EDA040: Concurrent Programming

Michael Noukhovitch

Fall 2015, Lund University

Notes written from Klas Nilsson's lectures.

Contents

1 Introduction

1.1 Concurrency

activity entity performing actions

process entity performing instructions with own resources

job sequential instructions to be performed by an activity

task a set of jobs being performed by some process

thread sequential activity performing instructions

1.2 Threads

execution thread the thread itself accessed via the `Thread` interface

2 Mutual Exclusion

2.1 Semaphores

semaphore simple counting interface for concurrency

2.1.1 Mutex Semaphore

used to lock and unlock critical sections

```
MutexSem mutex;
mutex.take()
// critical section
mutex.give()
```

2.1.2 Signaling

calls used to block or unblock a thread

```
CountingSem mutex = new CountingSem();
// thread A
*
mutex.take() // block this thread
*
// thread B
*
mutex.give() // unblock thread A
*
```

2.2 Mutex

must provide:

- mutual exclusion
- no deadlock
- no starvation
- efficiency