

# Monitores com uma variável de condição

Grupo de Sistemas Distribuídos  
Universidade do Minho

## Objectivos

Uso das facilidades de biblioteca em Java relativas a variáveis de condição, para resolver problemas com monitores que necessitem apenas de uma variável de condição.

## Mecanismos

Variáveis de condição associadas a `ReentrantLock`:

- `interface Lock`: método `newCondition()`;
- `interface Condition`: métodos `await()`, `signal()`, `signalAll()`;

## Exercícios propostos

- 1 Escreva uma abstracção para permitir que  $N$  threads se sincronizem:

```
class Barrier {  
    Barrier (int N) { ... }  
    void await() throws InterruptedException { ... }  
}
```

A operação `await` deverá bloquear até que as  $N$  threads o tenham invocado; nesse momento o método deverá retornar em cada thread. Escreva duas versões desta abstracção:

1. Suponha que cada thread apenas vai invocar `await` uma vez sobre o objecto.
2. Modifique a implementação para permitir que a operação possa ser usada várias vezes por cada thread (barreira reutilizável), de modo a suportar a sincronização no fim de cada uma de várias fases de computação.

## Exercícios Adicionais

2 Generalize a abstracção de barreira para uma abstracção de Agreement:

```
class Agreement {  
    Agreement (int N) { ... }  
    int propose(int choice) throws InterruptedException { ... }  
}
```

Esta deve permitir que as  $N$  threads se sincronizem para chegar a acordo num valor. Cada thread propõe um valor, ficando o método `propose` bloqueado até todas a  $N$  o terem feito; nesse momento o método deverá retornar em cada thread o máximo dos valores propostos. Tal como na barreira reutilizável, deverá ser possível uma sucessão de acordos.