Программная инженерия, лекция 7
Асинхронный ввод/вывод и DHT

Пожидаев М. С.

17 октября 2022 г.

IO-bound

Варианты реализации конкурентной системы

1. *Синхронный ввод/вывод:* программа разбивается на потоки, каждый из которых ждёт завершения операций ввода/вывода.
2. *Асинхронный ввод/вывод:* всё выполнение происходлит в одном потоке без ожидания завершения операций ввода/вывода, которые выполняются в фоновых потоках.

Событийная модель

1. Всё приложение разбивается на множество событий как можно меньшего размера.
2. События выполняются в одном общем потоке под управленнием главного цикла (event loop).
3. Операция ввода-вывода обычно разбивается на событие, её запускающее, и событие, обрабатывающее её результат.

Преимущества:

- можно не заботится о согласовании потоков;
- возрастает общая эффективность исполнения.

Промисы

И промисифайзеры

Промис (promise) — заготовленный объект в JavaScript, задающий порядок отложенного выполнения кода. Может быть в одном из трёх состояний:

- pending;
- fullfilled;
- rejected.

Последующие действия задаются функцией (then()), которая может быть вызвана несколько раз (chaining).

Недостатки

Минусы асинхронного программирования

1. Нельзя смешивать блокирующие и неблокирующие вызовы.
2. Плохо подходит для задач с большой вычислительной нагрузкой (не IO-bound).

async

Ключевое слово async

Ключевое слово `async` используется при объявлении функции и означает, что:

1. Вызывающая функция может не дожидаться выполнения функции с ключевым словом `async`.
2. Функция автоматически возвращает промис.
3. Внутри функции можно использовать ключевое слово `await`.

ƒawait

Ключевое слово await

Ключевое слово `await` используется для вызова функции, возвращающей в качестве результата промис. При вызове:

1. Текущее исполнение функции прерывается, поток заполняется исполнением других событий в очереди.
2. Работа будет продолжена, когда промис перейдёт в состояние fullfilled.
3. Результат выполнения промиса будет возвращён как значение функции.

Пример

Пример асинхронной функции

```
this.objDir = async (req, res, obj)=>{
  if (this.emptyStr(obj) || obj.trim().length < 8) {
    errors.internal(req, res);
    return;
  }
  const path = obj.trim();
  await fs.mkdirAsync(path, {recursive: true});
};
```

Архитектура

Когда применять асинхронный ввод/вывод

1. Высокая конкурентность запросов, при которой традиционная архитектура с использованием мьютексов и семафоров может оказаться сложной и запутанной.
2. Доминирование операций ввода/вывода над вычислениями.
3. Отсутствие необходимости смешивать код с традиционном кодом на основе блокирующего ввода/вывода.

DHT

Распределённая хеш-таблица

DHT является примером одноранговой системы со следующими ключевыми характеристиками:

1. Практическое отсутствие ограничений масштабирования.
2. Выдающаяся отказоустойчивость, практически недостижимая в клиент-серверной и микросервисной архитектурах.

Обычно применяется для отказоустойчивого хранения информации (распределённые СУБД, пиринговые системы и пр.).

Ключи записей

Формирование адресного пространства

Область близости

Пространство ключей позволяет формировать множество близлежащих узлов, логическое расстояние до которых невелико.

Логическое и физическое пространство

Множество логических адресов отображается во множество физических, причём близлежащие логические узлы могут находиться далеко друг от друга в физическом пространстве.

План Б

Дублирование и восстановление записей

Запись

В процессе записи данных в DHT данные записываются не только на узел с нужным ключом, но сразу же и на близлежащие узлы (BitTorrent делает это в процессе скачивания).

Чтение

В процессе чтения при недоступности необходимого узла требуется проверить близлежащие узлы, которые могут подсказать, где искать данные.

CAD

Content addressable storage

Контент вместо адреса

Для построения адреса хранимого блока можно использовать содержимое блока после обработки хеш-функцией.

Свойства:

- неизменяемость данных;
- изменение блока подразумевает получение нового адреса (хотя старый блок можно не удалять);
- гармоничное сочетание с DHT.

Архитектура

Когда применять DHT

1. Требуется особая отказоустойчивость без точек отказа.
2. Существует возможность формирования пространства ключей.
3. Система ориентирована на хранение информации.
4. Доступно большое количество вычислительных ресурсов. Их количество важнее, чем уровень ожидаемой доступности.

Спасибо за внимание!

Всё о курсе: https://marigostra.ru/materials/engineering.html
E-mail: msp@luwrain.org
Канал в Телеграм: https://t.me/MarigostraRu