

# ENGENHARIA DE SOFTWARE III

---



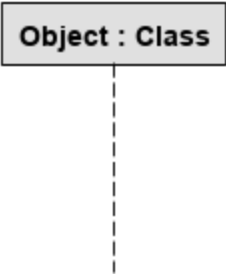
Prof. Me. Warner Brezolin  
wbrezolin@gmail.com


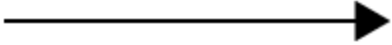

# Diagrama de Sequência

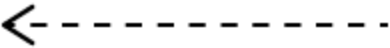

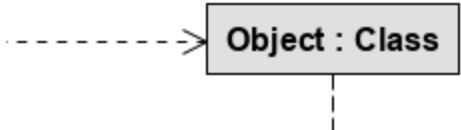
Esse diagrama descreve a ordem dos eventos e as trocas de mensagens entre os objetos do sistema.


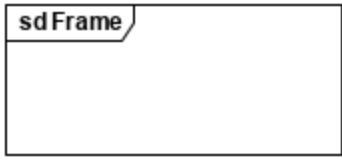

- Baseia-se no **Diagrama de Casos de Uso**, pois cada **Caso de Uso** geralmente possui um diagrama de sequência correspondente.
- Cada **Caso de Uso** envolve um **ator** que dispara um processo e uma sequência de eventos/mensagens.
- Também utiliza informações do **Diagrama de Classes**, já que os objetos e suas interações são representados com base nele.

**Exemplo prático:** Suponha um sistema de compra online. Um cliente (ator) inicia uma compra, o sistema exibe os produtos, o cliente seleciona um item, e o pedido é processado. O diagrama de sequência deve mostrar a ordem exata de mensagens entre o cliente e o sistema: do pedido até a confirmação.

Notação	Elemento
	<p><b>Linha de vida:</b> representa a existência de um elemento (objeto ou ator) participante da realização do caso de uso em um período de tempo. É representada por uma linha vertical tracejada abaixo do elemento, chamada de cauda.</p>
	<p><b>Ator:</b> representa os mesmos atores já criados no diagrama de casos de uso; são apoiados por uma linha de vida e enviam mensagens para os objetos como uma forma de interação para solicitarem a execução de uma operação ou simplesmente o envio de informações. No diagrama sempre representa o ator primário responsável por enviar a mensagem inicial, que começa a interação entre os objetos.</p>
	<p><b>Objeto:</b> representa os objetos que participam da realização do caso de uso; são também apoiados por uma linha de vida, que juntamente com os atores, formam um cabeçalho para o diagrama. Um objeto pode existir desde o início da interação ou ser criado ao longo da dela. Um objeto é representado por um retângulo com um nome único, conforme o padrão da notação de objeto.</p>

Notação	Elemento
	<p><b>Foco de controle:</b> representa o período de tempo durante o qual um elemento executa uma ação, diretamente ou não. É representado por um retângulo estreito na vertical sobre a linha de vida, podendo aparecer diversas vezes ao longo dela.</p>
	<p><b>Mensagem síncrona:</b> a mensagem é síncrona quando o emissor aguarda o retorno para continuar com a interação. São as mensagens comumente utilizadas no diagrama de sequência. É representada por uma linha horizontal com uma seta sólida na extremidade.</p>
	<p><b>Mensagem assíncrona:</b> a mensagem é assíncrona quando o emissor continua enviando mensagens sem aguardar o retorno, com isso o elemento receptor da mensagem assíncrona não precisa atendê-la imediatamente. É representada por uma linha horizontal com uma seta aberta.</p>

Notação	Elemento
	<p><b>Mensagem de retorno:</b> é uma mensagem que um objeto envia ao outro em resposta à mensagem recebida após a execução de uma ação. As mensagens de retorno são representadas por uma linha tracejada com uma seta na extremidade, apontando para o elemento que recebe a resposta.</p>
	<p><b>Mensagem reflexiva ou automensagem:</b> é uma mensagem indicativa de que o objeto remetente da mensagem é também o receptor. A mensagem reflexiva é representada por uma seta que sai do objeto e retorna para ele mesmo.</p>
	<p><b>Mensagem construtora:</b> indica o momento em que o objeto passa a existir no sistema, ou seja, o objeto é instanciado ao longo do processo por uma mensagem enviada. A mensagem construtora é representada por uma linha tracejada com seta na extremidade, apontando para o centro do objeto criado. O retângulo que representa o objeto é posicionado mais abaixo no diagrama.</p>

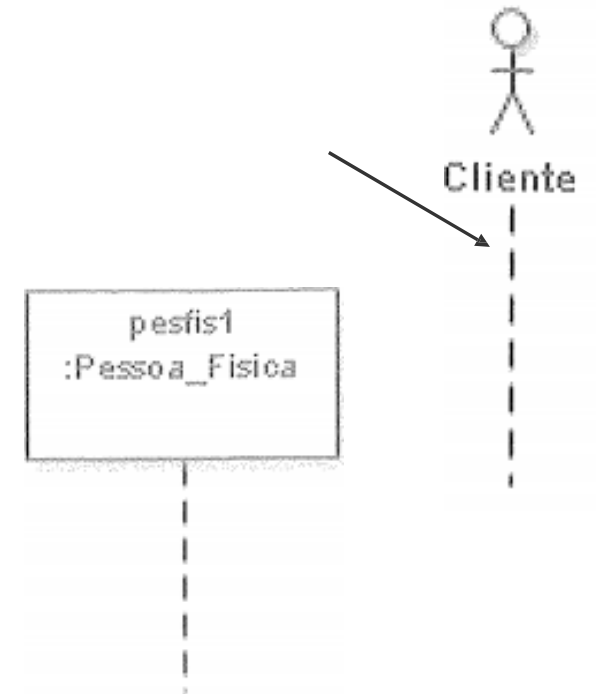
Notação	Elemento
	<p><b>Mensagem destrutora:</b> indica a destruição do objeto no decorrer da interação, o qual não se mostra mais necessário no processo. É representado pelo símbolo X na parte inferior da linha de vida do objeto que está sendo destruído.</p>
	<p><b>Quadro de interação:</b> representa uma interação independente, possibilitando isolar um diagrama de sequência com um contexto específico, formando uma fronteira para que possa ser integrado aos demais diagramas de sequência. O quadro é representado graficamente por um retângulo com um rótulo no canto superior esquerdo, que identifica o tipo do quadro.</p>
	<p><b>Nota ou comentário:</b> serve para escrever observações aos elementos que compõe o diagrama de sequência, contendo informações úteis para os desenvolvedores, porém não expressa força semântica específica aos elementos do diagrama.</p>

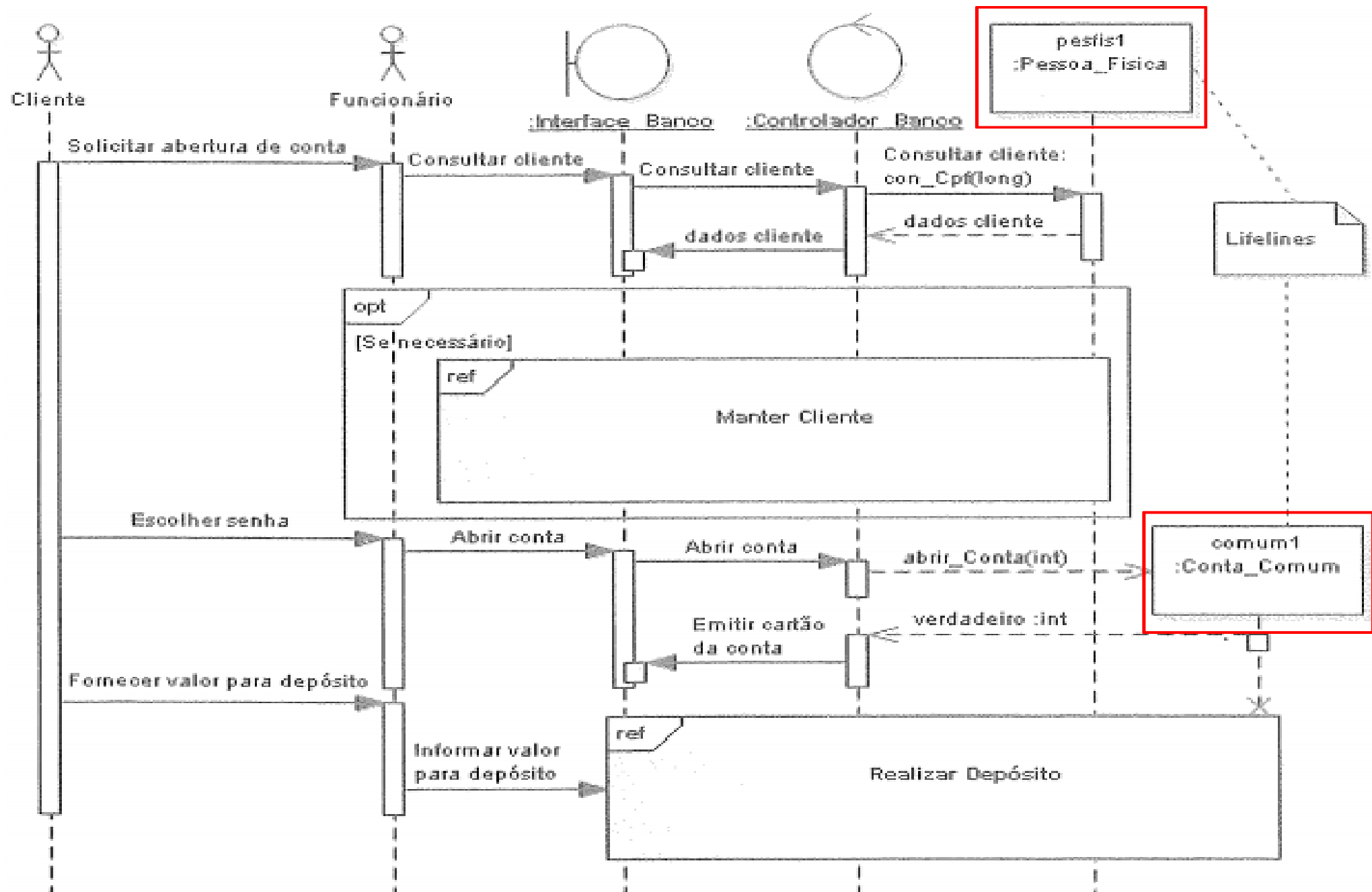
# Diagrama de Sequência

## Atores no Diagrama de Sequência

- Os **atores** são os mesmos definidos no **Diagrama de Casos de Uso**.
- Cada ator possui uma **linha de vida** ("lifeline"), que mostra sua existência ao longo do tempo.
- A **linha de vida** pode começar no início do diagrama ou ser criada a partir de um ponto específico.

**Exemplo prático:** Em um sistema de suporte técnico, um cliente (ator) inicia uma solicitação, e a linha de vida dele começa naquele ponto. Outro ator, como um técnico de suporte, pode ser incluído no processo em um momento posterior, quando a solicitação for escalada.







# Diagrama de Sequência

## Linha de Vida (Lifeline)

- A **linha de vida** mostra o tempo de existência de um objeto no processo.
- A linha termina com um '**X**' quando o objeto é destruído.
- Objetos podem ser criados apenas quando são necessários (não aparecem no início do diagrama).
- Objetos devem ser destruídos quando não forem mais usados.

**Exemplo prático:** Em um sistema de reserva de hotel, o objeto "Reserva" pode ser criado quando o cliente inicia o processo de reserva. A linha de vida desse objeto termina com um 'X' após a confirmação ou cancelamento da reserva, indicando sua destruição.

# Diagrama de Sequência

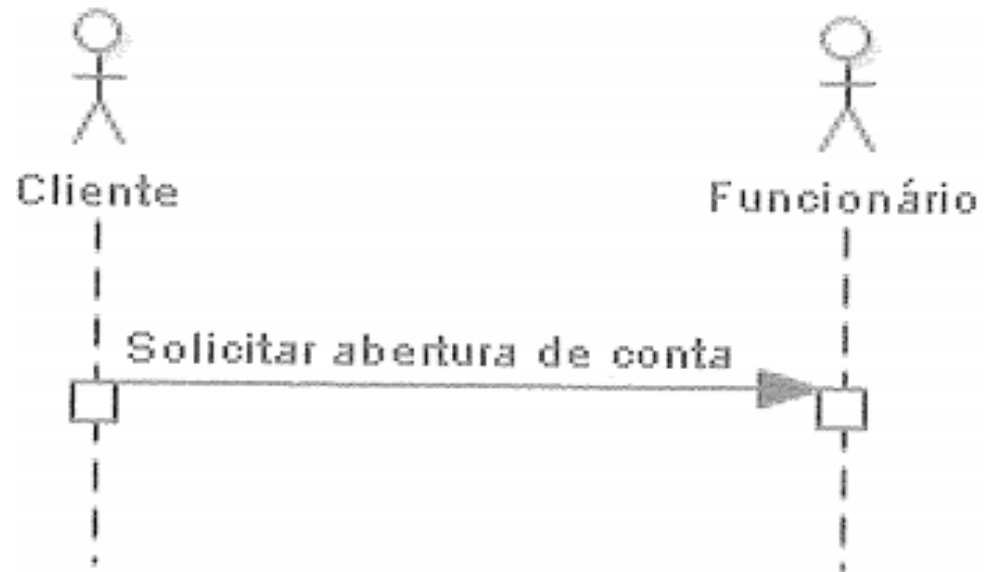
## Mensagens

- Representam eventos, como uma **chamada de método** ou uma **comunicação** entre dois atores.
- Podem ocorrer de várias formas:
  - **Ator ↔ Ator**
  - **Ator ↔ Objeto**
  - **Objeto ↔ Objeto**
  - **Objeto ↔ Ator**

**Exemplo prático:** Em um sistema de banco online, um cliente (ator) solicita um saldo (mensagem para o objeto "Conta"). A conta retorna o saldo ao cliente. Se houver comunicação entre dois clientes, como em uma transferência, isso será representado como mensagem entre dois atores.

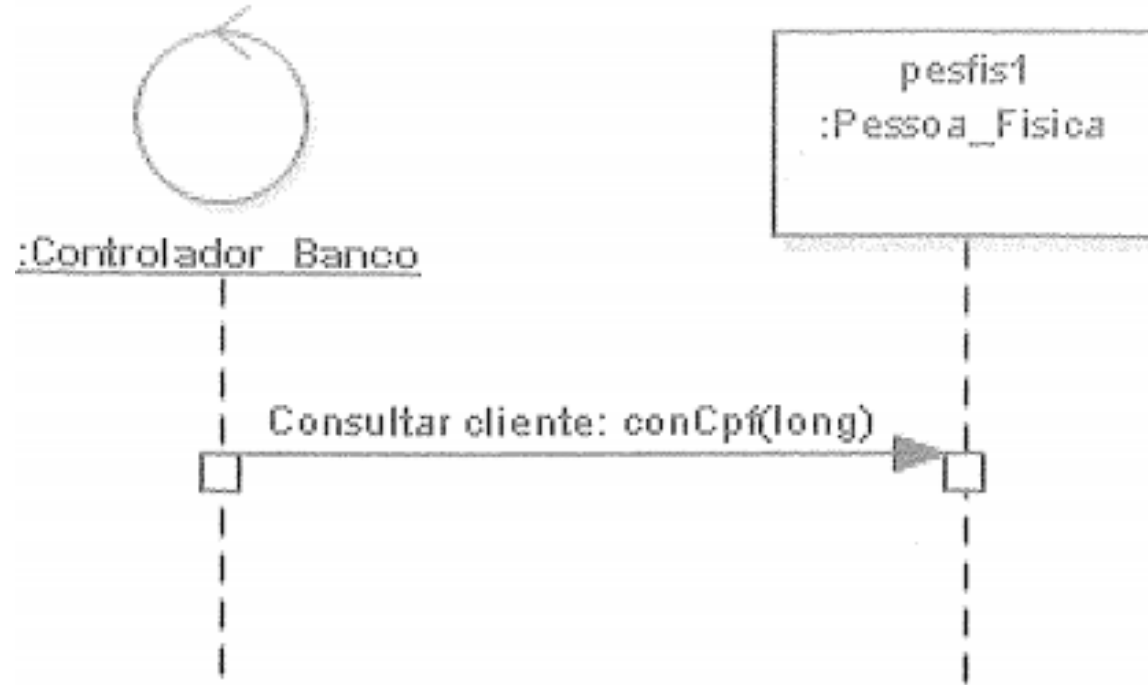
# Diagrama de Sequência

**Disparo de método entre Atores:**



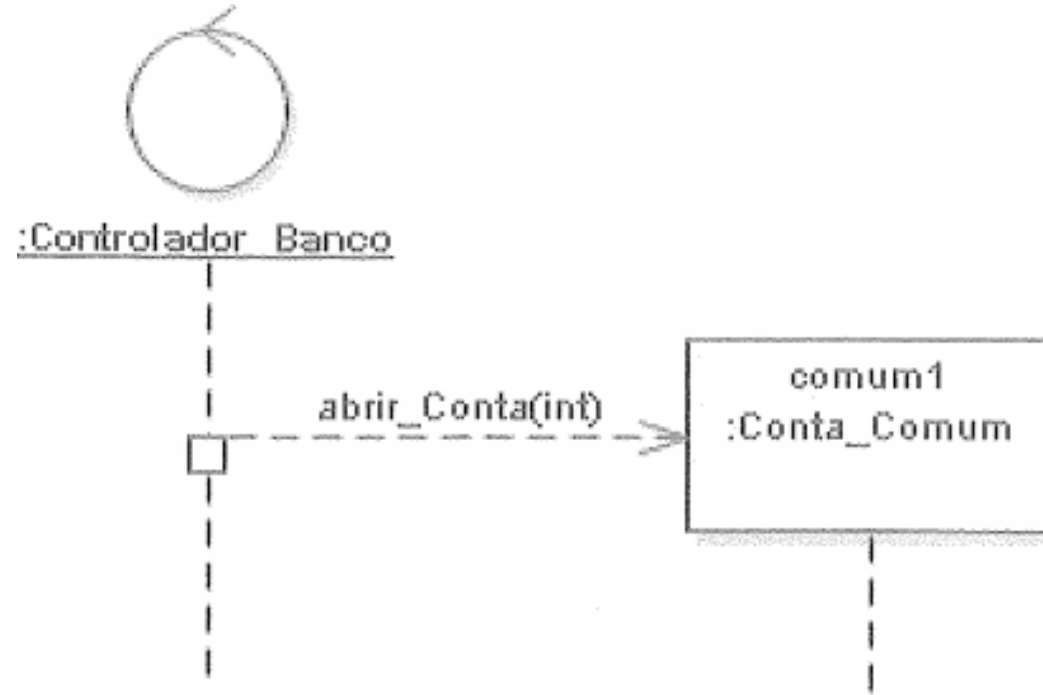
# Diagrama de Sequência

Disparo de método entre objetos:



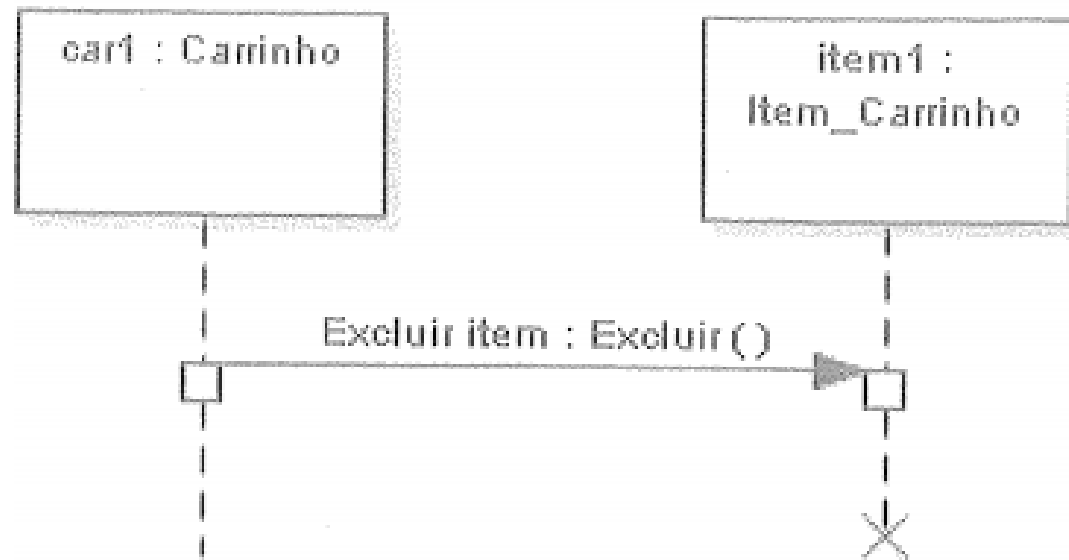
# Diagrama de Sequência

Criação de objeto:



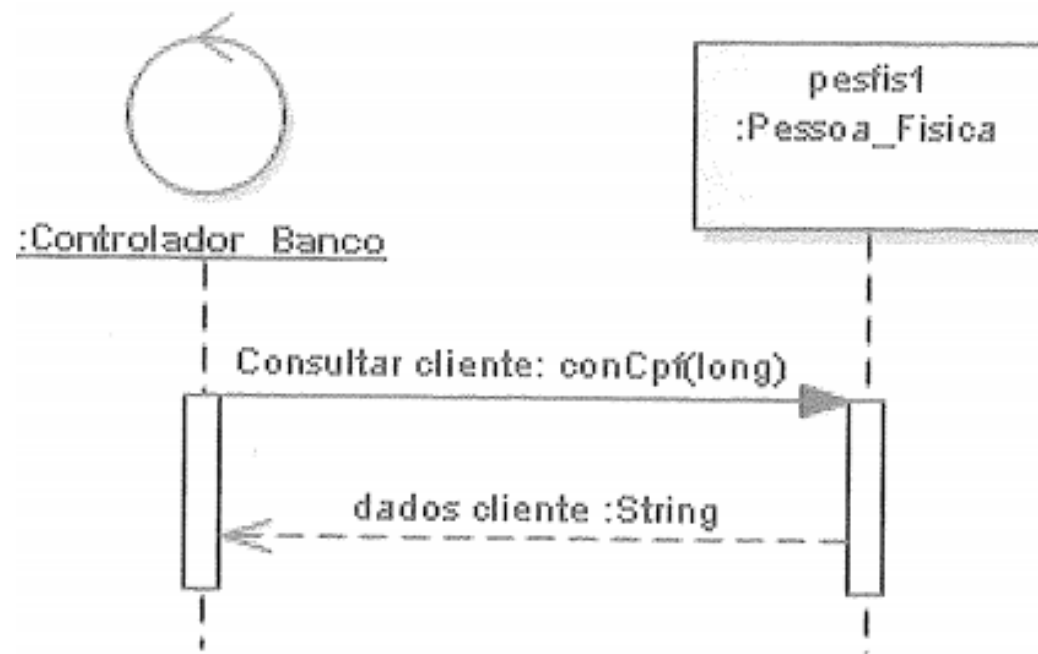
# Diagrama de Sequência

**Destruição de objeto:**



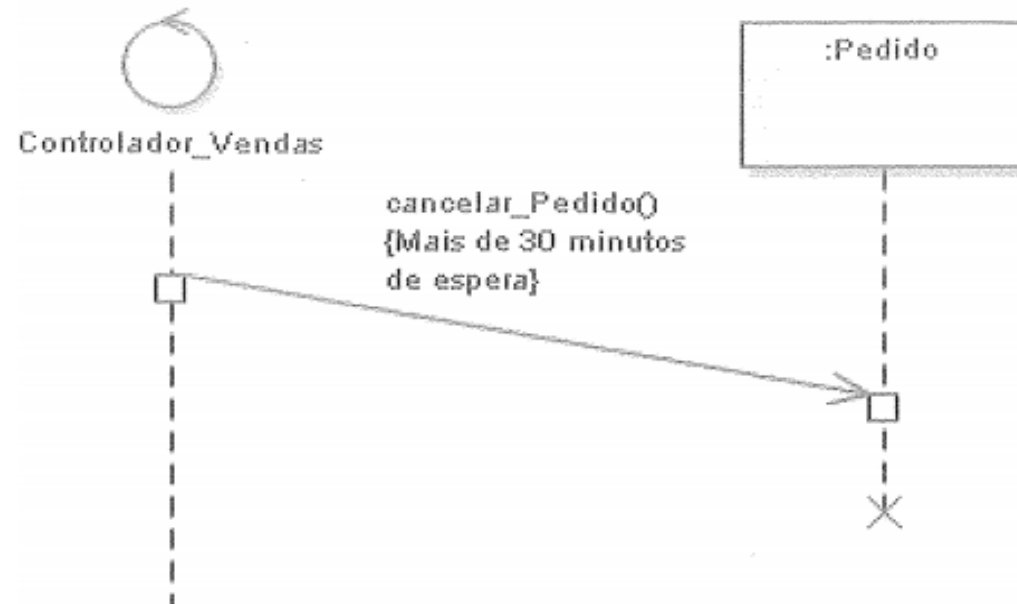
# Diagrama de Sequência

Mensagem de retorno:



# Diagrama de Sequência

Restrições de tempo:



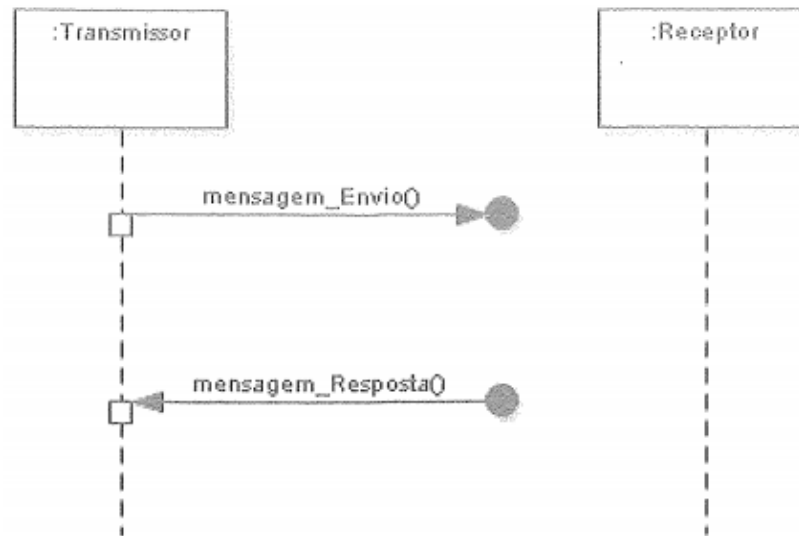
Nesse caso o pedido só será cancelado após 30 minutos



# Diagrama de Sequência

Parou aqui

## Mensagem Perdida e Mensagem Encontrada:

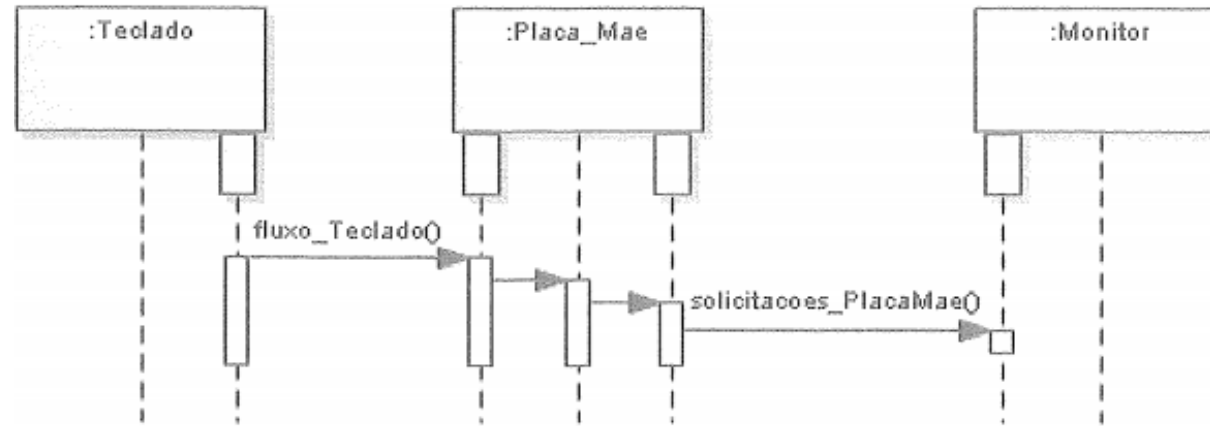


**Perdida:** não chegou ao destino ou este não está no diagrama

**Encontrada:** mensagem de remetente desconhecido ou não representado no diagrama

# Diagrama de Sequência

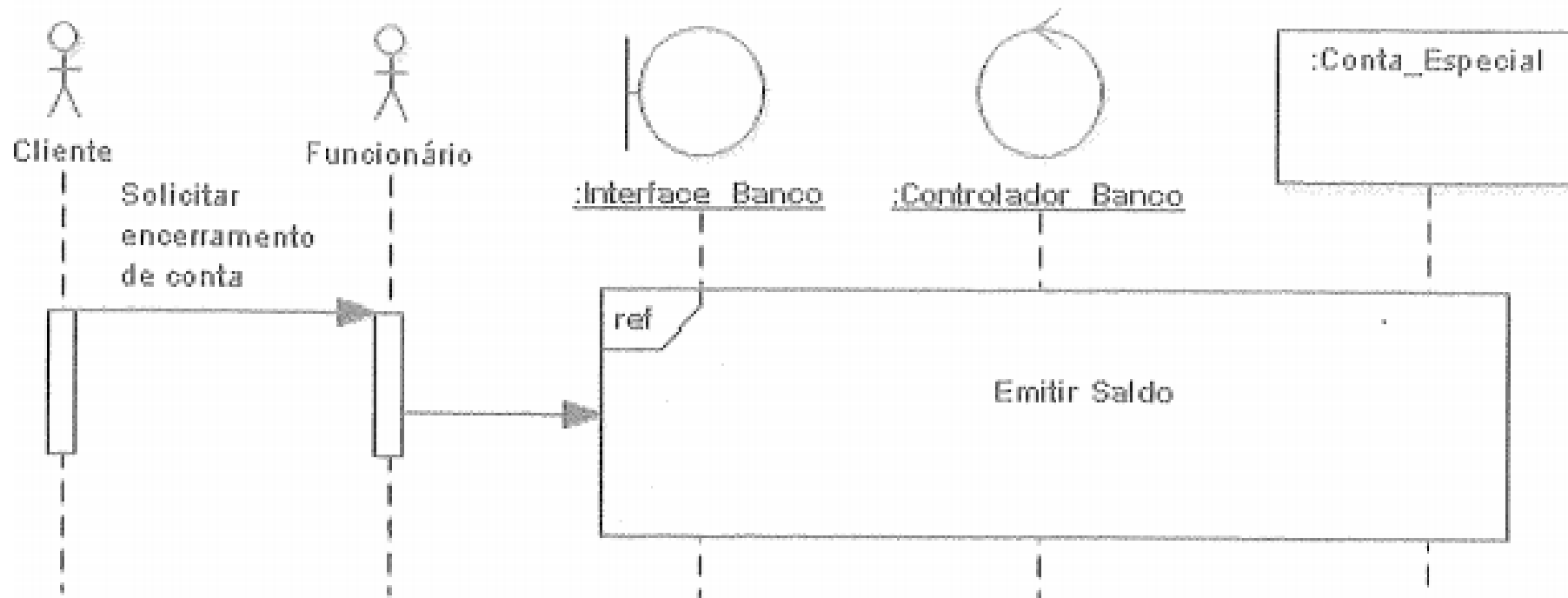
**Portas:** mesmo conceito do Diagrama de Classes



O objeto poderá ter mais de uma linha de vida, cada uma representando uma porta de comunicação

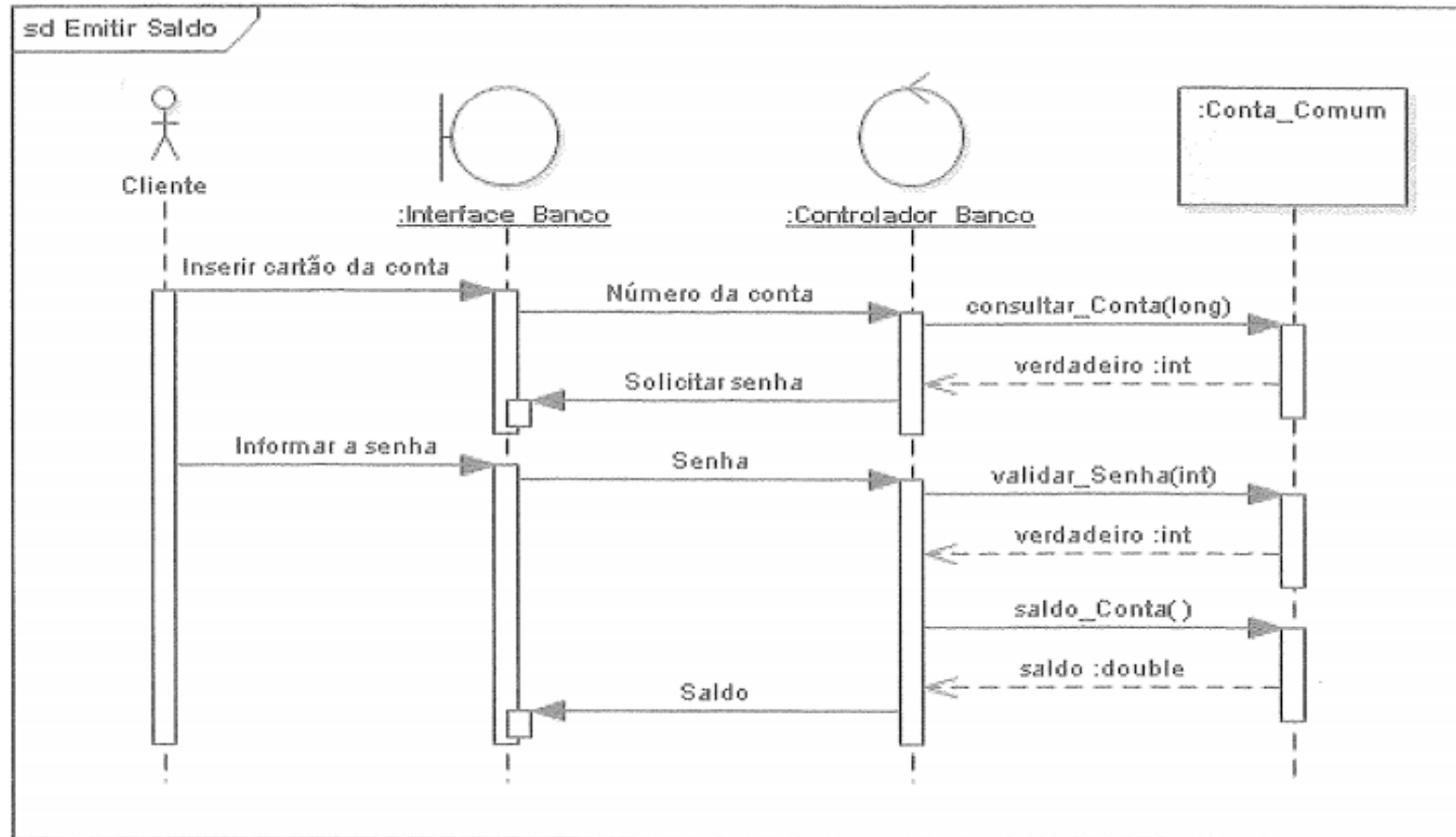
# Diagrama de Sequência

## Utilização do Fragmento de Interação

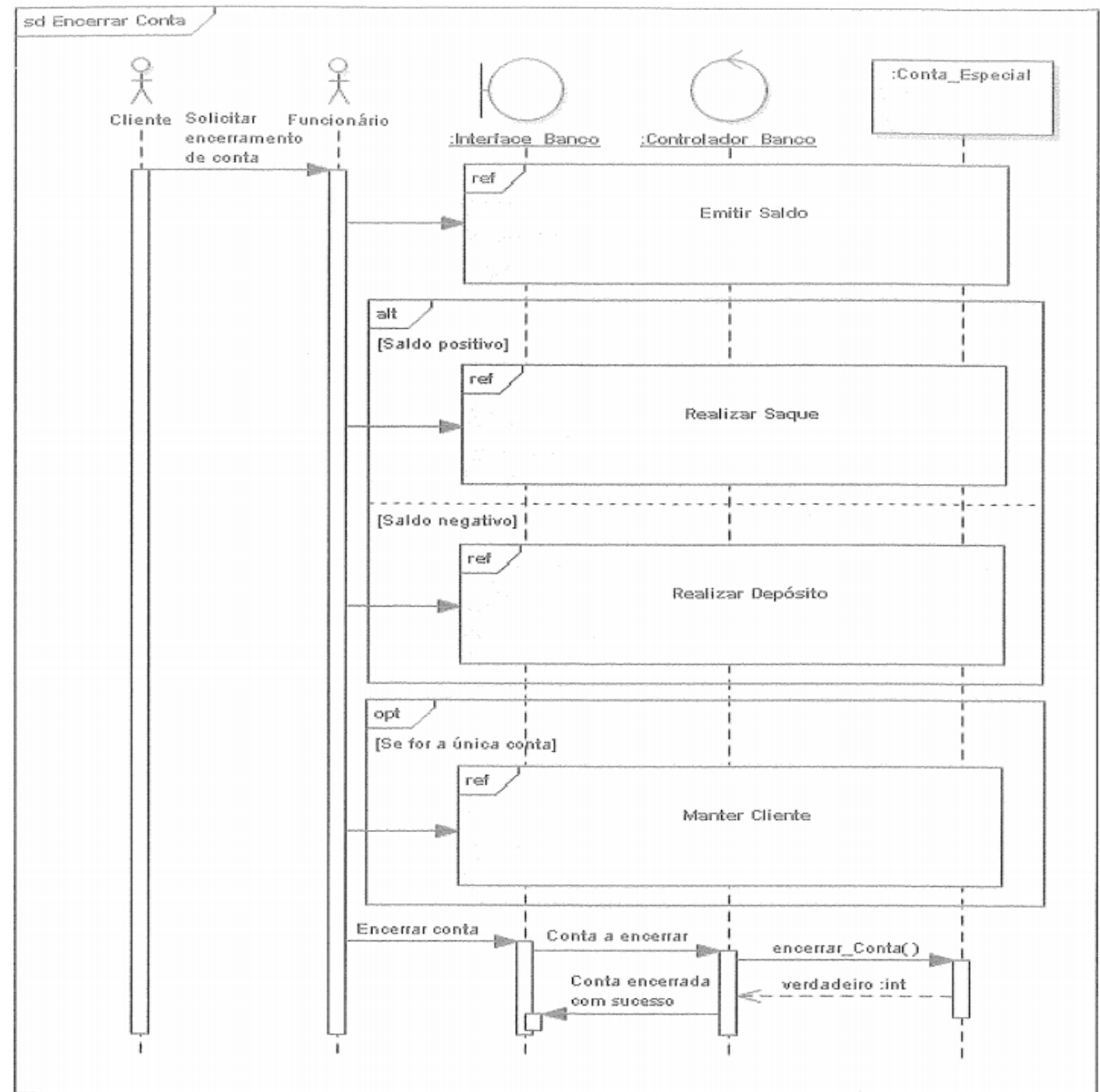


# Diagrama de Sequência

## Fragmentos de Interação



## Fragmentos Combinados



# Diagrama de Sequência

Os fragmentos em diagramas de sequência são usados para representar diferentes comportamentos e condições que afetam a interação entre objetos. Aqui estão exemplos de cada tipo de fragmento:

**REF (Reference):** Representa uma chamada a outro diagrama de sequência.

- Exemplo: Um fragmento "REF" pode indicar que a sequência de mensagens é detalhada em outro diagrama de sequência.

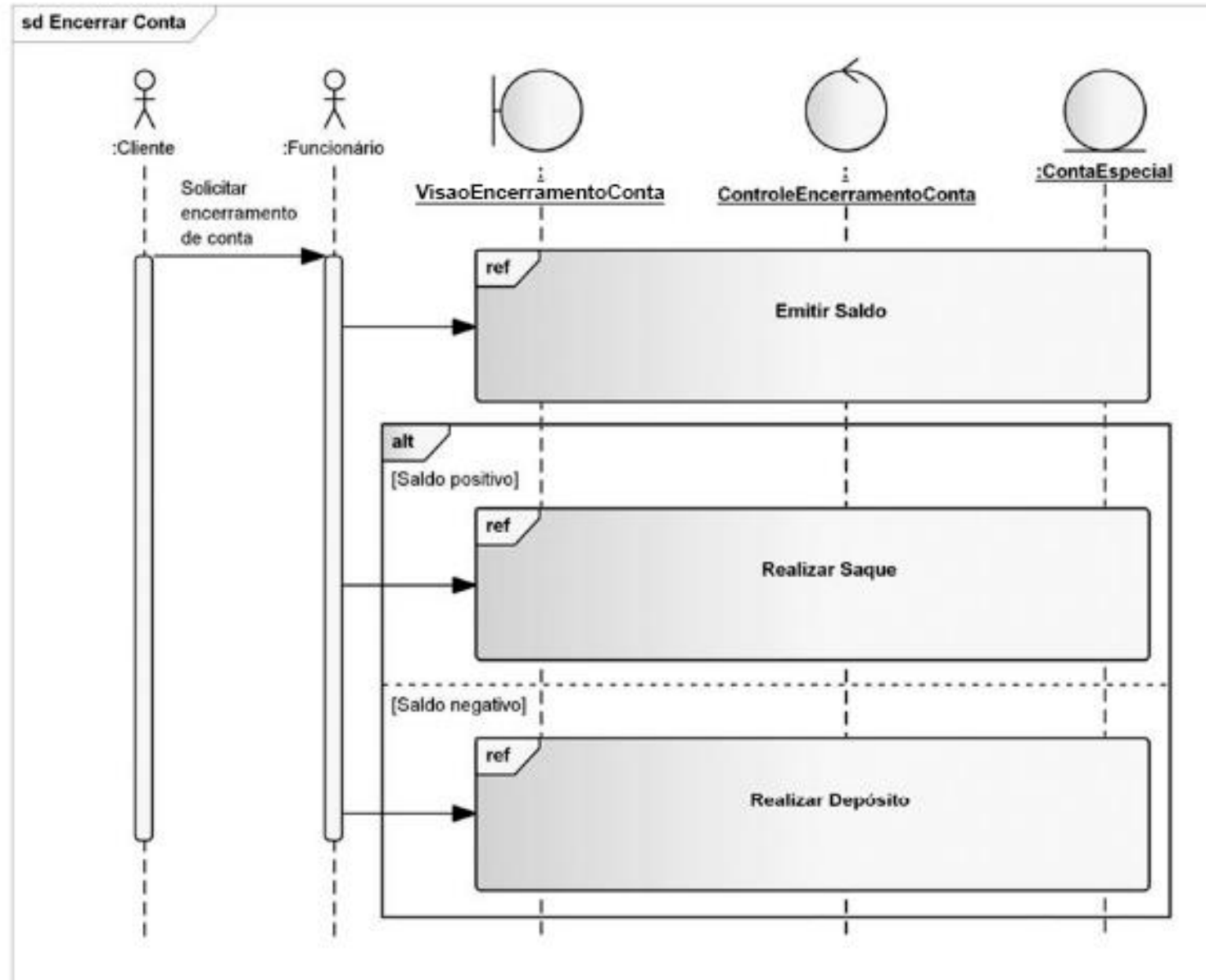
**OPT (Optional):** Executa um conjunto de mensagens condicionalmente.

- Exemplo: "Se o usuário estiver autenticado, exiba o menu." As mensagens relacionadas à exibição do menu estariam dentro de um bloco "OPT".

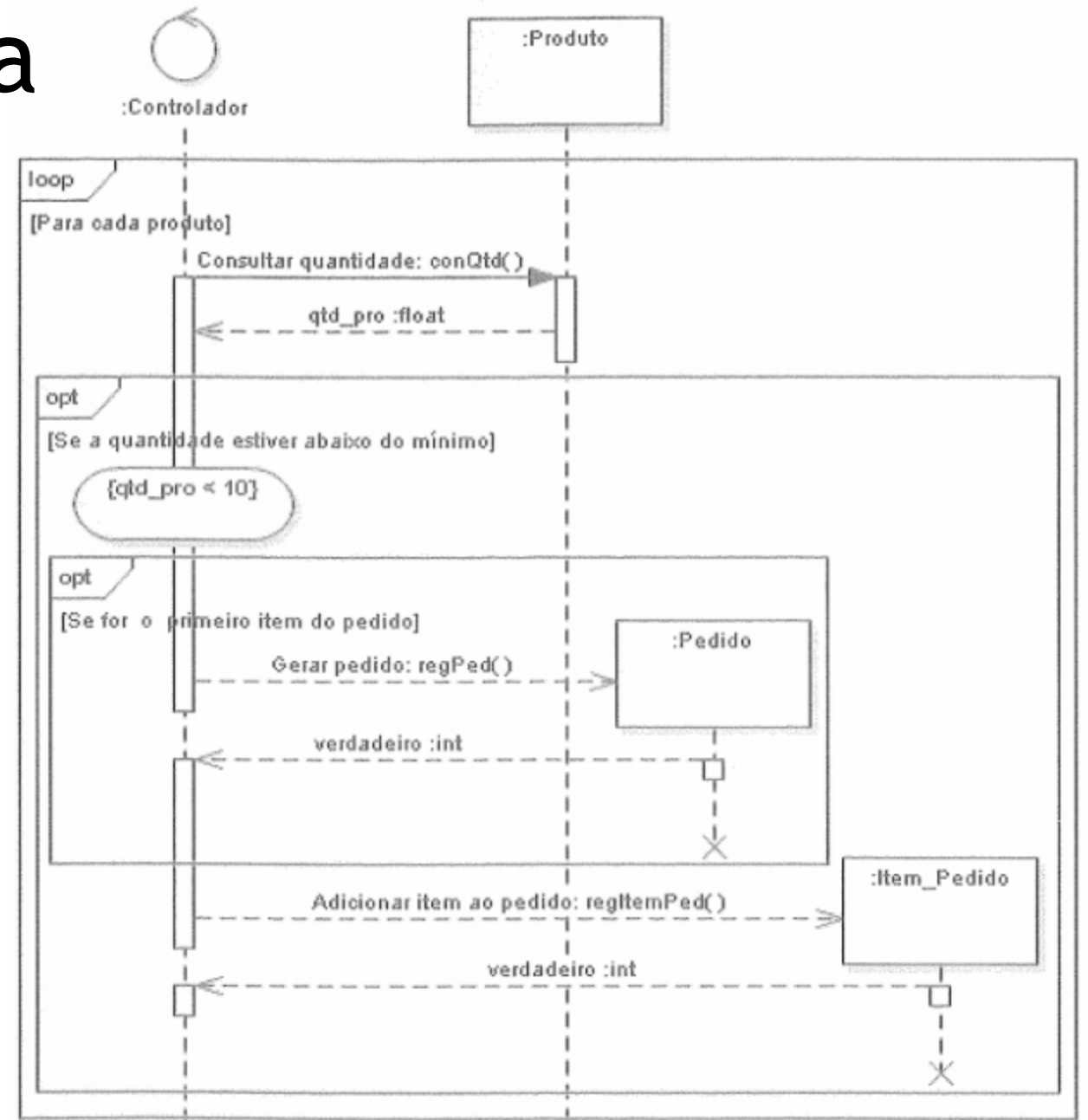
**ALT (Alternative):** Representa múltiplos fluxos alternativos, onde apenas um será executado.

- Exemplo: "Se o pagamento for aprovado, confirme a compra; caso contrário, exiba erro." O fragmento "ALT" conteria essas duas condições.

# Diagrama de Sequência



# Diagrama de Sequência

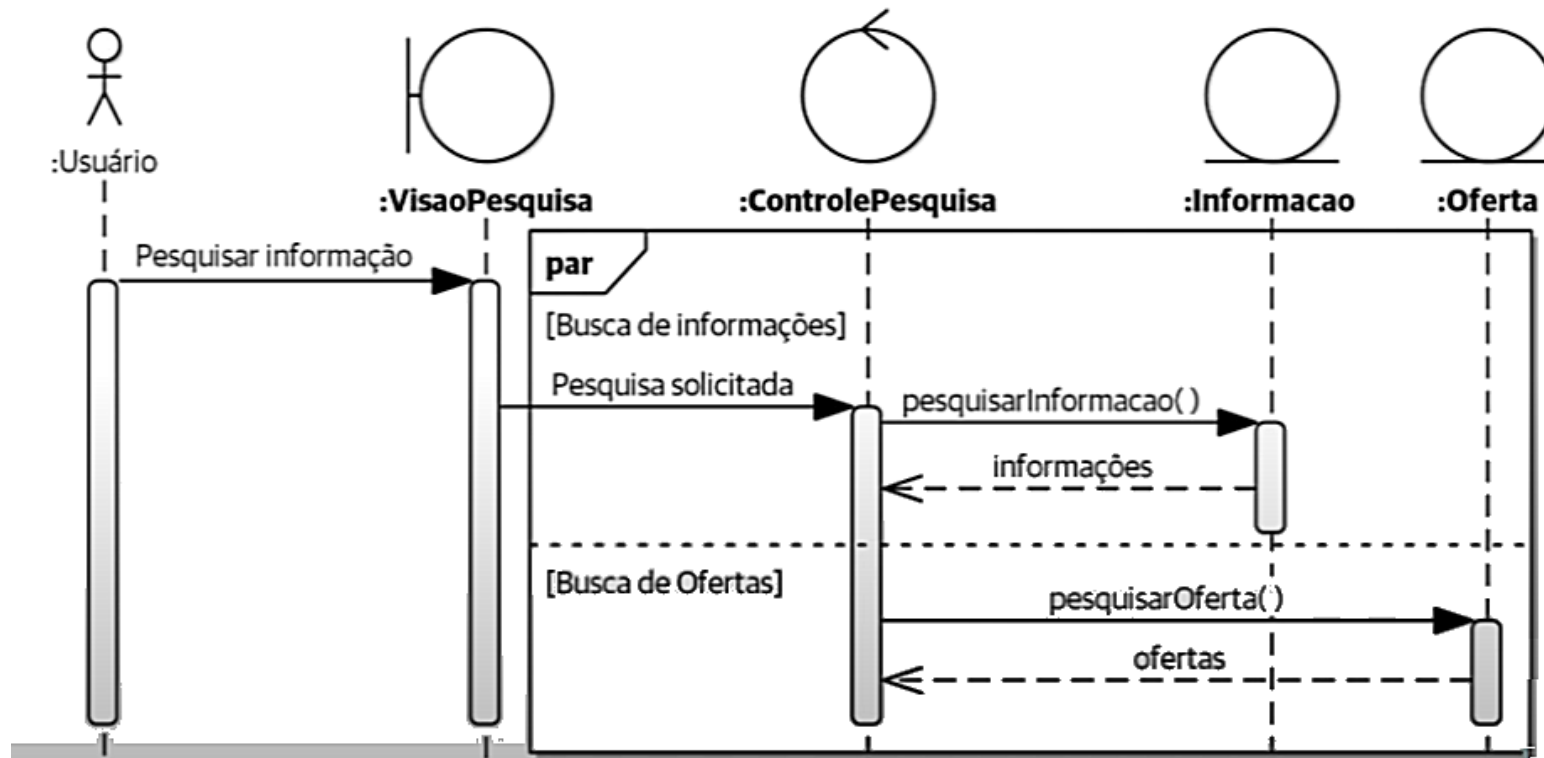




# Diagrama de Sequência

**PAR (Parallel):** Define fluxos que podem ocorrer em paralelo.

- Exemplo: "Enviar e-mail de confirmação e atualizar o estoque podem ocorrer simultaneamente." As mensagens de ambos os processos estariam em um fragmento "PAR".



# Diagrama de Sequência

**CRITICAL:** Denota uma região crítica onde apenas uma execução pode ocorrer por vez.

- Exemplo: "Atualizar o saldo do cliente deve ser feito em uma região crítica para evitar concorrência."

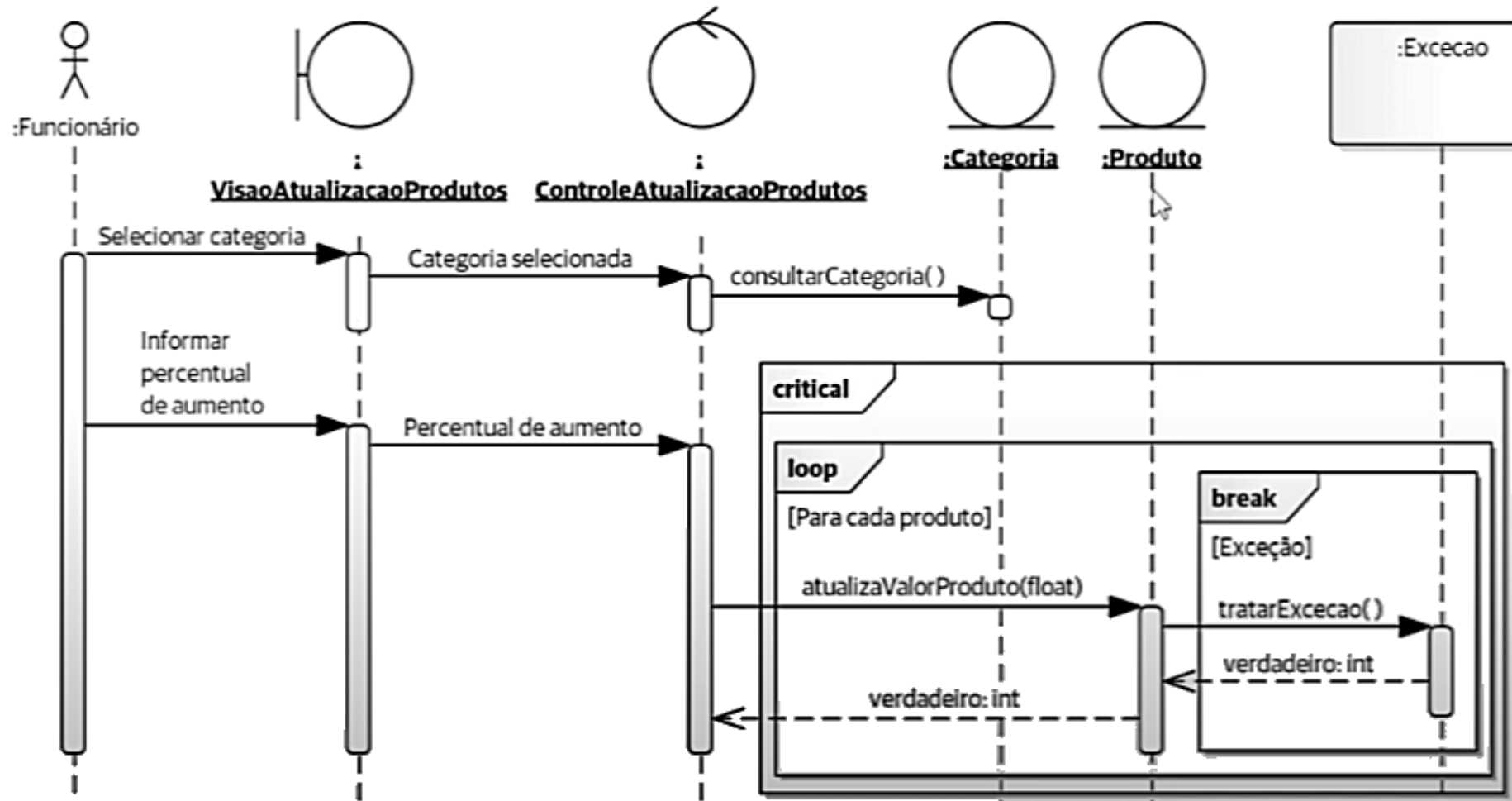
**LOOP:** Representa a repetição de um conjunto de mensagens por um número definido de vezes ou enquanto uma condição for verdadeira.

- Exemplo: "Repetir a consulta de status do pedido a cada 10 segundos até ser confirmado."

**BREAK:** Interrompe a execução da sequência quando uma determinada condição é atendida.

- Exemplo: "Se o cartão for inválido, interrompa o processo de pagamento." O fragmento "BREAK" encerraria o fluxo.

# Diagrama de Sequência



# Diagrama de Sequência

**ASSERTION:** Representa uma condição que deve ser verdadeira para que o sistema continue funcionando corretamente.

- Exemplo: "O sistema deve garantir que o estoque nunca seja negativo." Esse fragmento poderia incluir uma verificação de integridade.

**IGNORE:** Indica que certas mensagens podem ser ignoradas no contexto.

- Exemplo: "Ignorar mensagens relacionadas a logs enquanto processa o pedido."

**NEG (Negative):** Especifica um comportamento que não deve ocorrer no sistema.

Exemplo: "Um pagamento duplicado não deve ser permitido." O fragmento "NEG" representaria esse fluxo inválido.

# Diagrama de Sequência

**CONSIDER:** Especifica as mensagens que devem ser consideradas no contexto, ignorando todas as outras.

- Exemplo: "Considerar apenas mensagens de pagamento no diagrama."

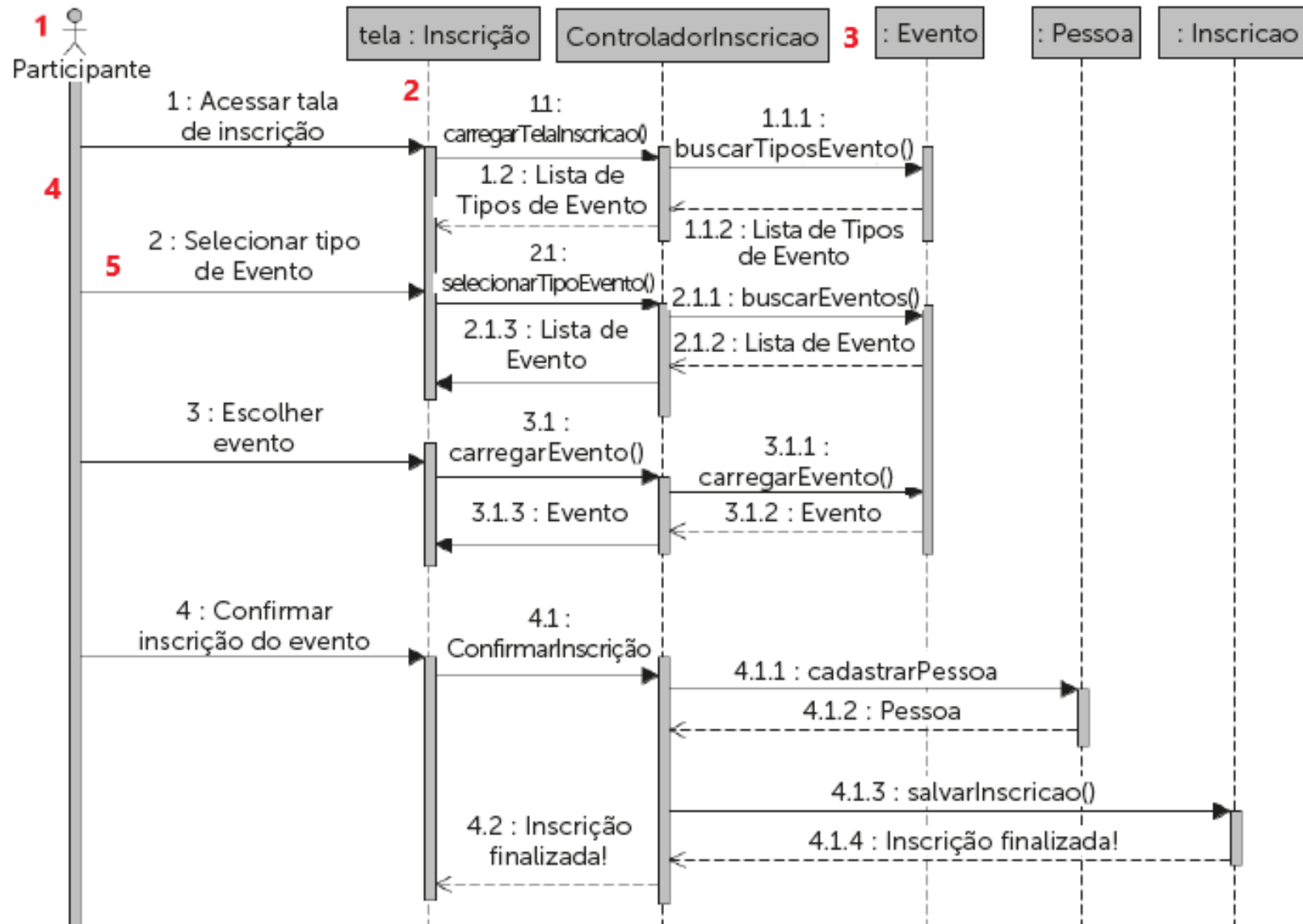
**SEQ (Sequential):** Indica que as mensagens devem ser processadas sequencialmente, uma após a outra.

- Exemplo: "Processar primeiro o pagamento, depois gerar a fatura e só então liberar o produto."

**STRICT:** Exige que as mensagens sejam executadas estritamente na ordem definida.

- Exemplo: "A verificação de segurança e a autenticação devem ocorrer exatamente nessa ordem."

# Diagrama de Sequência



# Exercício

Você foi contratado para modelar uma funcionalidade de um sistema de gestão de pedidos de uma loja online. O objetivo deste exercício é desenhar o diagrama de sequência que represente o processo de compra de um cliente no site da loja.

## Cenário:

- 1) O cliente acessa o site da loja e escolhe um ou mais produtos, adicionando-os ao carrinho.
- 2) Após revisar os itens no carrinho, o cliente confirma a compra.
- 3) O sistema envia uma solicitação para o serviço de pagamento para processar o pagamento.
- 4) O serviço de pagamento verifica os detalhes do cartão e, em caso de sucesso, confirma o pagamento ao sistema.
- 5) O sistema gera um pedido e envia a confirmação ao cliente.
- 6) O sistema também notifica o estoque para atualizar a quantidade dos produtos comprados.

## Tarefas:

- 1) Identifique os principais atores e objetos envolvidos no processo descrito.
- 2) Elabore um diagrama de sequência que represente toda a interação entre o cliente, o sistema, o serviço de pagamento e o estoque.
- 3) As interações devem incluir as mensagens de solicitação e resposta entre os atores e os objetos.