

Cenas da vida do Pai Natal *

O Pai Natal mantém-se quase sempre na sua casa do Pólo Norte e, face à sua idade avançada, passa boa parte do tempo a descansar. Só intervém em duas situações

- quando as renas, que foram passar férias ao Pacífico Sul, regressam na véspera de Natal para puxar o trenó na viagem de distribuição de presentes às crianças de todo o mundo
- quando alguns dos gnomos que trabalham na fábrica de brinquedos ao lado, têm algum problema e precisam do seu conselho.

Para que o Pai Natal não esteja continuamente a ser incomodado, os gnomos decidiram só lhe bater à porta quando houver exactamente três que pretendam ouvir a sua opinião. Havendo mais gnomos nessa situação, os restantes têm que aguardar pelo regresso dos primeiros à fábrica antes de poderem avançar (sempre em grupos de três).

Se a última rena já tiver voltado dos trópicos, o Pai Natal sabe que chegou o momento de aparelhar o trenó e de dar início à viagem porque é Véspera de Natal. Qualquer grupo de gnomos que esteja nesse momento à espera de ser atendido, terá que esperar pelo seu regresso, dado que a viagem é prioritária.

Supõe-se ainda que as renas regressam de férias uma de cada vez e de uma forma independente umas das outras e que aguardam no estábulo o início da viagem. A última, porém, ao constatar esse facto, vai também bater à porta da casa do Pai Natal para o alertar que as renas já estão todas reunidas. Quando a viagem termina, as renas partem imediatamente para férias para recuperar forças do esforço despendido.

Construa uma simulação das actividades de cenas da vida do Pai Natal baseada no modelo cliente-servidor, com replicação de servidor, em que o Pai Natal, os gnomos e as renas são os *clientes* e as regiões de interacção que tenha estabelecido representam os serviços que lhes são prestados pelos *servidores*.

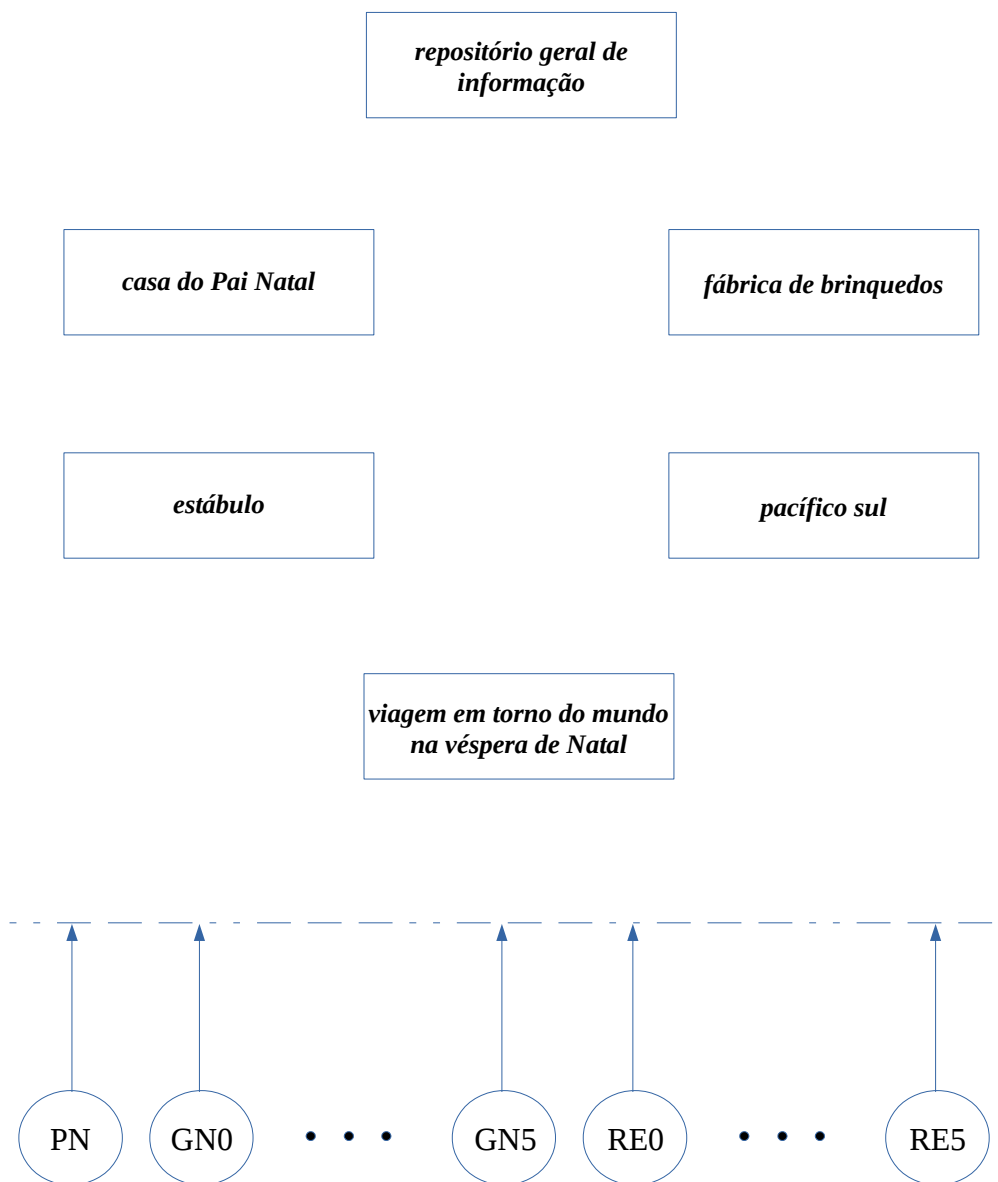
A solução deve ser implementada em Java, ser passível de execução em Linux sobre *sockets* TCP, poder ser executada de uma forma concentrada (numa única máquina), ou de uma forma distribuída (até oito máquinas diferentes), e terminar (deve contemplar a possibilidade de *shutdown* do serviço).

As operações que correspondiam, na versão concorrente já implementada, a actividades realizadas sobre a(s) estrutura(s) de dados partilhada(s), devem agora corresponder a solicitações independentes requeridas ao(s) servidor(es) através de mensagens. Em cada caso, deve ser estabelecida a ligação, ser feito o pedido, aguardar a resposta e fechar a ligação.

Incorpore um ficheiro de *logging* que descreva de um modo conciso, mas claro, a evolução do estado interno das diversas entidades envolvidas.

* Adaptação de um problema descrito em Stallings, *Operating Systems*, Prentice-Hall International Inc., 4.ª Ed.

Servidores



Etapas para construção da solução

1. Especificar para cada servidor representativo de uma *região de partilha de informação* a estrutura das mensagens trocadas.
2. Proceder à sua codificação em Java como tipo de dados de referência específicos.
3. Especificar a organização geral da arquitectura dos servidores.
4. Proceder à sua codificação em Java como tipo de dados de referência específicos.
5. Especificar a organização geral da arquitectura dos clientes.
6. Proceder à sua codificação em Java como tipo de dados de referência específicos.
7. Estabelecer o mapeamento dos servidores e dos clientes em nós diferentes da máquina paralela e definir *shell scripts* que possibilitem o deployment e a execução dos vários módulos que compõem a aplicação.
8. Construir os *diagramas de interacção* que descrevem de um modo compacto, mas preciso, a dinâmica da aplicação.
9. Validar a solução efectuando execuções múltiplas e verificando para cada uma a correcção dos resultados obtidos por análise do conteúdo do ficheiro de *logging*.