Princípy tvorby softvéru
Concurency and Paralelism

Robert Lukoťka
lukotka@dcs.fmph.uniba.sk
www.dcs.fmph.uniba.sk/~lukotka

M-255

Implementation

- **Concurency** - refers to the ability of different parts or units of a program, algorithm, or problem to be executed out-of-order or in partial order, without affecting the final outcome.
- **Paralelism** - schopnosť robiť výpočty súčastne.

Race conditions

What could possibly go wrong?

```
int etx_rcvd = FALSE;
void WaitForInterrupt()
{
    etx_rcvd = FALSE;
    while (!ext_rcvd)
    {
        counter++;
    }
}
```

Race conditions

What could possibly go wrong?

```
int etx_rcvd = FALSE;
void WaitForInterrupt()
{
    etx_rcvd = FALSE;
    while (!ext_rcvd)
    {
        counter++;
    }
}
```

Race conditions

Preložené na

```
int etx_rcvd = FALSE;
void WaitForInterrupt()
{
    while (1)
    {
        counter++;
    }
}
```

Race conditions

- Mutual exclusion (mutex, synchronized, locks, . . .)
- Zabrániť optimalizáciám (volatile, . . .)

Therac 25 - technicky, zdrojom problému bola race condition.

Concurrency konštrukty

- Python - Threading
- Java - BlockingQueue
- ThreadPools
- Event loops

Race conditions

Programovať s Lockami je ťažké

- Race conditions sa netýkajú len pamäte: Shared output devices
- Performance issues
- Race Conditions
- Deadlocks,
- Nechcete aktívne čakať? Spomínate na systémové volanie select?

Concurrent computing

Good practices:

- Minimal locks (čas aj priestor)
- Prefer higher level constructs
- Local variables
- Immutable types
- Pure functions
- Získať locky na začiatku v abecednom poradí
- **minimalizovať používanie lockov**

Príklad

Double Checked Locking

Čo ak potrebujeme lock na dlhšie?

Najmä, ak je scope locku priveľký:

- lock + uloženie stavu, dlhá operácia, lock + kontrola zmeny

Porobne ako prechádzame dlhým DB transakciám.

Architectonické / dizajové riešenia

- Nechať to na databázu
  - RESTfull api + stateless server
- Vytvoriť single threaded bubliny.
  - BlockingQueue
  - Reactor
- Immutable datové štruktúry - umožňujú bezpečné čítanie a atomické zmeny.

Immutable data types

+ No concurrency issues while reading.
+ Prepared for parallel algorithms.
+ Easy to detect if stuff changed.
+ Atomické zmeny stavu.
  - Pomalšie
  - Pravdepodobne potrebujeme garbage collector
  - Problém so syntaxou.

Asynchronous computing

- Futures
- Promises
- Callbacks

Ďalšie zdroje

- Concurrency - Wikipedia
- Parallel Computing - Wikipedia
- Race Condition - Wikipedia