Servidor de Mensagens

Sistema Supervisório

Execução: Individual
Data de entrega: 24 de maio de 2022 até 23h:59min

Introdução

Protocolo
- Detalhes de Implementação do Protocolo

Tratamento de Erros

Implementação
- Limites
- Materiais para Consulta

Avaliação
- Correção Semi-automática
- Entrega
- Desconto de Nota por Atraso

INTRODUÇÃO

O gerente de uma indústria siderúrgica está preocupado com a redução abrupta do rendimento em toda sua linha de produção. Ele não sabe o motivo da redução e não dispõe de recursos computacionais automatizados para analisar qual(is) processo(s) está(ão) gerando o problema em razão de quase todos processos da produção da empresa serem manuais. A solução que o gerente encontrou foi implantar um sistema supervisório responsável por coletar e armazenar dados dos equipamentos durante os processos de produção. Tais dados são obtidos a partir dos sensores que monitoram variáveis de processos (velocidade, temperatura, etc) na produção e são armazenados para futuras consultas pela Central de Monitoramento, como ilustra a Figura 1. Os componentes físicos do sistema supervisório são:

- **Sensores**: Responsável por mensurar dados de variáveis de processos e enviá-las para a Estação Remota
- **Rede de comunicação**: Infraestrutura responsável por prover comunicação entre os nós na rede;
- **Estação Remota (o servidor)**: Atende às solicitações da Central de Monitoramento e se comunica diretamente com os sensores distribuídos na indústria
- **Central de Monitoramento (o cliente)**: Responsável gerenciar (adicionar e remover logicamente) os sensores e por solicitar informações dos sensores.

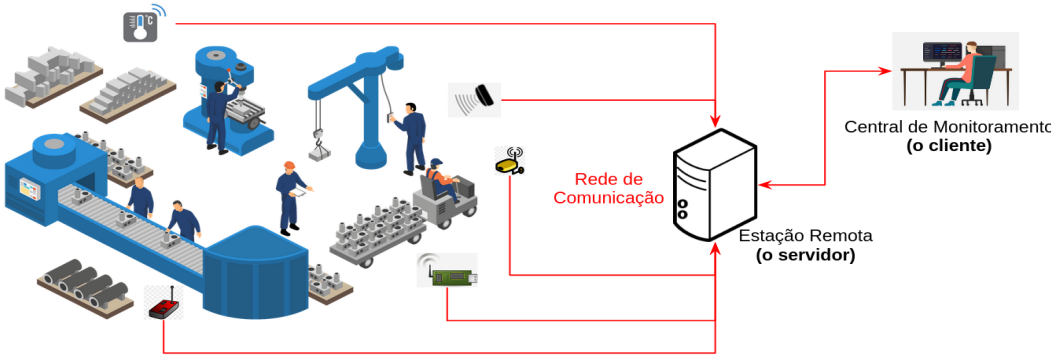

Figura 1 - Exemplo de sistema supervisório em indústria

Nesta indústria, o sistema supervisório deve monitorar apenas quatro tipos de variáveis, **temperatura, pressão, velocidade e corrente** dos equipamentos, proveniente de sensores que possuem uma identificação, como apresenta a **Tabela I**. Ademais, a indústria mantém apenas quatro tipos de equipamentos, esteira, guindaste, ponte rolante e empilhadeira, as quais também possuem identificadores específicos, como apresenta a **Tabela II.**

Sensor	SensorId
Temperatura	01
Pressão	02
Velocidade	03
Corrente	04

Tabela I - Sensores e seus identificadores

Equipamento	equipmentId
Esteira	01
Guindaste	02
Ponte Rolante	03
Empilhadeira	04

Tabela II - Equipamentos e seus identificadores

Diante do exposto, neste trabalho prático, você será responsável por implementar um **sistema modelo cliente servidor** para simular a interação entre a **Estação Remota (o servidor)** e a **Central de Monitoramento (o cliente)**. A Estação Remota deve atender às seguintes solicitações da Central de Monitoramento:

1. **Instalar sensor**: Adiciona um novo sensor à um equipamento na Estação Remota
2. **Remover sensor**: Remove um sensor existente de um equipamento na Estação Remota
3. **Consultar equipamento**: Informa os tipos de sensores instalados no equipamento
4. **Consultar variável de processo**: Solicita dados de um tipo de variável de processo de um equipamento (e.g. pressão, temperatura, velocidade, etc)

Cada uma dessas solicitações correspondem a uma mensagem enviada pela Central de Monitoramento à Estação Remota. Neste trabalho, **você deve implementar os quatro tipos de mensagens propostas**.

PROTOCOLO

A **Central de Monitoramento** poderá executar as seguintes ações: solicitar que seja adicionado ou removido logicamente um sensor de algum equipamento no banco de dados; solicitar alteração de sensor, o que implica em alterar informações do sensor no banco; requisitar leitura de variável de processo de algum equipamento.

A Estação Remota e a Central de Monitoramento trocam mensagens curtas de até 500 bytes utilizando o protocolo TCP. As mensagens carregam textos codificados segundo a tabela ASCII. Apenas letras, números e espaços podem ser transmitidos. Caracteres acentuados e especiais não devem ser transmitidos.

A seguir, estão descritas as ações que devem ser performadas, bem como a formatação de cada tipo de mensagem e a resposta desejada:

- **Instalar Sensor:** a Central de Monitoramento poderá solicitar a adição de um novo sensor em um equipamento. Isso deve ser feito através do comando **“add sensor [SensorId] in [equipmentId]”**, onde os campos **SensorId** e **equipmentId** são identificadores inteiros do tipo do sensor e do equipamento, respectivamente, e são descritos nas Tabelas I e II**.** Após adicionado, a Estação Remota deve retornar a resposta **"sensor [SensorId] added”**. Não deve ser possível adicionar novamente um sensor do mesmo tipo no mesmo equipamento. Caso isso ocorra, o servidor deve retornar uma mensagem **"sensor [SensorId] already exists in [equipmentId]”**. A estação remota só deve manter um total de 15 sensores no banco de dados e caso o limite seja ultrapassado o servidor deve retornar a mensagem **“limit exceeded”.**
- **Remover Sensor:** a Central de Monitoramento poderá solicitar a remoção de um sensor do banco. Essa ação deve ser feita através do comando **“remove sensor [SensorId] in [equipmentId]”**. Após removido, o servidor deve retornar a resposta **"sensor [SensorId] removed”**. Caso a Central de Monitoramento tente remover um

sensor que não exista no equipamento, a Estação Remota deve retornar a mensagem "**sensor [SensorId] does not exist in [equipmentId]”**.

- **Consultar equipamento**: a Central de Monitoramento pode pedir à Estação Remota que envie a lista de sensores instalados em um equipamento. Para isso, deve usar o comando **“list sensors in [equipmentId]”**. A estação Remota deve retornar todos os sensores instalados em **[equipmentId]** no formato **“[SensorId] [SensorId] [SensorId]”**. Caso não haja sensores instalados no equipamento deve retornar **“none”**.
- **Consultar variável de processo:** a Central de Monitoramento pode solicitar à Estação Remota que envie a leitura de um ou mais sensores específicos de um equipamento. Para isso, deve usar o comando **“read [SensorId]$_1$ [SensorId]$_2$ [SensorId]$_3$ … [SensorId]$_n$ in [equipmentId]”**. A Estação Remota deve retornar os valores lidos pelos sensores especificados do equipamento seguindo o formato **“value$_1$ value$_2$ value$_3$ … value$_n$”** e obedecendo a ordem dos sensores**.** Caso um ou mais sensores não estejam instalados no equipamento, o servidor deve retornar **“sensor(s) [SensorId]$_1$ [SensorId]$_2$ [SensorId]$_3$ … [SensorId]$_n$ not installed”**

Exemplo 1:

> add sensor 01 03 in 02
< sensor 01 03 added
> add sensor 01 in 02
< sensor 01 already exists in 02
> add sensor 02 in 02
< sensor 02 added
> list sensors in 02
< 01 03 02
> remove sensor 01 in 02
< sensor 01 removed
> remove sensor 01 in 02
< sensor 01 does not exist in 02
> list sensors in 02
< 03 02

Exemplo 2:

> add sensor 03 in 01
< sensor 03 added
> add sensor 01 in 01
< sensor 01 added
> list sensors in 01
< 03 01
> read 01 03 in 01
< 1.23 3.87
> read 01 in 01
< 2.76
> read 01 04 in 01

< sensor(s) 04 not installed

Detalhes de Implementação do Protocolo:

- As mensagens são terminadas com um caractere de quebra de linha `'\n'`. O caractere nulo `'\0'` para terminação de strings em C *não* deve ser enviado na rede.
- Você deve ser capaz de **adicionar** mais de um sensor por vez com o comando "`add`", lembrando que o fim da mensagem é identificado pelo `'\n'`. Os nomes devem ser separados por um espaço neste caso. Para efeitos de simplificação, considere adicionar no máximo 3 sensores por vez com este comando (sem necessidade de tratamento). Dica: faça o tratamento de strings com a biblioteca `<string.h>`.
    - Nesse caso , o servidor deve enviar todas as respostas em uma mesma linha, conforme mostrado nos exemplos acima.
- A **leituras dos sensores** devem ser representados por números decimais aleatórios entre 0 e 10 com 2 casas decimais (e.g. 2.34, 5.87 ou 10.00)
- O servidor deve desconectar o cliente caso receba uma mensagem com um comando desconhecido (exemplo: "`ad`" em vez de "`add`"), mas não precisa retornar mensagem inválida.
- A Estação Remota deve gerenciar apenas sensores da Tabela I. Desta forma, caso tente adicionar outro tipo de sensor, a Estação Remota deve retornar o erro **"invalid sensor"**
- A Estação Remota deve gerenciar apenas equipamentos da Tabela II. Desta forma, caso tente manipular um sensor em um equipamento inexistente, a Estação Remota deve retornar o erro **"invalid equipment"**
- Para funcionamento do sistema de correção semi-automática (descrito abaixo), seu servidor deve fechar todas as conexões e terminar sua execução ao receber a mensagem **"`kill`"** a qualquer momento.

TRATAMENTO DE ERROS

Os tratamentos de erros especificados a seguir correspondem à manipulação de múltiplos sensores no equipamento.

Instalar sensores:

- Caso cliente solicite adição de múltiplos sensores em que alguns já encontram-se instalados no equipamento, servidor deve adicionar sensores não instalados e informar sensores adicionados e não existentes com a mensagem:

  "sensor [SensorId] added [SensorId] already exists in [equipmentId]"

Exemplo: *(Supondo que exista apenas sensor 01 no equipamento 02)*
> add sensor 01 03 in 02
> sensor 03 added 01 already exists in 02

- Caso cliente solicite adição de um número de sensores que ultrapassem limite máximo do banco de dados (15 sensores), servidor deve responder:
  “limit exceeded”

Exemplo: *(Supondo que existam 14 sensores no banco de dados)*
> add sensor 01 03 in 01
< limit exceeded

Remover sensores

+ Caso cliente solicite remoção de múltiplos sensores em que alguns não encontram-se instalados no equipamento, servidor deve remover sensores instalados e informar sensores removidos e não existentes com a mensagem:
  "sensor [SensorId] removed [SensorId] does not exist in [equipmentId]”

Exemplo: *(Supondo que exista apenas sensor 01 no equipamento 02)*
> remove sensor 01 03 in 02
< sensor 01 removed 03 does not exist in 02

Leitura de sensores

+ Caso cliente solicite leitura de múltiplos sensores em que alguns não encontram-se instalados no equipamento, servidor deve informar leitura dos sensores instalados e os IDs dos não instalados, com a mensagem:
  “value$_1$ … value$_n$ and [SensorId]$_1$ … [SensorId]$_n$ not installed”

Exemplo: *(Supondo que exista apenas sensor 01 no equipamento 02)*
> read 01 03 in 02
< `1.23` and 03 not installed

IMPLEMENTAÇÃO

O aluno deve implementar tanto uma versão do servidor (Estação Remota) quanto uma versão do cliente (Central de Monitoramento). Ambos devem utilizar o protocolo TCP, criado com **[socket(AF_INET, SOCK_STREAM, 0)]** ou com **[socket(AF_INET6, SOCK_STREAM, 0)]**. Deve ser possível utilizar tanto o IPv4 quanto o IPv6.

O **cliente** deve receber mensagens do teclado e imprimir as mensagens recebidas na tela. O **servidor** deve imprimir na saída padrão todas as mensagens recebidas dos clientes. **Não é necessário** que o servidor aceite mais de um cliente simultaneamente.

Seu servidor deve receber, **estritamente nessa ordem**, o tipo de endereço que será utilizado (**v4** para IPv4 ou **v6** para IPv6) e um número de porta na linha de comando especificando em qual porta ele vai receber conexões (Sugestão: utilize a porta 51511 para efeitos de padronização do trabalho). Seu cliente deve receber, **estritamente nessa ordem**, o endereço IP e a porta do servidor para estabelecimento da conexão.

A seguir, um exemplo de execução dos programas em dois terminais distintos:

IPv4:

```
no terminal 1: ./server v4 51511
no terminal 2: ./client 127.0.0.1 51511
```

IPv6:

```
no terminal 1: ./server v6 51511
no terminal 2: ./client ::1 51511
```

O servidor pode dar **bind** em todos os endereços IP associados às suas interfaces usando a constante **INADDR_ANY** para IPv4 ou **in6addr_any** para IPv6.

Limites:

- Cada mensagem possui no máximo 500 bytes.
- A Estação Remota deve manter até 15 sensores

Materiais para Consulta:

- Capítulo 2 e 3 do livro sobre programação com sockets disponibilizado no Moodle.
- Playlist de programação com sockets.

AVALIAÇÃO

O trabalho deve ser realizado individualmente e **deve ser implementado com a linguagem de programação C** utilizando somente a biblioteca padrão (interface POSIX de sockets de redes). Deve ser possível executar seu programa no sistema operacional **Linux** e **não deve utilizar bibliotecas Windows, como o winsock**. Seu programa deve interoperar com qualquer outro programa implementando o mesmo protocolo (você pode testar com as implementações dos seus colegas). Procure escrever seu código de maneira clara, com comentários pontuais e bem indentados. Isto facilita a correção dos monitores e tem impacto positivo na avaliação.

Correção Semi-automática

Seu servidor será corrigido de forma semi-automática por uma bateria de testes. Cada teste verifica uma funcionalidade específica do servidor. O seu servidor será testado por um cliente implementado pelo professor com funcionalidades adicionais para realização dos testes. Os testes avaliam a aderência do seu servidor ao protocolo de comunicação inteiramente através dos dados trocados através da rede (a saída do seu servidor na tela, e.g., para depuração, não impacta os resultados dos testes).

Para a correção os seguintes testes serão realizados **(com IPv4 e IPv6)**:

- Adicionar sensores: **+2 pontos**
- Remover sensor: **+2 pontos**

- Consultar equipamentos: **+3 pontos**
- Consultar variável de processos: **+3 pontos**
- Testar casos de mensagem inválida: **+1 ponto**
- Cliente envia kill para o servidor e fecha a execução: **+1 ponto**

Obs.1: Caso os testes funcionem em apenas um tipo de endereço (IPv4 ou IPv6), a pontuação do respectivo teste será reduzida pela metade.

Obs.2: Considere para cada cenário acima todas as possibilidades possíveis (adicionar um sensor que já existe no equipamento, remover um sensor que não existe no equipamento, etc).

Obs.3: Não é necessário fazer tratamento para overflow de mensagens.

Note que apesar do programa cliente não ser avaliado no conjunto de testes, ele ainda será avaliado manualmente pelo monitor.

Entrega

Cada aluno deve entregar documentação em PDF de até 4 páginas (duas folhas), sem capa, utilizando fonte tamanho 10, e figuras de tamanho adequado ao tamanho da fonte. A documentação deve discutir desafios, dificuldades e imprevistos do projeto, bem como as soluções adotadas para os problemas. A documentação corresponde a 20% dos pontos do trabalho (3 pontos), mas só será considerada para as funcionalidades implementadas corretamente.

Será utilizado um sistema para detecção de código repetido, portanto não é admitido cola de trabalhos.

Será adotada média harmônica entre as notas da documentação e da execução, o que implica que a nota final será 0 se uma das partes não for apresentada.

Cada aluno deve entregar, além da documentação, o **código fonte em C** e um **Makefile** para compilação do programa. Instruções para submissão e compatibilidade com o sistema de correção semi-automática:

- O Makefile deve compilar o cliente em um binário chamado “`client`” e o servidor em um binário chamado “`server`”.
- Seu programa deve ser compilado ao se executar apenas o comando “`make`”, ou seja, sem a necessidade de parâmetros adicionais.
- A entrega deve ser feita no formato ZIP, com o nome seguindo o seguinte padrão: TP1_MATRICULA.zip
- Os nomes dos arquivos devem ser padronizado:
    - `server.c`
    - `client.c`
    - `common.c, common.h (se houver)`

Desconto de Nota por Atraso

Os trabalhos poderão ser entregues até a meia-noite do dia especificado para a entrega. O horário de entrega deve respeitar o relógio do sistema Moodle, ou seja, a partir de 00:01 do dia seguinte à entrega no relógio do Moodle, os trabalhos já estarão sujeitos a penalidades.

A fórmula para desconto por atraso na entrega do trabalho prático é:

$$desconto = 2^{d} - 1$$

onde *d* é o atraso em dias úteis. Note que após 3 dias, o trabalho não pode ser mais entregue.