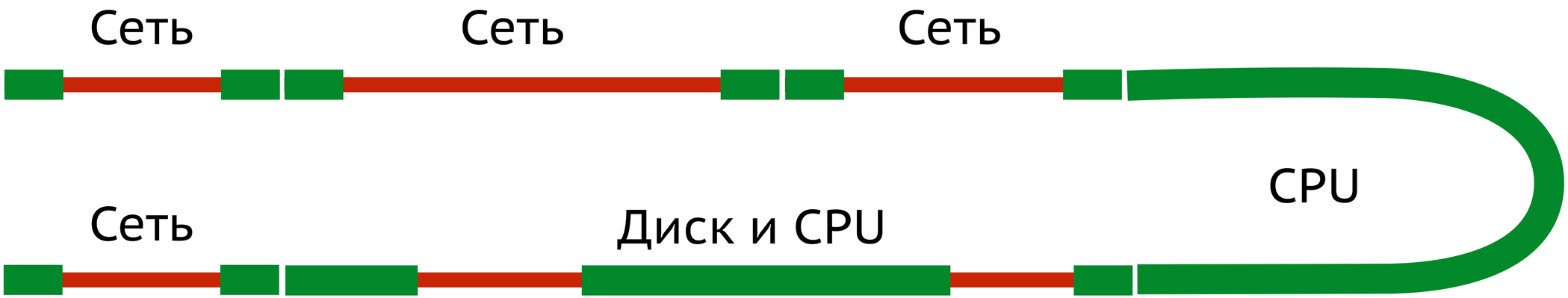
Обработка запроса



Обработка запроса



Обработка запроса

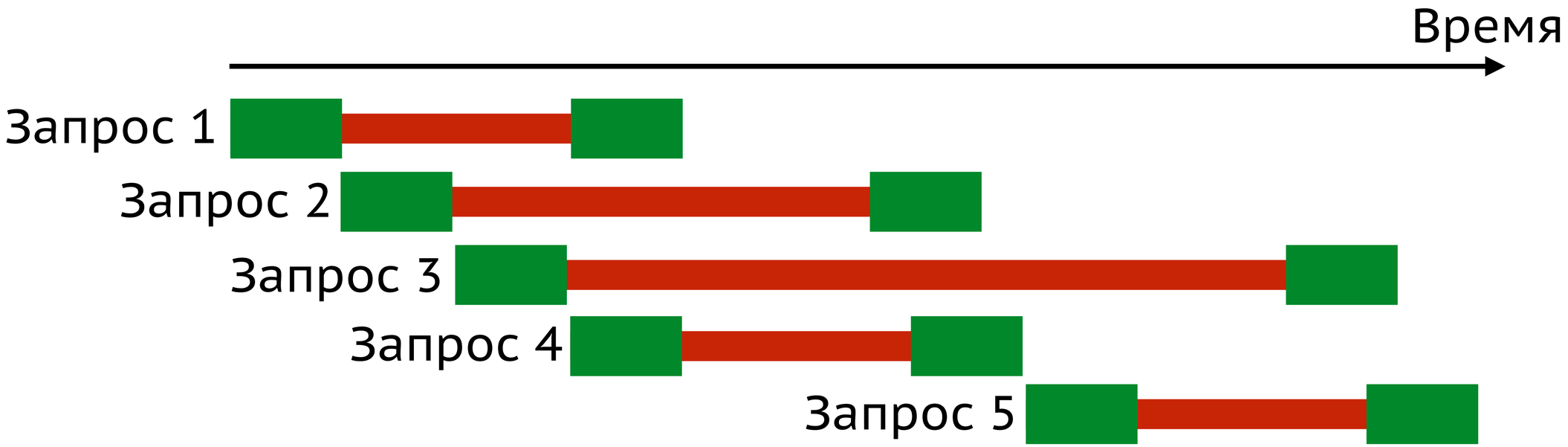


VS

- Event-driven
- kevent/epoll/select

- Process (Prefork) MPM
- Worker MPM
- Event MPM*

Конкурентная обработка запросов



Конкурентная обработка запросов

- Максимальная утилизация ресурсов в процессе
- Минимальное время простоя
- Эффект одновременной обработки нескольких запросов

Многопоточность
Многопоточность
Многопоточность
Многопоточность
Проблемы
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность

Синхронизация и координация

- Доступ к данным
- Выполнение операций
- Очерёдность

Подходы к синхронизации

- Mutex’ы
- Семафоры
- “Don’t communicate by sharing memory, share memory by communicating”

Mutex

```
var mutex = &sync.Mutex{}
var balance = float32

func Add(amount float64) {
  mutex.Lock()
  balance += amount
  mutex.Unlock()
}

func Display() string {
  mutex.Lock()
  current := balance
  mutex.Unlock()
  return fmt.Sprintf("Ваш текущий баланс %3.2f", current)
}
```

Mutex

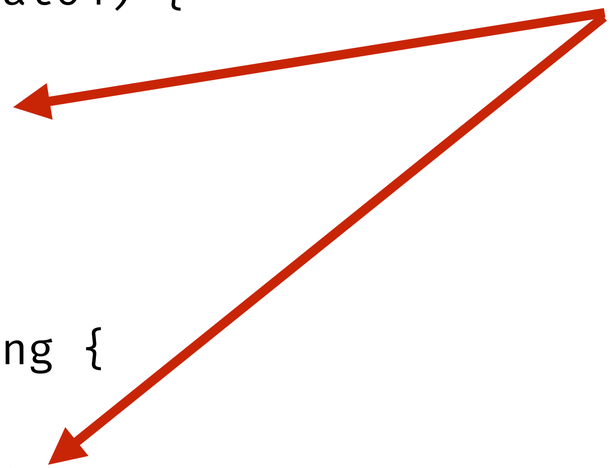
```
var mutex = &sync.Mutex{}
var balance = float32
```

Глобальное состояние (shared state)

```
func Add(amount float64) {
  mutex.Lock()
  balance += amount
  mutex.Unlock()
}

func Display() string {
  mutex.Lock()
  current := balance
  mutex.Unlock()
  return fmt.Sprintf("Ваш текущий баланс %3.2f", current)
}
```

Конкурентный доступ



Вытекающие проблемы

- Блокировки (dead lock)
- Состояние гонки (race condition, data race)
- Сложность отладки

Многопроцессорные проблемы

- Время на fork
- Синхронизация между процессами системы
- Масштабирование

Многопоточность
Многопоточность
Многопоточность
Многопоточность
Особенности
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность
Многопоточность

Особенности в разных языках

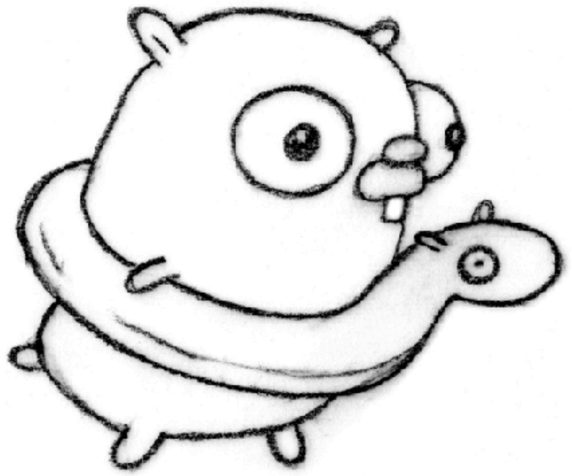
- Python (GIL, green threads, processes, threads)
- PHP-FPM (processes)
- Go (CSP, goroutines on threads)
- Erlang (CSP)
- Java (processes, threads, green threads)
- Node (event loop)

Выводы

- Помните про особенности реализации в вашем языке/платформе
- Асинхронное сложнее синхронного, не всегда это нужно
- Даже если вы явно не пишите асинхронный код, знать про это нужно
- Попробуйте Go! :)

Ссылки

- https://golang.org/s/concurrency-is-not-parallelism
- https://talks.golang.org/2012/concurrency.slide
- https://talks.golang.org/2013/advconc.slide



dev (nsk)

N1.RU
НЕДВИЖИМОСТЬ

Вопросы?

Олег Федосеев
olegfedoseev
oleg.fedoseev@me.com