Introdução aos Sistemas Distribuídos

Aula 1 * Sistemas Distribuídos

Prof. Elias P. Duarte Jr.
Universidade Federal do Paraná (UFPR)
Departamento de Informática
www.inf.ufpr.br/elias/sisdis

Sumário da Aula de Hoje

- Bem vindos!
- Como vai ser a disciplina (ERE4, iniciada 20/09/21)
- O que é um sistema distribuído? Vamos definir exatamente!
- 2 Paradigmas: *Troca de Mensages & Memória Compartilhada*
- 2 Topologias: *Fully Connected & Multi-Hop*
- 3 Primitivas de comunicação: *send(), receive(), deliver()*
- Modelos Temporais: Sistemas Síncronos, Assíncronos e Parcialmente Síncronos

Bem vindos!

- Como vai ser esta disciplina: 2a vez on-line!
- Disciplina em 12 semanas:
  - 2 provas (Prova1, Prova2) mais ProvaFinal
  - 2 Trabalhos Práticos: simulação
    - habilidade importante que você aprende em Sistemas Distribuídos – implementar simuladores
- Trabalhos Práticos vão ter peso: 40% da nota & Provas 60%
- Provas e trabalhos serão marcados oportunamente (impossível saber de antemão!)

Como vai ser a disciplina

- Proposta é seguir todas as segundas e quartas, exceto feriados com a prova 2 no dia 13 de dezembro de 2021
- Vamos começar 19:30 e ir *no máximo* até 21 horas
- Minha proposta é cada aula ser um tópico em si, acabando o tópico acabou a aula
- Chamada será feita todas as aulas! Sua presença e participação são importantes!

Referências

- Os slides serão disponibilizados para os alunos
- As aulas também vão sendo divulgadas para os alunos
- A disciplina não tem um livro texto com tudo
- Mas um deles é uma base importante:
  - "Introduction to Reliable Distributed Programming" que está disponível em versão digital na Biblioteca da UFPR (instruções para baixar www.inf.ufpr.br/elias/sisdis)
- Artigos dos diversos tópicos serão também divulgados

A Importância dos Sistemas Distribuídos

- As redes de computadores e a Internet representam uma verdadeira revolução para a humanidade...
- ... e todo software que executa sobre a rede é um sistema distribuído
- Varia desde aplicações específicas...
  - por exemplo o sistema que você usa para acessar o seu banco
- ... até sistemas de propósito geral
  - por exemplo um sistema gerenciador de banco de dados distribuído

O que é exatamente um sistema distribuído?

- Alerta! várias definições “por aí” :-0
- Inclusive livros com definições divergentes...
- Assim é muito importante definirmos precisamente o que é um sistema distribuído no contexto desta disciplina

DEFINIÇÃO: Um sistema distribuído é um conjunto de *processos* que se comunicam e cooperam para resolver uma tarefa específica.

Processo

- O que é exatamente um processo?

Processo

- O que é exatamente um processo?
- Um processo é um programa em execução

Processo

- O que é exatamente um processo?
- Um processo é um programa em execução
- Em que um processo difere de um programa?

Processo

- O que é exatamente um processo?
- Um processo é um programa em execução
- Em que um processo difere de um programa?
- Vamos pensar nos componentes de processo:
  - O próprio programa (sequência de instruções)
  - O PC (Program Counter)
  - O estado da memória e registradores
  - O estado das conexões TCP, etc. etc. etc.

Sistema Distribuído

- Sistema distribuído consiste de um conjunto de processos

  $S = \{P_1, P_2, \ldots, P_N\}$

- A expressão acima define o sistema distribuído como contendo um número pré-determinado de processos: N
- Este tipo de sistema distribuído é dito estático: consiste de um conjunto conhecido e constante de processos
- Por outro lado, sistemas distribuídos dinâmicos tem composição dinâmica

Estáticos Vs. Dinâmicos

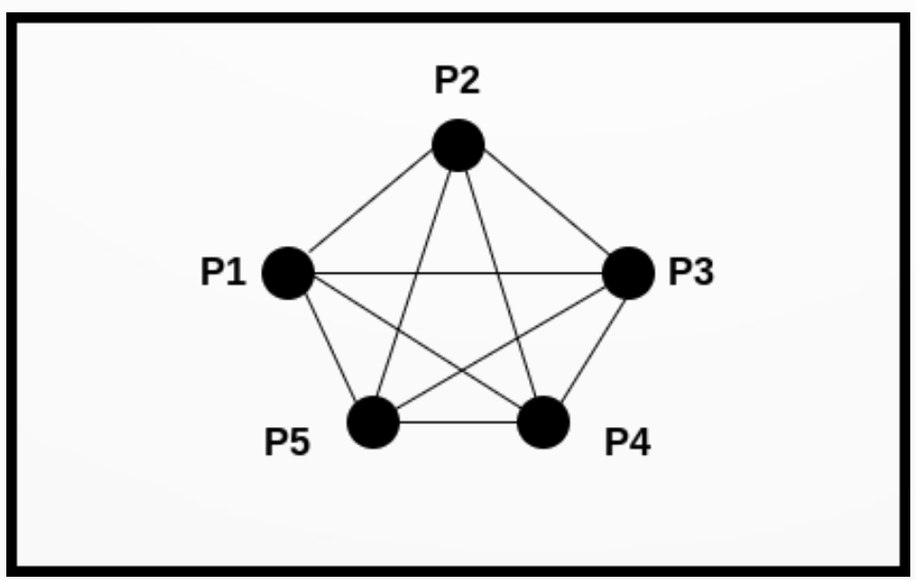
- Nesta disciplina vamos tratar exclusivamente de sistemas distribuídos estáticos
  - Consistem de um conjunto de N processos conhecidos desde o início
- Nos sistemas distribuídos dinâmicos, processos podem sair e novos processos podem entrar no sistema (exemplo: sistemas P2P)
- A composição de um sistema distribuído dinâmico varia com o tempo

Topologia do Sistema Distribuído

- Os sistemas distribuídos podem ter duas topologias principais
  - *Fully Connected* (“totalmente conectado”)
  - *Multi-Hop* (“topologia arbitrária”)

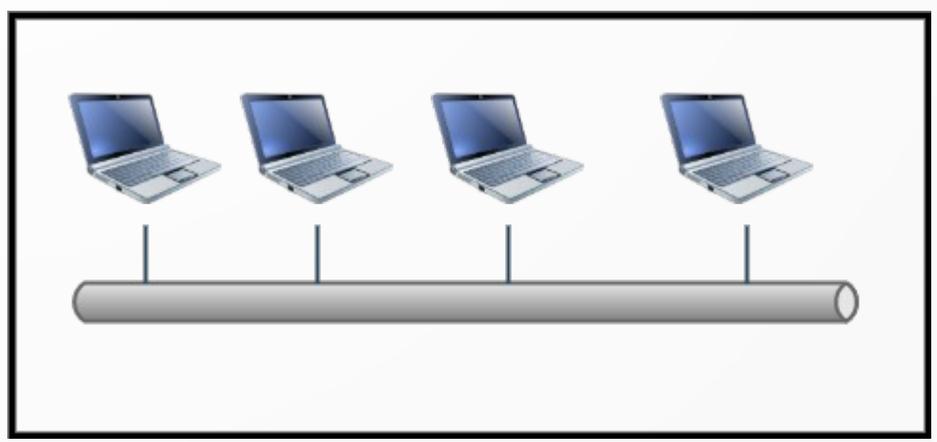
Sistemas *Fully-Connected*

- Sistemas *fully-connected são* representáveis por um grafo completo
- Todos os processos tem um canal de comunicação direto com todos os outros processos
- Não precisam de “intermediários” para comunicar



Sistemas *Fully Connected*

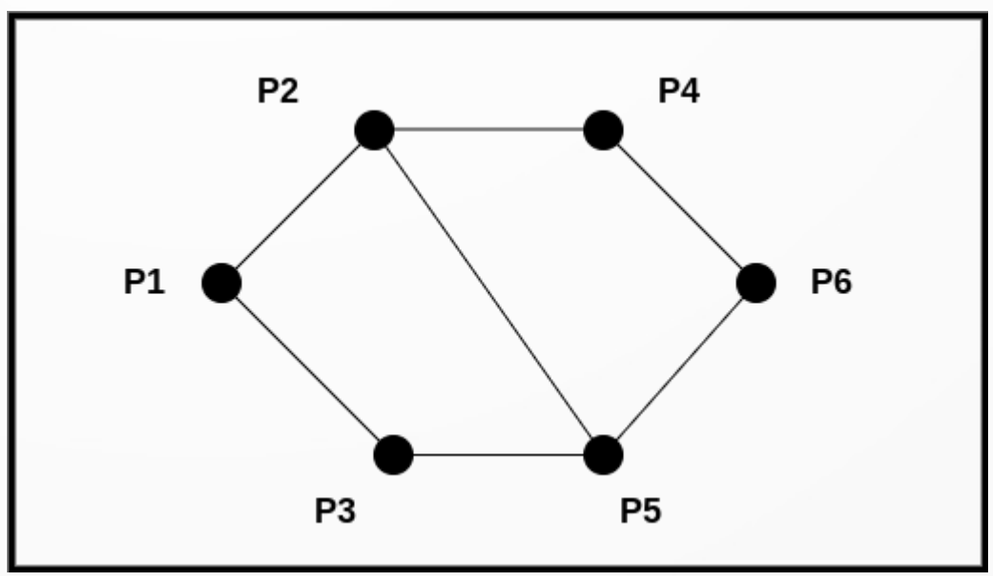
- Canais de comunicação não são enlaces físicos individuais!
- Na verdade: uma rede local corresponde a um sistema *fully-connected*



Em uma rede Ethernet, um processo em execução em qualquer host se comunica diretamente com qualquer outro processo em qualquer outro host, sem passar por intermediários.

Sistemas Multi-Hop

- Não há um canal de comunicação direto entre todos os pares de processos do sistema distribuído
- Em alguns casos é necessário passar por um ou mais intermediários
- Envolve roteamento



Comunicação de Processos

- Os processos se comunicam para cooperar e resolver uma tarefa
- Como se comunicam?

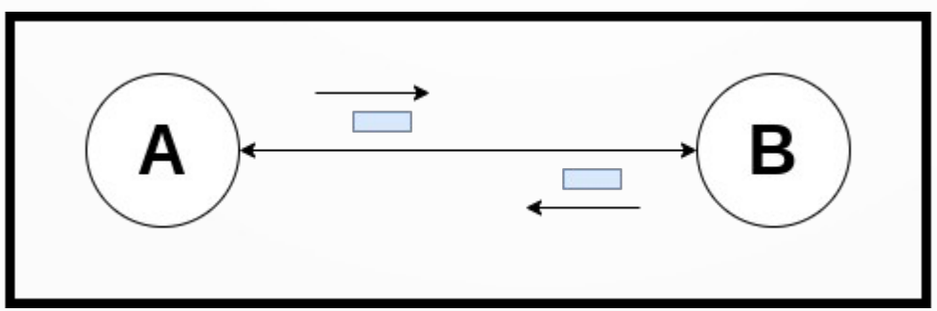
Paradigmas de Comunicação de Processos

- Existem dois grandes paradigmas de Sistemas Distribuídos
- O que diferencia um do outro é a forma como os processos se comunicam:

1) Troca de Mensagens (*Message Passing*)
2) Memória Compartilhada (*Shared Memory*)

Troca de Mensages

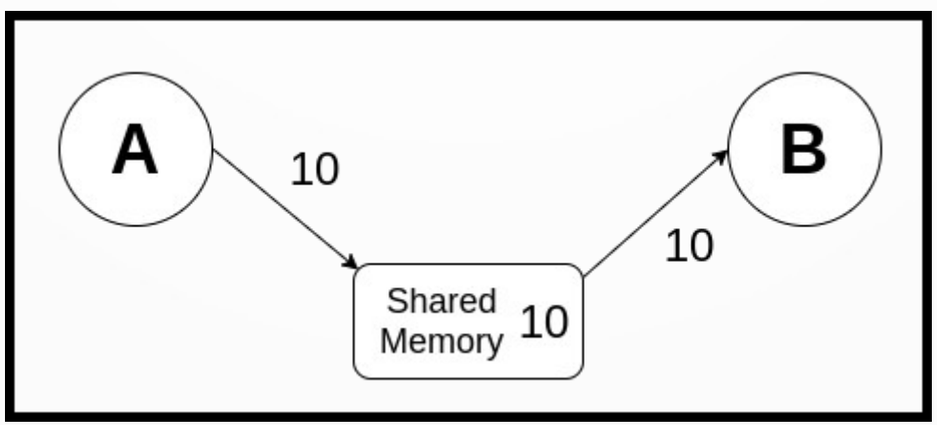
- Os processos comunicam transmitindo/recebendo mensagens através de uma rede



Dois processos A e B trocam mensagens através de um canal de comunicação.

Memória Compartilhada

- Os processos compartilham memória através da qual se comunicam, escrevendo e lendo dados



O processo A escreve o valor 10 na memória compartilhada que é lido pelo processo B.

Primitivas para Troca de Mensagens

- Considere dois processos conectados por um canal de comunicação
- As primitivas utilizadas na comunicação de mensagens são:

  send(msg): primitiva utilizada para o envio de uma mensagem (*msg*)

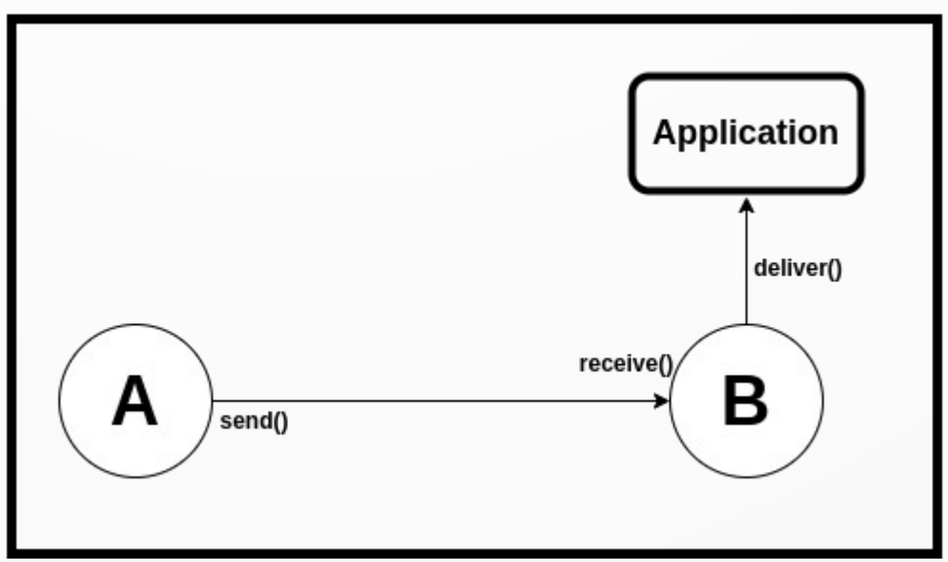
  receive(msg): primitiva utilizada no recebimento da mensagem (*msg*)

  e uma terceira primitiva

A Primitiva *Deliver*

- A terceira primitida é executada pelo processo destino

  deliver(msg): primitiva utilizada pelo destino para a entrega da mensagem recebida à aplicação



A Primitiva ***Deliver***

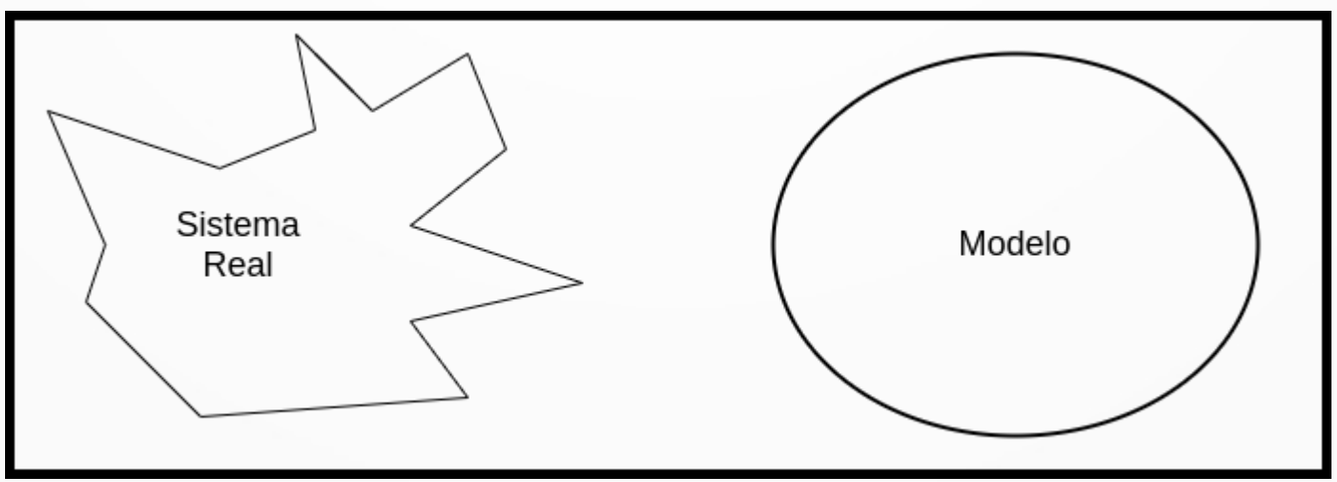
- O ponto é: após receber uma mensagem, o processo pode ainda decidir se ela deve ser entregue à aplicação ou descartada
- Exemplo: pode só entregar depois de garantir que todos os processos do sistema vão entregar
- O processo de aplicação é o “usuário” do sistema

Primitivas de Comunicação

- As primitivas são utilizadas para programar os sistemas distribuídos
- Diferentes linguagens/sistemas operacionais oferecem diferentes primitivas
- A biblioteca *socket* é praticamente um padrão, e tem as duas primitivas *send(msg)* e *receive(msg)*
- A primitiva *deliver(msg)* é executada interna e localmente pelo processo destinatário da *msg*

Modelos

- Um modelo é uma forma simplificada de representar um sistema real



Pensando sobre Modelos

- Os modelos são extremamente importantes na Ciência da Computação e em outras áreas
- Um modelo é *sempre* uma simplificação de um sistema real
- Muitas vezes o sistema real é muito complexo para ser estudado
- É importante que o modelo tenha aquelas características do sistema que permitam a análise e a chegada a conclusões corretas

Modelos de Sistemas Distribuídos

- Os modelos de sistemas distribuídos são extremamente importantes
- Por exemplo: a maioria dos algoritmos distribuídos vem em versões, que mudam umas das outras de acordo com o modelo
- Em alguns ambientes cabe um modelo, em outros ambientes outro
- Por exemplo: o modelo adequado para a Internet é diferente do modelo para uma rede industrial

Os Dois Modelos Mais Importantes de Sistemas Distribuídos

- Modelo temporal
- Modelo de falhas
- *Toda vez que se fala sobre um sistema distribuído específico é essencial mencionar em qual modelo temporal e em qual modelo de falhas ele se encaixa*

O Tempo e os Sistemas Distribuídos

- Existem dois modelos temporais básicos:
  - Sistemas Síncronos
  - Sistemas Assíncronos
- A classificação leva em conta dois fatores:
  - O tempo para transmitir uma mensagem (desde que é transmitida pelo processo origem até ser recebida pelo processo destino)
  - O tempo que o processo leva para executar uma tarefa

Sistemas Síncronos

- Um sistema é síncrono se existem limites de tempo **conhecidos** para
  - A transmissão de uma mensagem entre dois processos
  - A execução de uma tarefa por uma processo
- Um ponto causa muita confusão: o limite tem que ser respeitado 100% das vezes
  - Se respeitar 99.999999% das vezes mas furar em alguns casos [ainda que raros!] o sistema *não* é síncrono

Sistemas Assíncronos

- Em um sistema assíncrono *não* há limites conhecidos nem para o tempo de transmissão de mensagens, nem para o tempo de execução de tarefas
- Na verdade vai além (apesar de que a definição acima é muito usada): um sistema assíncrono é definido livre de qualquer premissa temporal
  - Pois é definido sem qualquer noção de tempo

Vamos pensar sobre estas definições?

- A Internet, por exemplo, é síncrona?

Vamos pensar sobre estas definições?

- A Internet, por exemplo, é síncrona?
- Não é! Com certeza! Basta pensar quando você tenta acessar um servidor recebe a seguinte resposta indicando insucesso:

→ Receber esta mensagem não é garantia que o servidor Web contactado está falho... a resposta pode não ter chegado ainda

Pensando sobre Sistemas Síncronos e Assíncronos

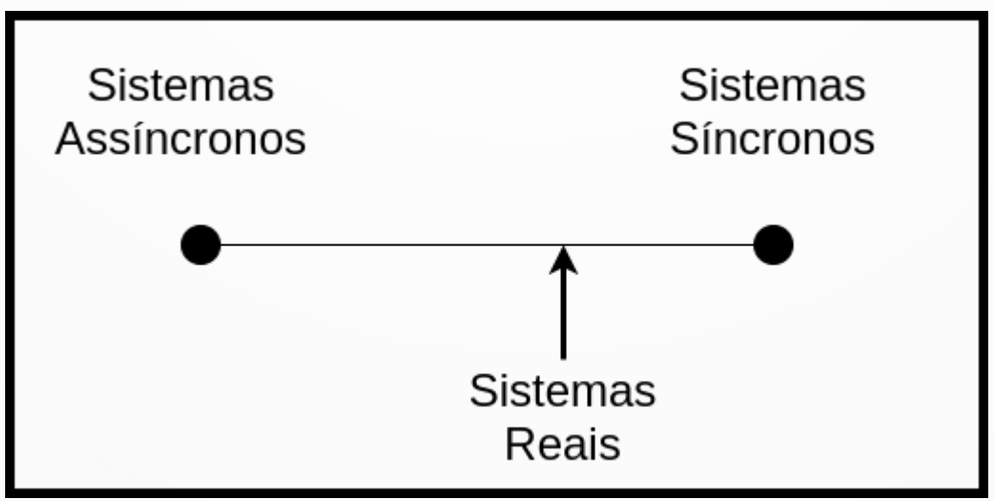
- Por outro lado: será que a Internet é puramente assíncrona, no sentido que nenhum limite temporal pode ser estabelecido?
  - Apesar de não ter limites conhecidos, nós conseguimos ter uma boa noção do comportamento temporal dos processos que se comunicam na Internet

Síncrono e Assíncrono: Extremos de u Segmento de Possibilidades

- Desta forma, não podemos dizer nem que a Internet é síncrona, nem que é assíncrona
  - Observe que se você disser que é assíncrona não está cometendo um “erro” propriamente dito
- É possível dizer que os modelos síncrono e assíncrono são extremos de um segmento de possibilidades,
- Os sistemas reais (como a Internet) estão em algum lugar no meio do segmento

Sistemas Reais, Síncronos e Assíncronos

- Sistemas reais no contexto de extremos: sistemas síncronos e assíncronos



Modelos Parcialmente Síncronos

- Diversos modelos intermediários foram propostos, mas nenhum deles é perfeito...
- ... no sentido de que reflete com precisão os aspectos temporais dos sistemas distribuídos reais
- Variam desde alterações simples (p.ex. “limite do tempo de transmissão de mensagens conhecido mas tempo de execução de mensagens desconhecido”) até alternativas mais elaboradas

GST: *Global Stabilization Time*

- Este é o modelo mais usado hoje pela comunidade de sistemas distribuídos
- É um modelo parcialmente síncrono
- Assume-se que o sistema inicia assíncrono, não respeitando qualquer limite conhecido
- MAS, a partir de um instante de tempo (justamente denominado GST) o sistema passa a se comportar como síncrono *para sempre*

“Para sempre”

- Esta expressão é muito comum em sistemas distribuídos (em inglês: *forever*)
- Não quer dizer até o final dos tempos ;-)
- Simplesmente: até o final da execução do sistema ou do algoritmo sendo especificado

E se a premissa não for respeitada?

- O que acontece se você executar um algoritmo proposto para o modelo GST e o sistema não fica perfeitamente síncrono a partir de um instante de tempo até o final?
- Resposta: o algoritmo pode não funcionar
- O modelo é, de certa forma, um “contrato” que diz como deve ser o sistema real para que o algoritmo funcione

Conclusão da Aula 1

- Hoje definimos como vai ser a disciplina
- Depois: definição precisa do que é um Sistema Distribuído
- Sistemas Estáticos & Dinâmicos
- Paradigmas de comunicação: troca de mensagens e memória compartilhada
- Topologias: *fully-connected & multi-hop*

Conclusão da Aula 1

- Primitivas de comunicação: *send(), receive(), deliver()*
- Modelos Temporais: síncrono, assíncrono, parcialmente síncronos
- Próxima aula: tolerância a falhas e modelos de falhas

Obrigado!

Lembrando: a página da disciplina é:

www.inf.ufpr.br/elias/sisdis