

INSTITUTO SUPERIOR TÉCNICO Departamento de Engenharia Informática Enunciado do Projeto de Sistemas Operativos – Parte 2 LEIC/LETI 2013-2014 – 1º Semestre

1 KOS: Key value store of the OS course - Parte 2

A 2ª parte do projecto dá continuidade à 1ª parte já publicada aumentando a funcionalidade do KOS (*Key-value store of the Operating Systems course*) no sentido de a tornar mais próxima do que se encontra em serviços semelhantes de sistemas operativos.

Vamos agora assumir que:

- 1. O número de clientes será diferente e tendencialmente superior do número de servidores, e
- 2. O KOS é persistente, isto é, os dados nele armazenados sobrevivem às actividades que os criaram ou a eles acederam.

As novas funcionalidades são especificadas de modo a permitir a reutilização do que já foi desenvolvido na 1ª parte do projecto: mantêm-se todas as características conceptuais e a arquitectura geral do KOS (figuras 1 e 2 do Enunciado – Parte 1) bem como a estrutura de dados de cada partição (figura 3 do Enunciado – Parte 1). Mantém-se, designadamente, o modelo de interação cliente-servidor:

As tarefas clientes não acedem directamente ao estado do KOS. Em vez disso, estas tarefas devem interagir com as tarefas servidoras usando as funções especificadas no ficheiro kos client.h

As novas funcionalidades incidirão na gestão da comunicação entre clientes e servidores, no controlo da concorrência nos acessos aos dados guardados no KOS em memória e no armazenamento persistente destes dados.

2 Requisitos da 2ª Parte do Projecto

2.1 Concorrência

Considera-se nesta parte, de uma forma mais realista, que o número de clientes é diferente do número de servidores (tendencialmente será superior).

Número de Clientes ≠ Número de Servidores

Na $1^{\underline{a}}$ parte do projecto havia uma correspondência biunívoca entre um cliente e o seu servidor. Pretende-se agora que o *buffer* de comunicação entre clientes e servidores veicule pedidos de M clientes para N servidores, sendo M \neq N.

Um cliente coloca um pedido de serviço no *buffer* sendo atendido por um servidor que no momento esteja disponível. Estabelecida a associação de um servidor a um cliente para o atendimento de um pedido a interacção entre ambos – alocação de *slots* do *buffer*, modelo de sincronização, transferência de dados através do *buffer* – deverá ser ajustada (relativamente ao que foi desenvolvido na 1ª parte) para permitir a ligação dinâmica entre o cliente e o servidor do seu pedido. Para tal sugere-se que se considere o tipo de problema dos produtoresconsumidores.

Na inicialização assume-se que primeiro são lançados os servidores que ficam à espera de pedidos de clientes

Paralelismo no acesso ao KOS

Aumentando o número de clientes a carga nos servidores, medida pelo número de pedidos, tenderá a aumentar. É por isso importante minimizar a contenção entre os servidores no acesso aos dados do KOS em memória.

Mantendo-se o modelo de dados do KOS e as operações nele invocadas – *get, put, remove, getAllKeys* – são possíveis várias aproximações para aumentar a concorrência. Indicam-se de seguida algumas opções:

- O1. Permitir acessos paralelos a partições diferentes (opção trivial provavelmente já implementada na 1ª parte do projecto);
- 02. Permitir acessos em leitura (get, getAllKeys) paralelos numa partição;
- 03. Permitir acessos paralelos a listas distintas numa partição;
- O4. Permitir sequências de pesquisa (varrimento de uma lista antes de atingir o local do *put* ou do *remove*) e acessos em leitura paralelos num partição.

Note-se que a opção seguida tem implicações na gestão da concorrência dos acessos ao sistema de ficheiros para propagar as alterações ao KOS em memória.

Fica ao critério do alunos a escolha da opção a seguir de entre as sugeridas (para além da primeira que, como referimos, é trivial) ou outras que <u>justificadamente</u> considerem adequadas. No entanto recomenda-se que o projecto final, cumprindo todos os requisitos pedidos, seja realizado seguindo opções de paralelismo do tipo O2 ou O3 (problema do tipo leitores-escritores) e, só após a realização e teste com sucesso de uma versão operacional do projecto

completo, se parta para o refinamento da solução (≥ 04), mantendo intacta a versão operacional inicial para entrega.

2.2 Persistência dos dados do KOS

Pretende-se que os dados do KOS sejam armazenados de forma persistente no sistema de ficheiros UNIX.

Cada partição deverá ser guardada num ficheiro de nome fshardId. O ficheiro é actualizado à medida que a partição em memória é alterada. O fluxo de dados do KOS entre a memória e o sistema de ficheiros obedece às regras seguintes:

- A inicialização de cada partição em memória é feita a partir do ficheiro correspondente na fase de inicialização do sistema.
 - Se não existir este ficheiro assume-se que a partição não tem dados, devendo ser inicializada vazia em memória e criado e inicializado o ficheiro correspondente.
- Os operações de leitura (*get, getAllKeys*) não alteram a partição em memória, logo não implicam qualquer alteração no ficheiro-partição.
- A escrita é atómica, isto é, a operação de escrita decorrente da execução de um pedido *put ou remove* é realizada na partição em memória e é imediatamente propagada para o ficheiro-partição.
 - O Uma operação que implique uma escrita (put, remove) só se considera concluída após a escrita da informação em memória e no ficheiro. Considera-se que a escrita no ficheiro foi realizada após o retorno da chamada à função da API do sistema de ficheiros UNIX que efectua a escrita.

Uma vez que os pares *Key-Value* continuam a ser manipulados pelos servidores nas partições em memória, os dados a armazenar no ficheiro-partição podem seguir uma estrutura mais simples e compacta:

Número de pares *Key-Value* armazenados no ficheiro – int NPKV – seguido da sequência dos pares *Key-Value* guardados sem nenhuma ordem particular (opção P0).

Esta solução, embora muito simples, obriga a procurar os pares <key,value> a que se pretende aceder – para apagar (num *remove*) ou actualizar (num *put*) – num varrimento sequencial do ficheiro. Sugere-se que, após ter implementado desta forma e testado com sucesso a persistência do KOS, sejam também investigadas alternativas de implementação mais eficientes que permitam evitar o varrimento sequential dos ficheiros e mantê-los compactos.

Colocam-se à consideração as seguintes optimizações:

- P1. Armazenamento na partição em memória primária da posição (deslocamento) do registo <key,value> em ficheiro, evitando o varrimento.
- P2. Compactação periódica do ficheiro, eliminando os registos vazios em resultado de *removes*.
- P3. Reocupação de *slots* <key,value> vazios em novas inserções, optimizando o endereçamento destas posições pelo armazenamento do seu deslocamento no ficheiro em lista ou em *bitmap* residentes em memória.

3 Calendário proposto

Data	Acção	Quem faz
4 Nov	Publicação do enunciado da 2ª parte do projecto.	Corpo docente
5 Nov – 3 Dez	Realização da 2ª parte do projecto e eventuais	Alunos
	ajustes à 1ª parte.	
	A especificação das funcionalidades relacionadas	
	com a melhoria da concorrência e com a	
	persistência permitem o desenvolvimento e	
	teste destas partes com grande independência.	
	Fases sugeridas:	
	1. Desenvolvimento e teste do projecto	
	completo seguindo as opções de	
	paralelismo 02-03 e de persistência P0.	
	2. Guardar esta versão do projecto para	
	eventual entrega.	
	3. Só após a conclusão do projecto completo	
	se deve tentar optimizar o paralelismo e o	
	acessos aos ficheiros.	
3 Dez, 23:59	Entrega final do projecto.	Alunos
4 Dez – 10 Dez	Visualização do projecto.	Alunos + Corpo docente
11 Dez – 20 Dez	Avaliação oral do projecto.	Alunos + Corpo docente

Recordam-se as informações sobre a avaliação do projecto já incluídas no enunciado da 1ª parte:

Os alunos devem entregar uma concretização completa do projeto por via electrónica através do sistema Fénix. Esta entrega final deverá incluir a 2ª parte e a eventual revisão de aspectos da 1ª parte que tenham sido melhorados após a entrega de controlo em 4 de Novembro. O formato do ficheiro de entrega com o código será indicado oportunamente.

A avaliação do Projecto completo será feita no fim do semestre e consta da sua visualização na semana imediatamente a seguir à entrega e de uma discussão oral na semana seguinte. Em ambos os actos deverão estar presentes todos os elementos do grupo, sendo atribuída uma classificação individual a publicar na página da disciplina. Na atribuição dessa nota, para além da qualidade do programa apresentado (Parte I e Parte II), serão levados em conta factores como o desempenho individual na discussão do projeto, a participação nas aulas de laboratório e o acompanhamento do progresso do projecto feito pelos docentes das aulas práticas.

O peso da Parte I vale 40% da nota final do projeto, enquanto que a Parte II contará 60%.

As visualizações e as discussões orais decorrerão preferencialmente nos horários de laboratório entre 4 e 20 de Dezembro. Nos turnos com mais grupos os docentes do turno marcarão – no mesmo período de 4 a 20 de Dezembro – horários extra que serão oportunamente divulgados.

A informação sobre a entrega e avaliação aqui descrita está sujeita a alterações que, caso ocorram, serão afixadas na página da cadeira.