

Programação Concorrente em Java

Aula 02

Modelos de Concorrência

Programação Concorrente em Java

- Os modelos de concorrência podem ser:
 - Sistemas Distribuídos
 - Modelos Paralelos
 - Linhas de Montagem
 - simples e paralela
 - Paralelismo Funcional

Sistemas Distribuídos

- Modelos de concorrência são semelhantes a arquiteturas de sistemas distribuídos.
- Em sistemas concorrentes, diferentes threads trocam informações entre si.
- Em sistemas distribuídos, diferentes processos trocam informações entre si (possivelmente através de uma rede).
- Processos e Threads são naturalmente parecidos.

Sistemas Distribuídos

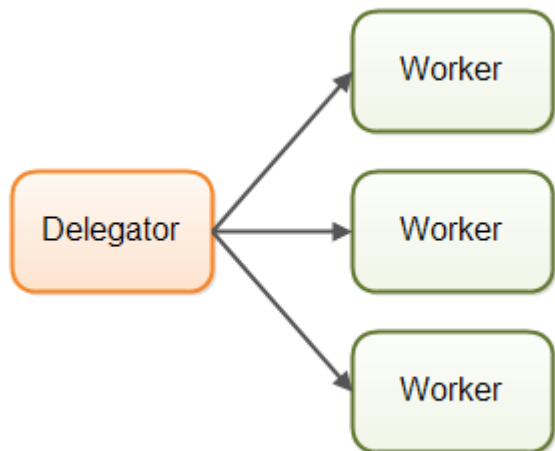
- A rede pode falhar;
- Um computador ou processo remoto podem falhar;
- Um sistema concorrente, executando em um grande servidor também pode experimentar os mesmo problemas:
 - Falha dos nós (nodes)
 - Falha na intranet (rede conectando os nós)
 - Falha nos discos, memória, etc...

Sistemas Distribuídos

- Como ambos sistemas são semelhantes (distribuídos e concorrentes), eles frequentemente emprestam ideais uns dos outros.
 - Balanceamento de carga;
 - Técnicas de tratamento de erros;
 - Idempotência de jobs;
 - Fail-over;

Workers Paralelos

- O primeiro e mais simples modelo de concorrência.



Workers Paralelos

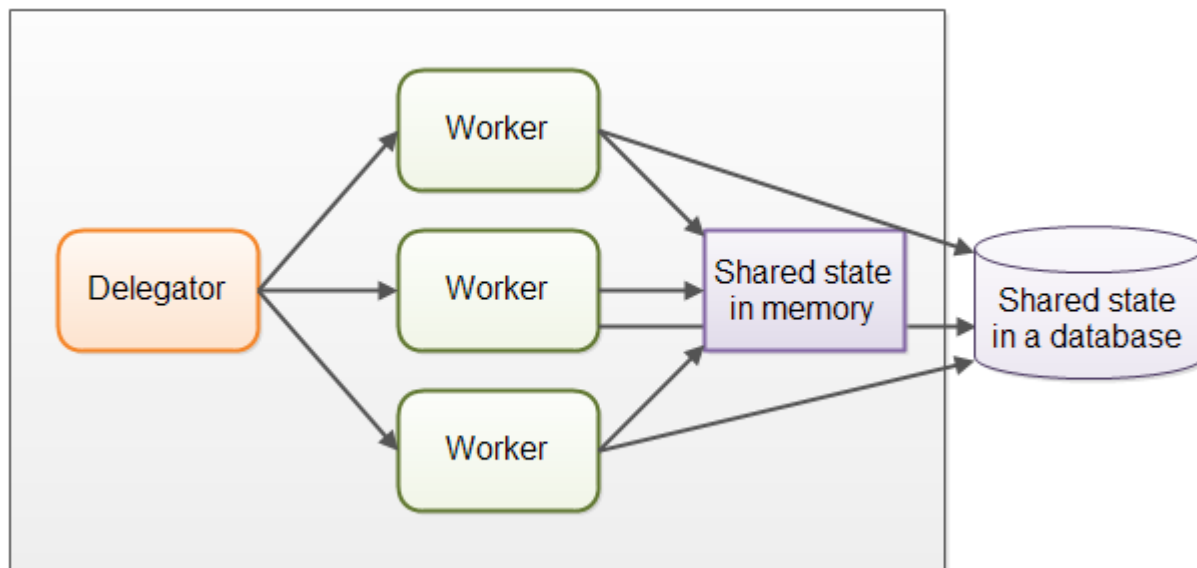
- Exercício
 - Implemente um exemplo simples, baseado em um delegador e threads workers que efetuem computações falsas.

Workers Paralelos - Vantagens

- É o modelo mais utilizado em Java e em várias aplicações de outras linguagens;
- É fácil de entender e implementar;
- É extremamente escalável;
- Não necessita de troca de mensagens entre os workers;
- Aplicações Bag-of-Tasks

Workers Paralelos - Desvantagens

- Estado compartilhado pode ficar complicado...



Workers Paralelos - Desvantagens

- Exercício
 - Implemente agora que cada Worker irá fazer a multiplicação de dois vetores (produto vetorial) de tamanho n e armazenar em uma variável local o seu valor.
 - Ao final de sua computação, todos os workers repassar o valor calculado para uma variável global (uma List, em Java).

Workers Paralelos - Desvantagens

- Estado compartilhado pode ficar complicado...
 - O dado compartilhado, modificado por uma thread, deve ser visto por outra.
 - As threads não podem ficar esperando demais para acessar a região crítica, ou o paralelismo perde seu efeito.
 - Algoritmos não blocantes a estruturas de dados são complicados de implementar.
 - Estruturas de dados persistentes

Workers Paralelos - Desvantagens

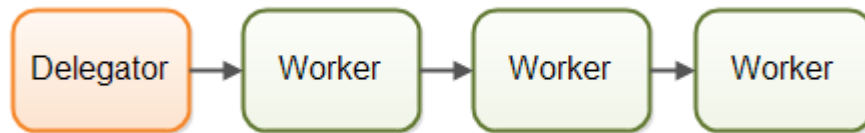
- Stateless Workers
 - Estado compartilhado pode ser modificado por outras threads no sistema.
 - Workers devem ler novamente o estado toda vez que precisarem se atualizar sobre o quê foi modificado.
 - Um worker que não mantém o estado de uma variável mas deve lê-lo toda vez que o necessita é chamado de **stateless**.
 - Rerler o dado toda vez que você necessita dele pode ser algo extremamente lento.

Workers Paralelos - Desvantagens

- Ordenação dos Jobs não é determinística
 - Não há nenhuma garantia que o Job A será executado primeiro que o Job B.
 - Torna-se muito complicado dizer qual o estado geral da aplicação em um determinado momento.

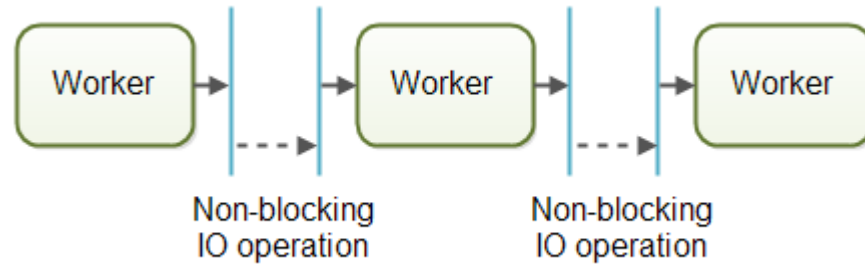
Linha de Montagem

- Os workers são organizados tais quais uma linha de montagem em um fábrica (montadora de carros, por exemplo).
- Cada worker faz apenas uma parte do trabalho total.
- A saída de um worker é a entrada de outro.
- Modelo também conhecido como **shared nothing**.



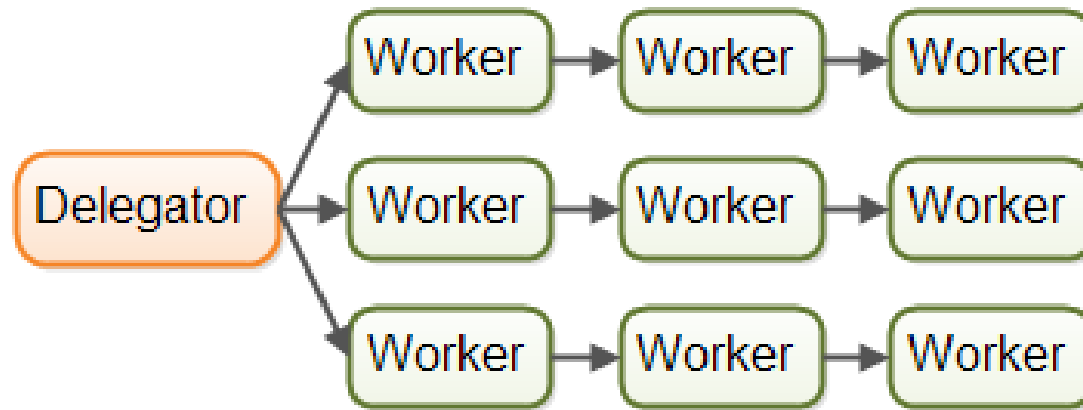
Linha de Montagem

- Non blocking IO



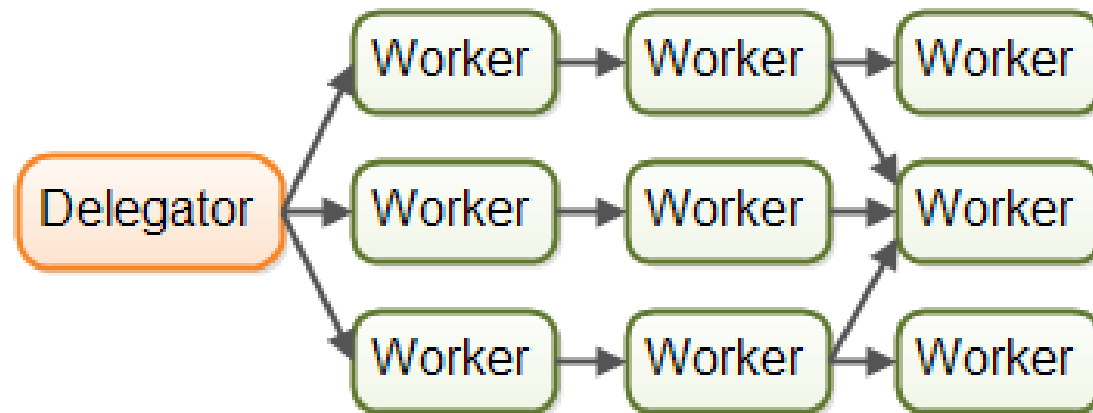
Linhas de Montagem

- Múltiplas linhas.



Linhas de Montagem

- Múltiplas linhas.

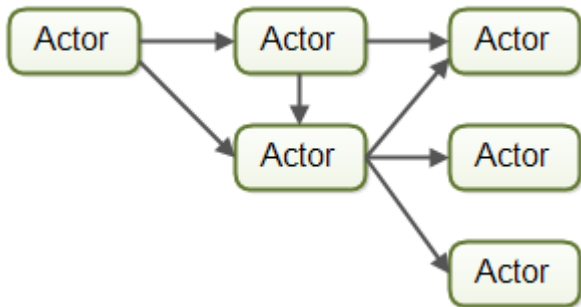


Linhas de Montagem

- Sistemas reativos ou sistemas voltados a eventos.
 - Os workers reagem a eventos que ocorrem no sistema, sejam esses eventos externos ou vindos de outros workers.
 - Requisições HTTP ou o término da leitura de um determinado arquivo.

Linhas de Montagem

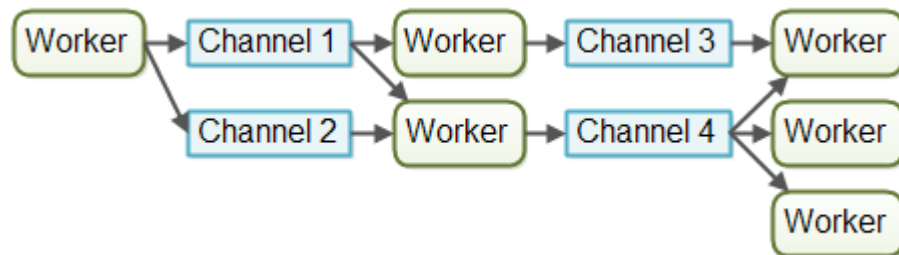
- **Atores vs Canais.**
 - Workers são chamados de “**atores**”
 - Atores enviam mensagens diretamente para cada um
 - Atores podem ser usando para processarem linhas de montagem



Linhas de Montagem

- Atores vs **Canais**

- Workers não se comunicam diretamente
- Eles publicam suas mensagens em canais
- Outros trabalhos podem ler as mensagens sem o remetente saber quem está lendo



Linhas de Montagem

- Vantagens
 - Sem estado compartilhado
 - Stateful workers (workers mantem dados próprios)
 - ***“Mechanical Simpathy”***
 - Ordenação de Jobs é possível

Linhas de Montagem

- Desvantagens
 - Múltiplos trabalhadores, múltiplas classes
 - **Callback Hell:** conjunto de callbacks aninhadas.

Paralelismo Funcional

- Também chamado de Paralelismo de Tarefas ou Controle
- Baseado em funções/tarefas que são ações ou atores
- Quando uma função chama a outra, é semelhante a passagem de mensagens entre workers
- Os parâmetros são copiados
- Cada função pode ser executada em CPUs diferentes

Paralelismo Funciona

program:

```
...  
if CPU = "a" then  
    do task "A"  
else if CPU="b" then  
    do task "B"  
end if  
...  
end program
```


Qual modelo usar?

- Depende muito do problema, da arquitetura, dos dados, etc.

Computação Paralela

- Exercício
 - Implemente a multiplicação de matrizes (versão serial e em paralelo) e compare os tempos para matrizes de diferentes tamanhos.
 - A sua implementação paralela deve criar uma Thread para cada conjunto linhaxcoluna.
 - Ao final, a matriz resultante deve ser mostrada.
 - Entrega e apresentação 05/03 (amanhã!).

Referências

- <http://tutorials.jenkov.com/java-concurrency/concurrency-models.html#concurrency-models-and-distributed-system-similarities>