ACH2147 – Desenvolvimento de Sistemas de Informação Distribuídos

1° semestre de 2025

Exercício Programa: parte 2

Felipe Freire Silva – 14749596

Vitor Pazos – 14611814

Sumário

1 Objetivos

Nesta segunda parte do exercício, vamos implementar um protocolo simples baseado no relógio de Lamport para prover consistência à informação de status dos peers. Além disso, vamos começar a implementar a funcionalidade de busca e download de arquivos, ou seja, o comando 4 do menu do programa.

2 Decisões de projeto

Nesta seção, são apresentadas as principais decisões de projeto tomadas para a implementação da Parte 2 do EACHare. O foco desta etapa foi a atualização do código de acordo com as atualizações mencionadas no relatório. As escolhas feitas buscaram manter a linha de raciocínio da primeira parte do exercício.

2.1 Dificuldades encontradas

Garantir o funcionamento do código com as alterações exigidas para a segunda parte foi a maior dificuldade desta etapa, decidir quais partes deveriam ser refeitas ou apenas modificadas para implementar as mudanças necessárias.

2.2 Influência da parte 1

Na primeira parte do EACHare, algumas escolhas de implementação importantes foram feitas para estruturar o sistema P2P de forma simples e modular. Entre as principais decisões, destacam-se:

- **Estrutura dos peers conhecidos:**
  No EP1, os peers conhecidos eram armazenados em um dicionário simples, associando cada peer ao seu status (ONLINE/OFFLINE). No EP2, foi necessário estender essa estrutura para também armazenar o valor do relógio lógico de Lamport de cada peer. Como a estrutura já era centralizada e de fácil acesso, a adaptação foi direta, bastando adicionar o campo do relógio.
- **Classe do relógio lógico:**
  A existência da classe Relogio no EP1 permitiu modificar facilmente o comportamento do relógio para seguir as regras do relógio de Lamport no EP2 (incrementar antes de enviar, atualizar para o máximo ao receber, etc.), sem precisar reescrever toda a lógica de eventos.
- **Padronização das mensagens:**
  Como as mensagens já eram encapsuladas na classe Mensagem, foi simples incluir o valor do relógio no cabeçalho das mensagens e garantir que todas as trocas de mensagens seguissem o novo protocolo exigido no EP2.

- **Separação de responsabilidades:**
  A divisão do código em módulos (peer.py, menu.py, mensagem.py, relógio.py) facilitou a implementação de novas funcionalidades, como o comando de busca e download de arquivos, além das alterações no gerenciamento de peers e relógio.

- **Leitura de vizinhos por arquivo:**
  A leitura dos vizinhos a partir de arquivos de texto continuou útil no EP2, permitindo testar facilmente diferentes cenários de rede.

Essas escolhas de design modular e padronizado no EP1 permitiram que as mudanças do EP2 fossem implementadas de forma incremental, sem a necessidade de grandes reestruturações no código, tornando o desenvolvimento mais ágil e seguro.

2.3 Refatoração de código

Foi necessário refatorar parte do código para adaptar o sistema às exigências do EP2. As principais refatorações incluíram:

- **Estrutura dos peers conhecidos:**
  No EP2, além do status, era necessário armazenar o valor do relógio lógico de Lamport para cada peer, para garantir a consistência das informações distribuídas. Seria possível manter duas estruturas separadas (um dicionário para status e outro para relógio), mas isso aumentaria a complexidade e o risco de inconsistências. Refatorar para uma estrutura única torna o código mais limpo e seguro.

- **Atualização do relógio lógico:**
  A classe Relogio precisou ser ajustada para seguir exatamente o algoritmo de Lamport, incluindo a atualização do relógio ao receber mensagens (usando o valor máximo entre o local e o recebido, e incrementando em seguida).

- **Processamento de mensagens:**
  Os métodos de envio e recebimento de mensagens foram refatorados para garantir que o valor do relógio fosse corretamente incluído no cabeçalho das mensagens e atualizado conforme o protocolo.

- **Formato das mensagens:**
  Foi necessário adaptar o formato das mensagens para incluir o valor do relógio e, em comandos como PEER_LIST, adicionar o campo do relógio de cada peer listado.

As refatorações foram necessárias para garantir clareza, manutenção e aderência ao novo protocolo. Seria possível implementar as mudanças de forma menos estruturada, mas isso aumentaria a complexidade, dificultaria a manutenção e aumentaria o risco de erros. Refatorar foi a escolha mais adequada para evoluir o sistema de acordo com os requisitos do EP2.

2.4Testes

Os testes foram totalmente manuais para garantir que o funcionamento e interface fossem compatíveis com a descrição do enunciado. Dentre os testes realizados, destaca-se o comando 4, em que foram criados diferentes diretórios com o intuito de testar a funcionalidade de downloads.

3 Implementação

Nesta seção, apresentamos um resumo das principais partes da implementação do projeto EACHare – Parte 2. A arquitetura do sistema foi desenvolvida com foco em modularidade, clareza e facilidade de manutenção, permitindo que as responsabilidades do sistema fossem distribuídas de forma lógica entre diferentes arquivos e classes.

A seguir, serão descritos os principais componentes do sistema, abordando a forma como cada um contribui para o funcionamento geral da aplicação. Cada subseção resume a finalidade de uma parte do código, que se encontra devidamente comentado nos arquivos entregues.

3.1 Comando [4]

O comando [4] foi implementado para permitir que um peer realize a busca e o download de arquivos compartilhados por outros peers na rede. Ao selecionar essa opção no menu, o usuário pode informar o nome do arquivo desejado. O peer, então, envia uma mensagem de busca para todos os peers conhecidos que estejam ONLINE, solicitando informações sobre a disponibilidade do arquivo.

Quando um peer recebe uma solicitação de busca, ele verifica se o arquivo está presente em seu diretório compartilhado. Caso o arquivo exista, o peer responde informando sua disponibilidade. O peer solicitante, ao receber as respostas, exibe ao usuário quais peers possuem o arquivo e permite escolher de qual peer deseja realizar o download.

O processo de download é iniciado com o envio de uma mensagem de requisição ao peer escolhido. O peer que possui o arquivo abre o arquivo em modo leitura e envia seu conteúdo em blocos pela conexão de socket. O peer solicitante recebe os dados e salva o arquivo em seu diretório compartilhado local.

Toda a comunicação segue o protocolo de mensagens padronizado, incluindo o valor do relógio lógico de Lamport para garantir a consistência dos eventos. O comando 4 reforça a modularidade do sistema, aproveitando as estruturas já existentes para troca de mensagens, controle de peers e manipulação de arquivos, tornando a implementação eficiente e alinhada com o restante do projeto.

4 Conclusão

A Parte 2 do exercício programa EACHare expandiu as funcionalidades e robustez para um sistema distribuído de compartilhamento de arquivos. Todas as funcionalidades relacionadas ao gerenciamento do status dos peers, busca e download de arquivos e controle de relógio lógico foram desenvolvidas conforme especificado.