Douglas Pereira Luiz 18203343

Atividade A.4 Middleware Orientado a Mensagens

Sumário

A ferramenta utilizada foi o Zero MQ na linguagem C.

Os sistema resolve contas. Uma "conta" é uma expressão consistindo de somas, subtrações, multiplicações e divisões de operandos. A expressão é em formato infixo e possui prioridade de tipo de operador, ainda podendo apresentar parenteses.

O sistema é composto por 5 procesos:

- **Processo que gera contas,** definido em "conta.c". Esse processo publica contas à serem resolvidas. A publicação tem um identificador numérico no início, para indicar a publicação à um resolvedor;
- **Processos que querem resolver as expressões**, definidos em "exp.c". Esses processos sabem parcialmente resolver uma expressão em notação infixa. Eles tem um identificador cada (foram definidos somente 1 e 2) que, estando inscritas nas publicações de "conta.c", os permitem filtrar as expressões direcionadas à eles. Esses processos não sabem somar, subtrair, multiplicar ou dividir, somente sabem interpretar a notação infixa e organizar quais e em que ordem operações de 2 operandos deve ser feitas. Para efetivamente resolverem suas contas, as operações com 2 operandos são publicadas para que quem souber resolver, resolva e publique então a resposta;
- **Processo de soma.** Sabe somar/subtrair uma expressão simples com 2 operandos. Está inscrito nas publicações dos processos resolvedores de expressão. Ele filtra por expressões de soma, sendo que a mensagem deve conter um identificador de quem enviou. Depois de resolver, a publicação será composta pelo identificador de quem solicitou a resposta, para permitir filtragem, e da resposta em si;
- **Processo de multiplicação.** Faz o mesmo que o processo de soma, mas com operadores multiplicativos.

Foram obtidas expressões através do código da ferramenta encontrada em https://github.com/dshepsis/ExpressionGenerator

conta.c

O "publisher" de "conta.c" é definido na porta 5555.
Ele publica mensagem na forma

id: conta

, sendo id o identificador direcionando para um resolvedor de expressões (1 ou 2), e conta uma expressão à ser resolvida. As expressões estão definidas no arquivo "contas" (Obs: o arquivo precisa de uma última linha vazia)

É definido um intervalo entre as publicações por “usleep(10000);”.

exp.c

Os processos de resolução de expressão deve ser executados com o argumento 1 ou 2, que define seu id. Executando "./exp.out 1" ou "./exp.out 2"
Cada um conta com 2 "subscribers". Um que se inscreve para consumir contas, e outro para consumir respostas das requisições de operação.
Ambas inscrições filtram por publicações direcionadas ao processo, ou seja, que começam com "seu_id:".

O "publisher", usado para publicar a produção de requisições de operações simples, fica na porta ou 5558 (quando executamos "exp.out 1") ou 5559 (quando executamos "exp.out 2").
As mensagens publicadas são no formato

sum: v1 op v2 seu_id

ou

mul: v1 op v2 seu_id

. Quando ele publica uma mensagem, ele espera pela publicação de uma resposta direcionada à "seu_id".

É definido um intervalo entre as resoluções por “usleep(10000);”.

sum.c e mul.c

Os processos de soma e multiplicação, definidos respectivamente em "sum.c" e "mul.c", são "subscribers" dos resolvedores de expressão, filtrando por "sum:" ou "mul:".
Os "publishers" de soma e multiplição ficam na porta 5556 e 5557, respectivamente. Depois de "pegar" uma mensagem que possui uma conta e um id, publicam uma resposta na forma

id: resultado

. Dessa forma, o resolvedor de expressões pode filtrar por mensagem de sua requisição.

Instalação do ZeroMQ

(ubuntu/mint)

apt-get install libzmq3-dev

apt-get install libczmq-dev

Instruções do site:

- https://zeromq.org/download/
- https://zeromq.org/languages/c/

Compilar

gcc -o mul.out mul.c -lzmq
gcc -o sum.out sum.c -lzmq
gcc -o exp.out exp.c -lzmq
gcc -o conta.out conta.c -lzmq

Executar

(cada um em um terminal diferente):

./mul.out
./sum.out
./exp.out 1
./exp.out 2
./conta.out
./conta.out <repetições>

<repetições> é o número de vezes que serão publicadas as contas do arquivo:

- ./conta.out 1 vai publicar uma vez todas as contas;
- ./conta.out 100 vai publicar cem vezes todas as contas do arquivo.

O ideal é executar nesta ordem, já que exp precisa estar ativo quando conta publicar, e mul e sum precisam estar ativos quanto exp precisar resolver uma conta.