INE5410
Laboratório 1 - Thread

Prof. Lau Cheuk Lung

Departamento de Informática e Estatística
Universidade Federal de Santa Catarina

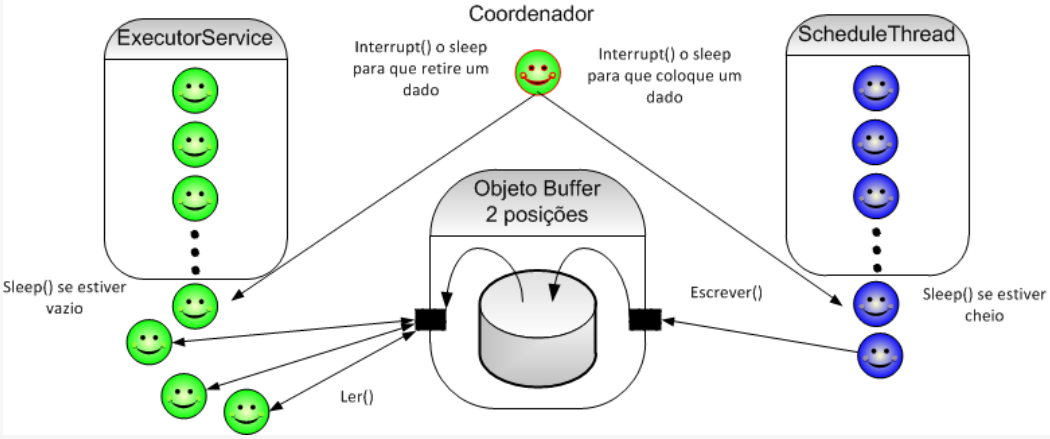
Descrição

Nesta atividade de laboratório você deve implementar um sistema Escritor/Leitor, sem nenhuma mecanismos de controle de concorrência do Java (Monitores, Locks ou Semáforos). As threads Azul são responsáveis por produzir um dado inteiro (aleatório) e colocar no buffer de duas posições apenas. As threads Verde são responsáveis por consumir esses dados, removendo-os do buffer e setando para zero (vazio).

Quando um thread Azul tentar colocar um dado no buffer e esse se encontrar cheio, este deve dormir por 60 segundos. Quando uma thread Verde tentar consumir um dado do buffer e este se encontrar vazio (ambas posições em zero), esta thread deve domir por 60 segundos.

O ScheduleThread deve controlar 100 threads produtor (Azul), lançando uma a cada 100ms.

O ExecutorService deve controlar 100 threads e deixar em estado de pronto apenas 4 threads por vez.



Neste modelo, existe a figura de um coordenador que avisa as threads Verde quando o buffer não está vazio e as threads Azul quando o buffer não está cheio. Esse coodenador faz esse aviso interrompendo as threads que estão dormindo usando o método interrupt(). Para isso, essa thread coordenador deve ter, de alguma forma, acesso a referência de todas as threads Azul e Verde do sistema.

Implementação

Implemente o Coordenador de duas formas, uma como objeto simples e outra como uma thread.

O buffer pode ser implementado, por exemplo, como um array simples de duas posições para variáveis inteiras.

Como objeto, deve ter dois métodos uma para ser avisado quando o buffer não está vazio (bufferNaoVazio()) e outra quando não está mais cheio (bufferNaoCheio()). Por exemplo, quando uma thread Azul coloca um dados no buffer ele invoca o método bufferNaoVazio() no objeto Coordenador e este invoca interrupt nas threads Verde que estão dormindo, para que estes interrompam seus sonos e tentem consumir um dado, se não conseguir volta a dormir. De forma similar, quando um thread Verde consome um dado ele invoca o método bufferNaoCheio() no objeto Coordenador para que este acorde as threads Azul que estiverem dormindo para que produzam um dados. Se uma thread tentar inserir um dado no buffer cheio ele volta a dormir.

Como uma thread, neste caso, a thread Coordenado não precisa ser avisada quando o buffer não está vazio ou cheio. Ela mesma fica monitorando o buffer a cada 150ms para verificar se o mesmo está cheio ou vazio e tomando as devidas providências, invocando o método interrupt() nas threads Azul ou Verde, para cada situação.

Crie 100 thread e use ExecutorService exec = Executors.newFixedThreadPool(4) para simular a liberação de 4 threads por vez.

Se preciso for, altere os tempo de sono (sleep) e/ou a taxa de liberação das threads Azul ou Verde para que o coordenador invoque o método interrupt() tanto nas threads Azul como Verde.

Apresentação

A atividade pode ser desenvolvida **dupla**. Em caso de **cópia** do código de outro aluno, ambos terão nota igual a **zero**.

O programa deverá ser apresentado ao professor no laboratório **até o dia 23/04**. Será verificado o funcionamento do programa e em seguida os alunos devem responder a questões sobre a forma como foram utilizados *threads* e monitores no programa. **Trabalho não entregue no prazo terão 2 pontos descontados por semana de atraso. Após duas semanas de atraso o trabalho não será mais aceito.**

Dúvidas

Atendimento aos Alunos

- Horário: Quartas-feiras das 16:00 às 17:40.
- Local: Prédio do INE - Sala 305.

E-Mail

l a u . l u n g @ i n f . u f s c . b r (turma A)

Mantida por Lau Cheuk Lung. Atualizada em 08/04/2014.