Nome: Rafael Pais Cardoso DRE: 116140788

Laboratório 2
Aplicação cliente/servidor básica

Sistemas Distribuídos (MAB-367)
Profa. Silvana Rossetto

Instituto de Computação/UFRJ

Atividade 1

Objetivo: Refinar a arquitetura de software — usando o estilo arquitetural em camadas — apresentada abaixo.

Camadas:

1. **Funcionalidades da camada de interface com o usuário:** recebe do usuario o nome do arquivo e a palavra de busca e exibe na tela o resultado do processamento. O resultado do processamento poderá ser: *(i) uma mensagem de erro indicando que o arquivo não foi encontrado; ou (ii) o numero de ocorrências da palavra no arquivo.* As mensagens virão em um formato json e, em seguida, a informação contida nessa estrutura será extraída/editada e exibida na tela do usuário informando a contagem ou alguma mensagem de erro.

2. **Funcionalidades da camada de processamento:** solicita o acesso ao arquivo texto. Se o arquivo for válido, realiza a busca pela palavra informada e prepara a resposta para ser devolvida para a camada de interface. Se o arquivo for inválido, responde com a mensagem de erro. O resultado para a camada de interface será entregue no formato json contento o número de palavras que foram contadas no arquivo. Além disso, se o arquivo for inválido ou se algo inesperado ocorrer, também existirá uma mensagem de erro relatando o problema.

3. **Funcionalidades da camada de acesso aos dados:** verifica se o arquivo existe em sua base. Se sim, devolve o seu conteúdo inteiro. Caso contrário, devolve uma mensagem de erro.

Segue abaixo a estrutura do json que será utilizado e que foi mencionada anteriormente.

```
{
  "arquivo": "receitas.txt",
  "palavra":"abacate",
  "contagem": 137,
  "sucesso": True,
  "mensagemErro": None
}
```

Código 1: estrutura da mensagem trocada entre o cliente e o servidor. Esta é a mensagem de resposta do servidor para o cliente.

Nome: Rafael Pais Cardoso DRE: 116140788

Atividade 2

Objetivo: Refinar a proposta de instanciação da arquitetura de software da aplicação definida na Atividade 1 para uma arquitetura de sistema cliente/servidor de dois níveis, com um servidor e um cliente, apresentada abaixo.

Proposta de arquitetura de sistema:

1. Lado cliente: implementa a camada de interface com o usuário. O usuário poderá solicitar o processamento de uma ou mais buscas em uma única execução da aplicação: o programa espera pelo nome do arquivo e da palavra de busca, faz o processamento, retorna o resultado, e então aguarda um novo pedido de arquivo e palavra ou o comando de finalização.

2. Lado servidor: implementa a camada de processamento e a camada de acesso aos dados. Projete um servidor iterativo, isto é, que trata as requisições de um cliente de cada vez, em um unico fluxo de execução (estudaremos essa classificação depois). Terminada a interação com um cliente, ele poderá voltar a esperar por nova conexão. Dessa forma, o programa do servidor fica em loop infinito (depois veremos como lidar com isso).

Especificações:

- Mensagem

Como mencionado anteriormente, no Código 1 temos a representação da estrutura que será a mensagem trocada entre o cliente e o servidor.

O cliente irá mandar essa estrutura json apenas com os campos “arquivo” e “palavra” preenchidos. Os demais campos ficarão com “None”. Um exemplo dessa estrutura pode ser observado no Código 2.

O servidor devolve essa estrutura json para o cliente com os campos “contagem” e “sucesso” preenchidos com o número da contagem e com o valor “True”, respectivamente, caso encontre o arquivo e consiga fazer a contagem da palavra buscada. Caso o arquivo não exista, o servidor devolve essa estrutura json para o cliente sem o valor da contagem, mas com o campo “sucesso” preenchido com o valor “False” e o campo “mensagemErro” preenchido com uma mensagem de erro. Um exemplo dessa estrutura pode ser observado no Código 3.

- Lado cliente:

No lado cliente, verifica-se a validade das entradas digitadas pelo usuário. Por exemplo, se o usuário digitar apenas espaços ou algo vazio, isso não será considerado um arquivo ou uma palavra válida. Nesse sentido, o usuário será convidado a digitar novamente algo que seja válido.

Quando o nome do arquivo e a palavra a ser buscada são válidos, ocorre a estruturação da mensagem em um json e esta é enviada para o servidor. No código 2, podemos ver um exemplo dessa estrutura.

```
{
  "arquivo": "receitas.txt",
  "palavra":"abacate",
  "contagem": None,
  "sucesso": None,
  "mensagemErro": None
}
```

Código 2: estrutura da mensagem enviada pelo cliente para o servidor.

Após o processamento no lado do servidor, o cliente recebe uma mensagem estruturada informando a contagem, como a observada no Código 1, ou informando algum erro que ocorreu durante o processamento no lado do servidor, como observado no Código 3. A partir dessa estrutura, a mensagem que será impressa para o usuário é construída e, em seguida, apresentada para o mesmo.

Em caso de sucesso, a mensagem que será impressa para o usuário é "No arquivo receitas.txt foi possível encontrar 137 ocorrências da palavra abacate.". Se algo deu errado, será impresso, por exemplo, a mensagem "Arquivo não encontrado.".

```
{
  "arquivo": "massas.txt",
  "palavra":"espinafre",
  "contagem": None,
  "sucesso": False,
  "mensagemErro": "Arquivo não encontrado."
}
```

Código 3: estrutura da mensagem enviada pelo servidor para o cliente. Note que esta mensagem apresenta uma mensagem de erro, uma vez que o arquivo especificado não existe na base.

Caso o usuário não tenha mais o interesse de consultar os arquivos da base, pode-se utilizar o comando de finalização "||END||" para encerrar a sua conexão com o servidor e, assim, encerrar a execução do lado do cliente.

- Lado servidor:

O lado servidor irá receber a estrutura observada no Código 2 e irá realizar o processamento. Aqui, mais uma vez, será verificado a validade do nome do arquivo e da palavra e, além disso, se o caminho para o arquivo existe. Caso ocorra algum problema nesse momento, será enviado para o cliente uma mensagem estruturada, como observada no Código 3. Do contrário, o processamento será iniciado.

No processamento é feito a normalização do nome de arquivo buscado e da palavra consultada para que, por exemplo, as palavras "Alface" e "alface" sejam consideradas a mesma palavra. Além disso, o conteúdo do arquivo é normalizado e alguns caracteres são

removidos, como “,”,”.”, etc., para que não atrapalhem no consulta da palavra. Caso isso não fosse feito, a palavra “alface,” não seria contabilizada por conta da vírgula. Em seguida, cada palavra do arquivo é transformado em um token, que é adicionado a uma lista e, em seguida, esta é varrida para contabilizar o número de ocorrências de uma determinada palavra.

Após isso, o servidor estrutura a mensagem que será enviada para o cliente com a contagem da palavra no arquivo especificado e aguarda por novas requisições do cliente. Se o atual cliente optar por encerrar a sua conexão com o servidor, este irá atender o próximo cliente, caso tenha alguém na fila, ou aguardará pelo próximo.