Sistemas Distribuídos

Universidade do Minho

2011/2012

- Exclusão mútua: garantir que dois processos ou threads não tenham acesso simultaneamente a um recurso partilhado.
- Proteger secções críticas do código.

Exclusão Mútua

```
synchronized void metodo(){
...
}
```

Exclusão Mútua explícita

```
import
java.util.concurrent.locks.Reentran
tLock;

lock = new ReentrantLock();

void metodo(){
    this.lock.lock();
    try{
        ...
    } catch (InterruptedException e) {
        e.printStackTrace();
        }
    finally{
        this.lock.unlock();
        }
}
```

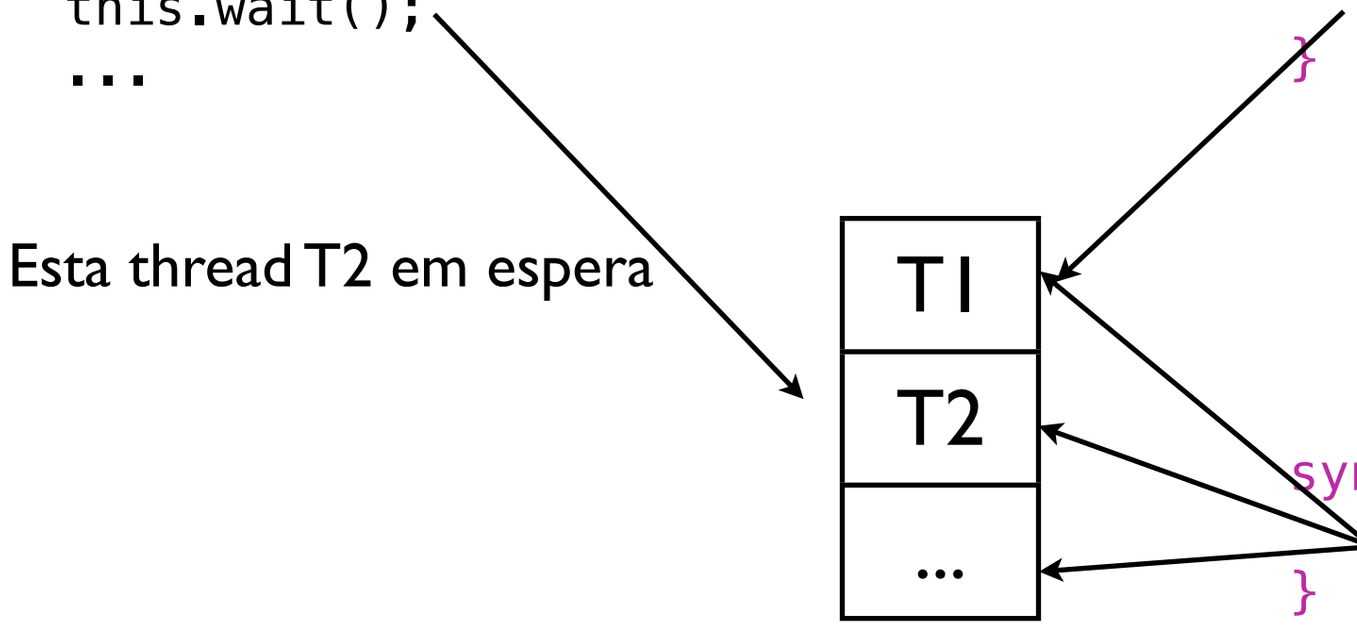
- Variáveis de condição:
  - suspensão/retoma de execução dentro de zona crítica;
  - mecanismo intrínseco de variável de condição em todos os objectos;
  - métodos wait(), notify(), notifyAll();

- Exemplo:

```
synchronized void metodo(){
...
while(condição){
   this.wait();
   ...
}
```

```
synchronized void metodo2(){
   this.notify();
}
```

```
synchronized void metodo3(){
   this.notifyAll();
}
```



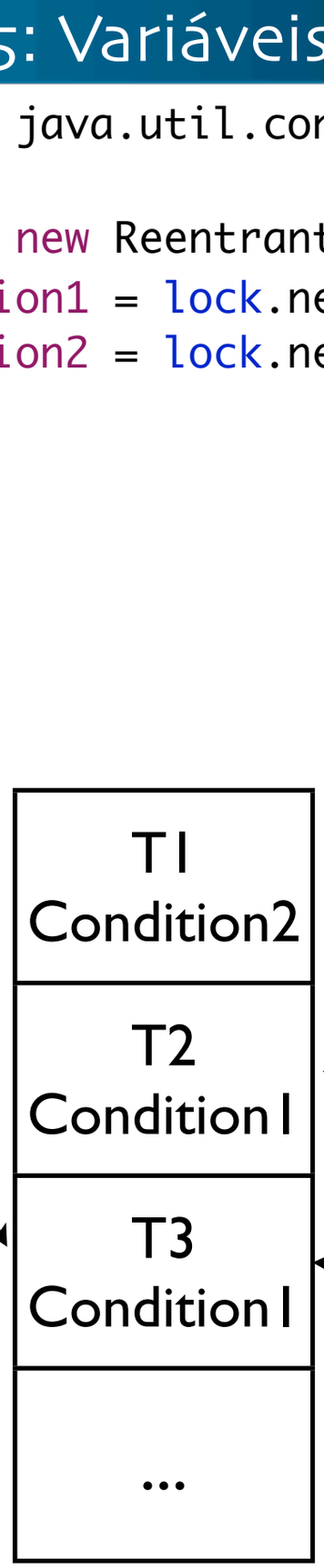
- Variáveis de condição:
  - suspensão/retoma de execução dentro de zona crítica;
  - usada em conjugação com o ReentrantLock();
  - java.util.concurrent.locks.ReentrantLock
    – métodos lock(), unlock(), newCondition();
  - java.util.concurrent.locks.Condition
    – métodos relevantes: await(), signal(), signalAll()

```
import java.util.concurrent.locks.Condition;

lock = new ReentrantLock();
condition1 = lock.newCondition();
condition2 = lock.newCondition();
```

- Exemplo:

```
void metodo(){

lock.lock();
try{
while(condição){
    condition1.await();
    ...
}
finally{
 lock.unlock();
}
```

T3 em espera na condição 1




```
void metodo2(){
   lock.lock()
    try{
    ...
    condition2.signal();
    finally{
     lock.unlock();
    }
 }

void metodo3(){
   lock.lock();
   try{
   ...
   condition1.signalAll();
   finally{
    lock.unlock();
   }
}
```

Aula 5: Exercicio 2

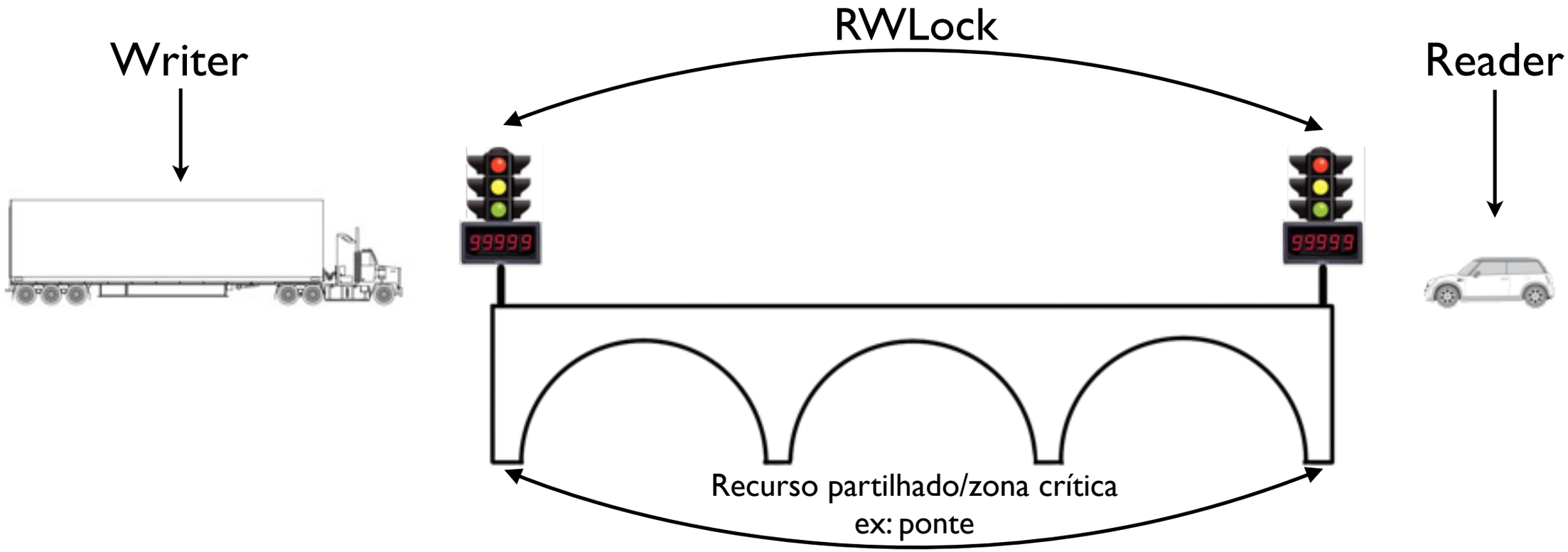
- Implemente a classe RWLock com os métodos readLock(), readUnlock(), writeLock() e writeUnlock() de modo a permitir o acesso simultâneo de múltiplos leitores a uma dada região crítica, ou em alternativa, o acesso de um único escritor.

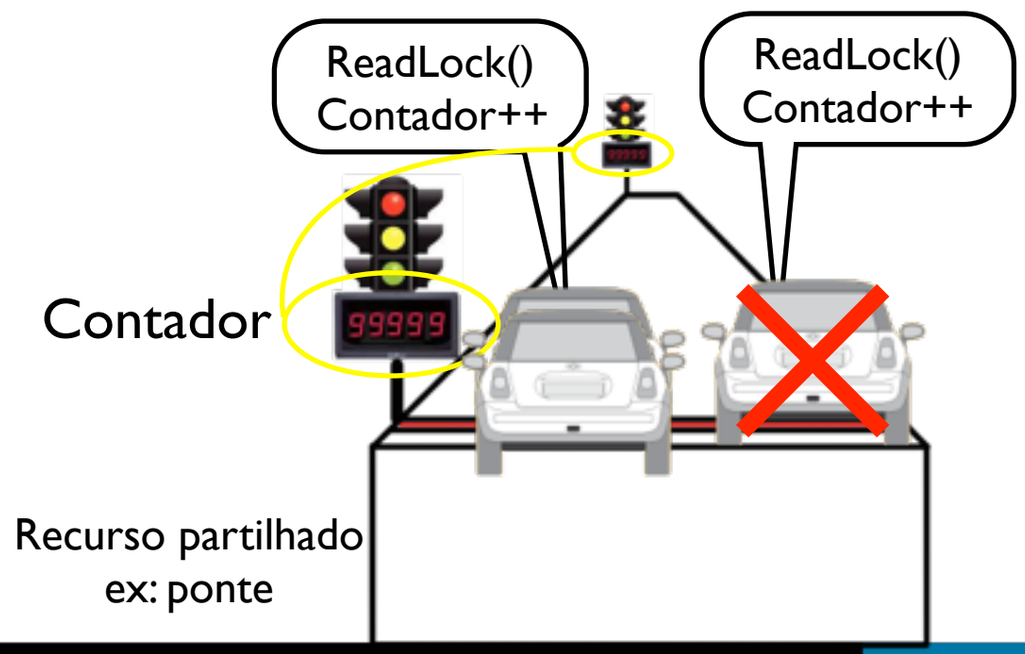
- Usando ReetrantLock e respectivas variáveis de condição.

Analogia (sincronização de acessos)

Recurso partilhado/zona crítica
ex: ponte

Analogia (sincronização de acessos)



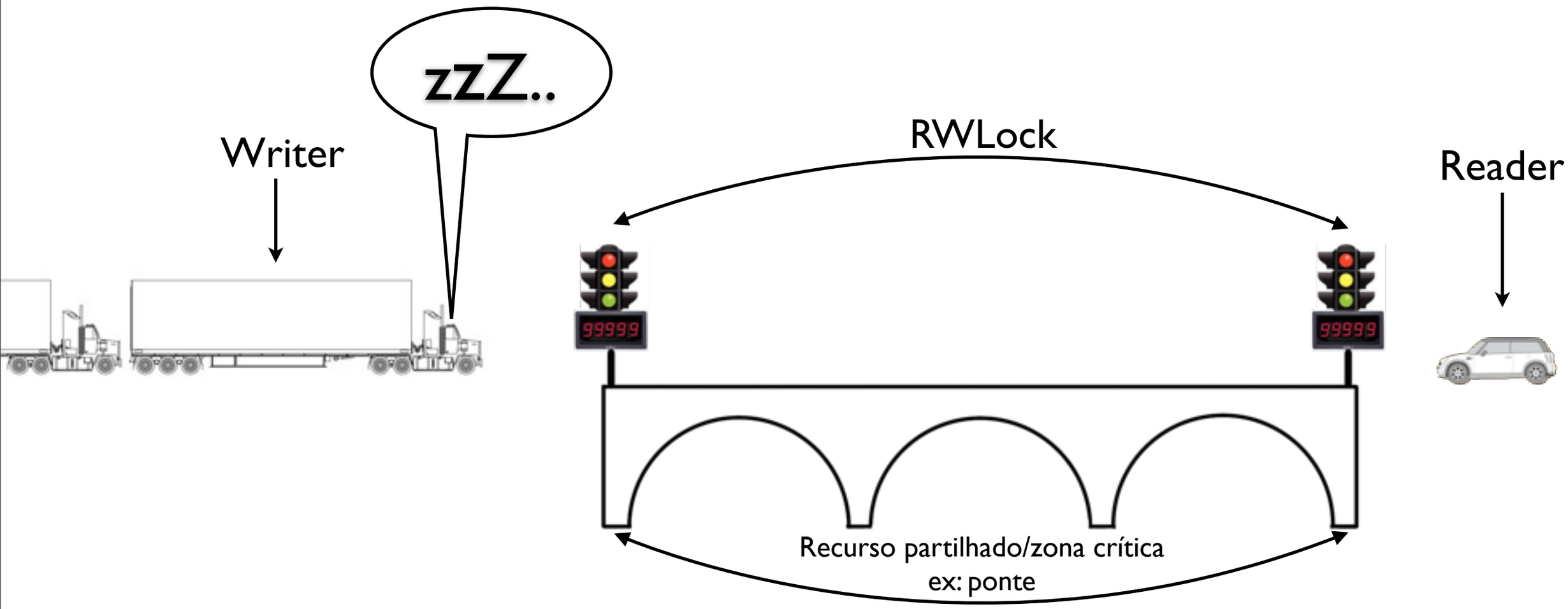


- Aula 1 ex Contador: acessos concorrentes a um contador partilhado pode levar a "perdas" na contagem.

Logo...

- <u>Garantir exclusão mútua no acesso à zona crítica!</u>

- Analogia (situação de starvation)



- Exercício: Procure reimplementar a classe RWLock de modo a evitar o fenómeno de starvation dos escritores.

Aula 6: Excercício 2

- Procure reimplementar a classe RWLock de modo a evitar o fenómeno de starvation dos escritores.

- Mas agora usando 2 variáveis de condição.