

Tópico 6:

Diagrama de Sequência – Parte 2

Luiz Antônio M. Pereira

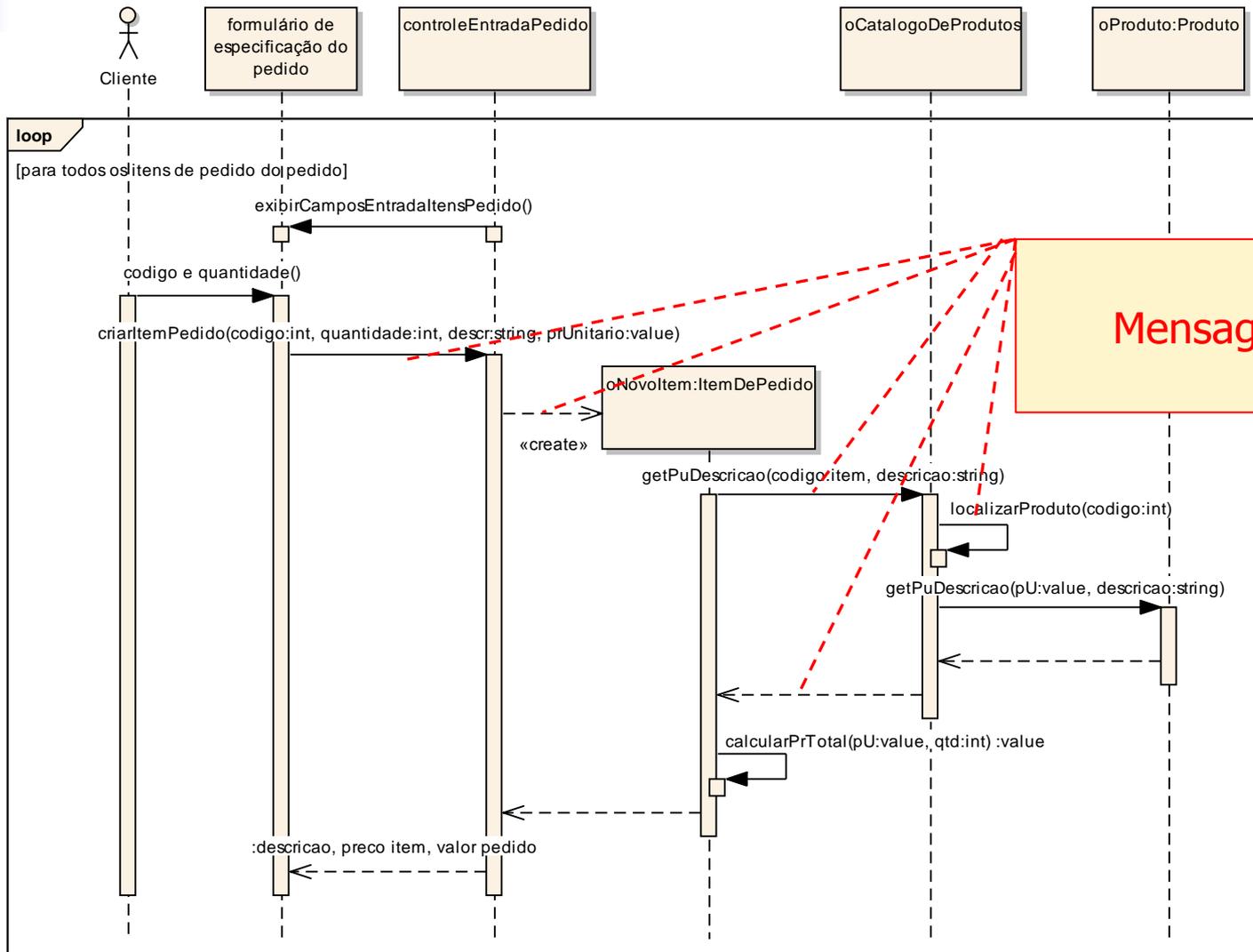
lpereira@uninet.com.br

lpereira@luizantoniopereira.com.br

Mensagens

- Vimos que um objeto interage com outro objeto trocando mensagens:
 - Que delegam a ele a execução de alguma operação de seu rol de operações públicas;
 - Que informam que a operação terminou de ser executada e que, portanto, o controle está sendo passado de volta;
 - De criação e destruição de objetos.

Mensagens



Mensagens



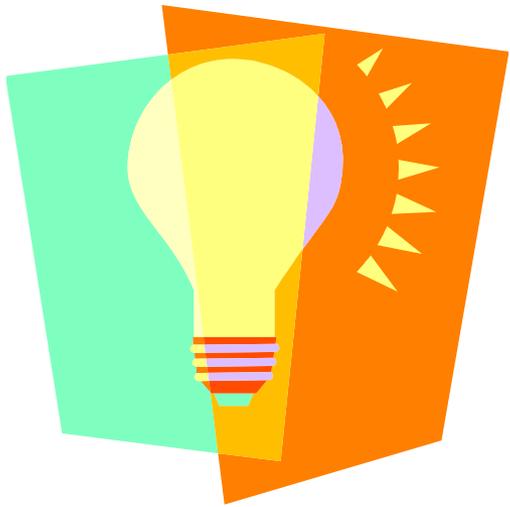
Mensagens

- Mensagens são representadas por setas
 - Que partem da linha de vida do objeto que envia para a linha de vida do objeto que recebe e trata a mensagem;
 - As setas podem ser rotuladas com o nome do estímulo e com os valores ou expressões de seus argumentos ou retornos;
 - O objeto que recebe pode ser o mesmo que envia → *auto-chamada* ou *auto-delegação*, resultando numa invocação local de uma operação.



Mensagens

- O envio da mensagem só é possível se o objeto possuir a referência do objeto para o qual deseja enviá-la.



Por isso é importante que os endereços sejam bem guardados.

Mensagens

- Mensagens na UML 2.X podem corresponder a quatro tipos de ações:
 - Chamada;
 - Retorno;
 - Criação;
 - Destruição.



Mensagens

- Chamada:

- Invoca uma operação de um objeto (alvo).

Notação: \longrightarrow

- Retorno:

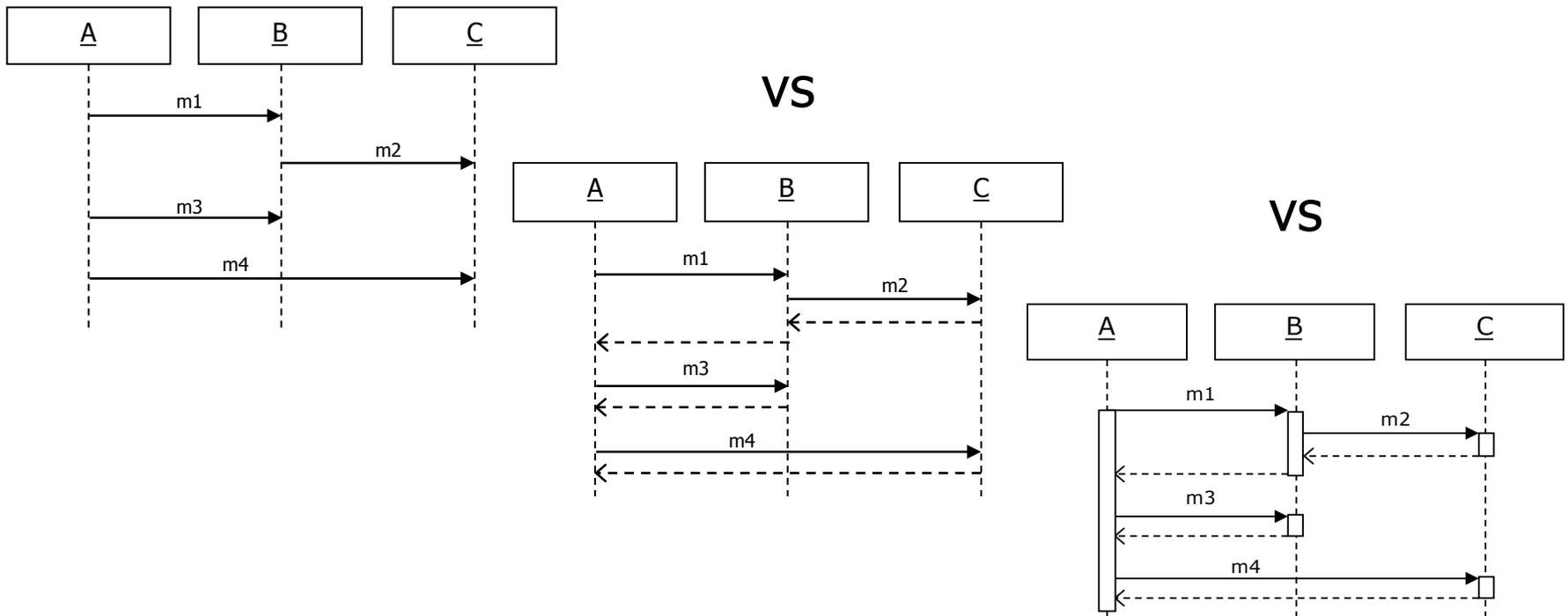
- Retorna um valor e o controle para o chamador.

Notação: \dashrightarrow



Mensagens

- Retornos são opcionalmente representados, segundo a UML. Em determinadas situações não há perda de expressividade.





Mensagens

- Chamada de Criação

- Cria um objeto.

Notações:



- A caixa do identificador do objeto fica alinhada com a mensagem de criação.

- Chamada de Destruição

- Destrói um objeto. Um objeto pode cometer suicídio, se auto-destruindo.

Notações:



- A "morte" de um objeto é denotada por um "X"



Mensagens

Mensagens podem ser ainda:

- Síncronas: objeto que chama “espera” pelo término do processamento da mensagem pelo objeto chamado.
Notação: vide exemplos anteriores.
- Assíncronas: quando não síncronas.

Notação: \longrightarrow

Mensagens

O rótulo de uma mensagem também pode conter*:

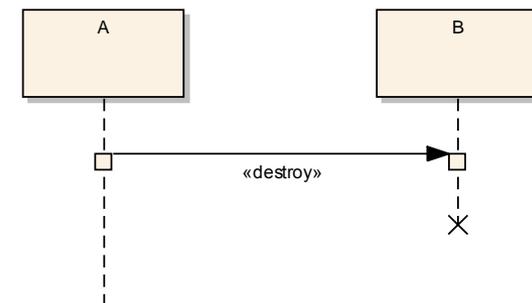
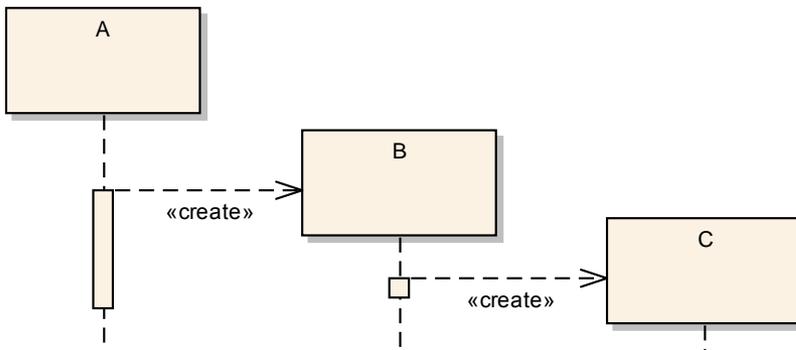
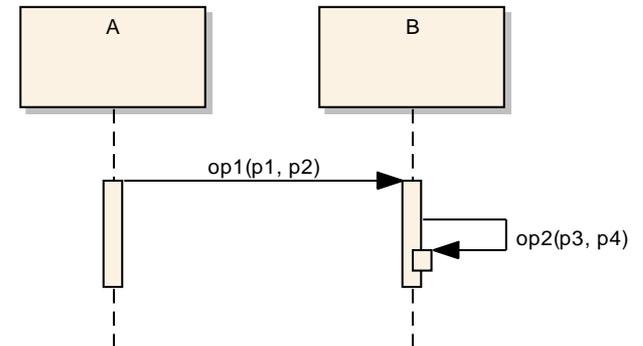
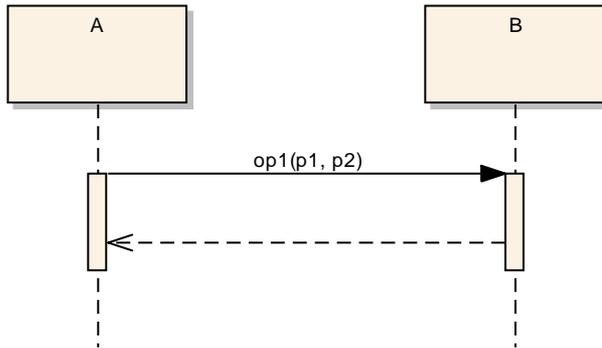
- A expressão de condição de envio: representada entre “[” e “]”, que indica a condição para a mensagem ser enviada.
- O símbolo de *iteração*: “*” no início da mensagem indica que a mesma é enviada repetidas vezes.

(*) Notação anterior à UML 2.0, porém ainda usada.

Mensagens

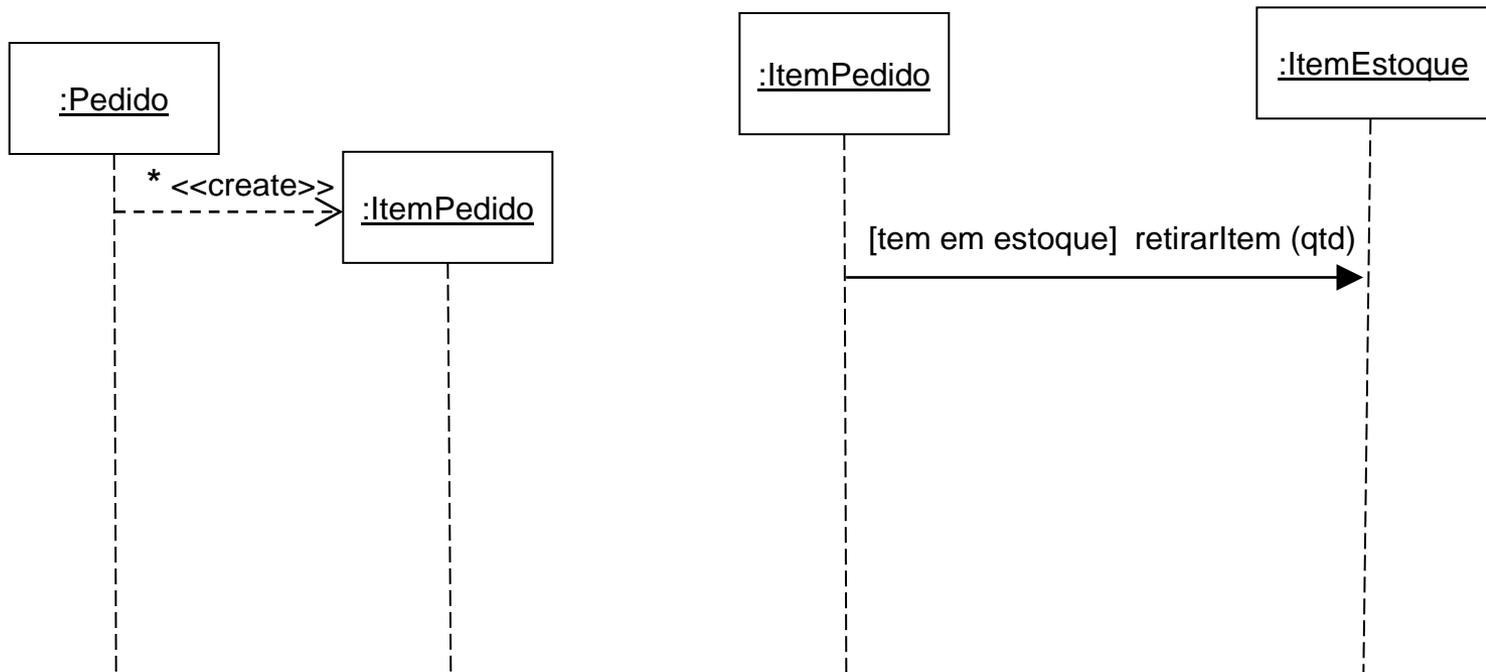
- Chamada, retorno, criação e destruição de objetos

Exemplos:



Mensagens

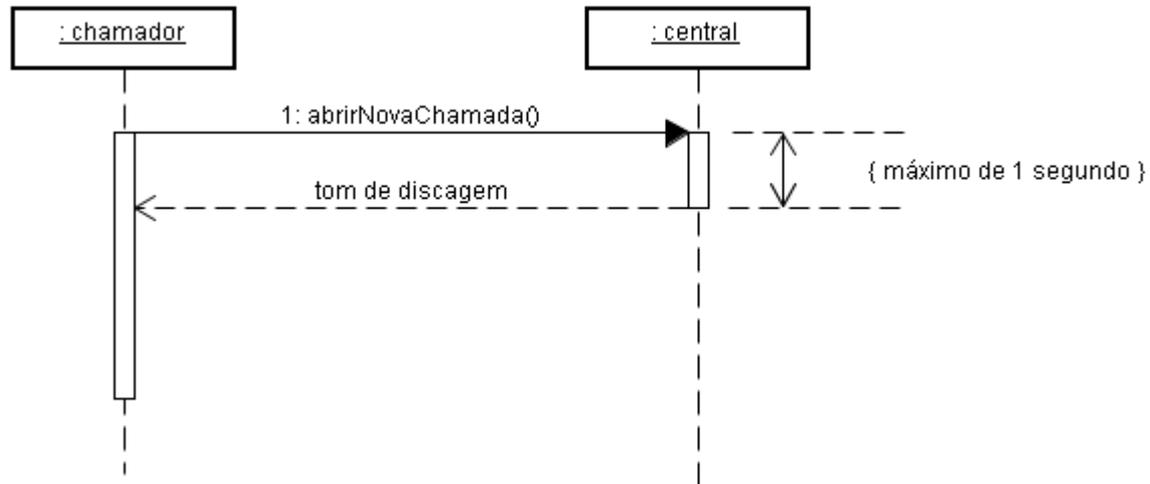
- Iterações e condições
Exemplos:



Mensagens

- Restrições temporais podem ser representadas.

Exemplo:



Mensagens

- Parâmetros da chamadas e tipos de retorno:
 - Precisam ser representados, ou seja, a assinatura completa da operação precisa constar do DS;
 - Tornam o DS visualmente ainda mais complexo;
 - ❖ Uma longa lista de parâmetros *pode* significar que o objeto do qual estamos solicitando a colaboração não seja o mais adequado.

Exercício

- 5.3 - Secos e Molhados do Seu Manoel

Interfaces

- As interfaces usualmente proveem meios de comunicação entre ambientes distintos
 - entre dois (sub) sistemas,
 - entre duas tecnologias diferentes,
 - entre os usuários e os sistemas.

Interfaces

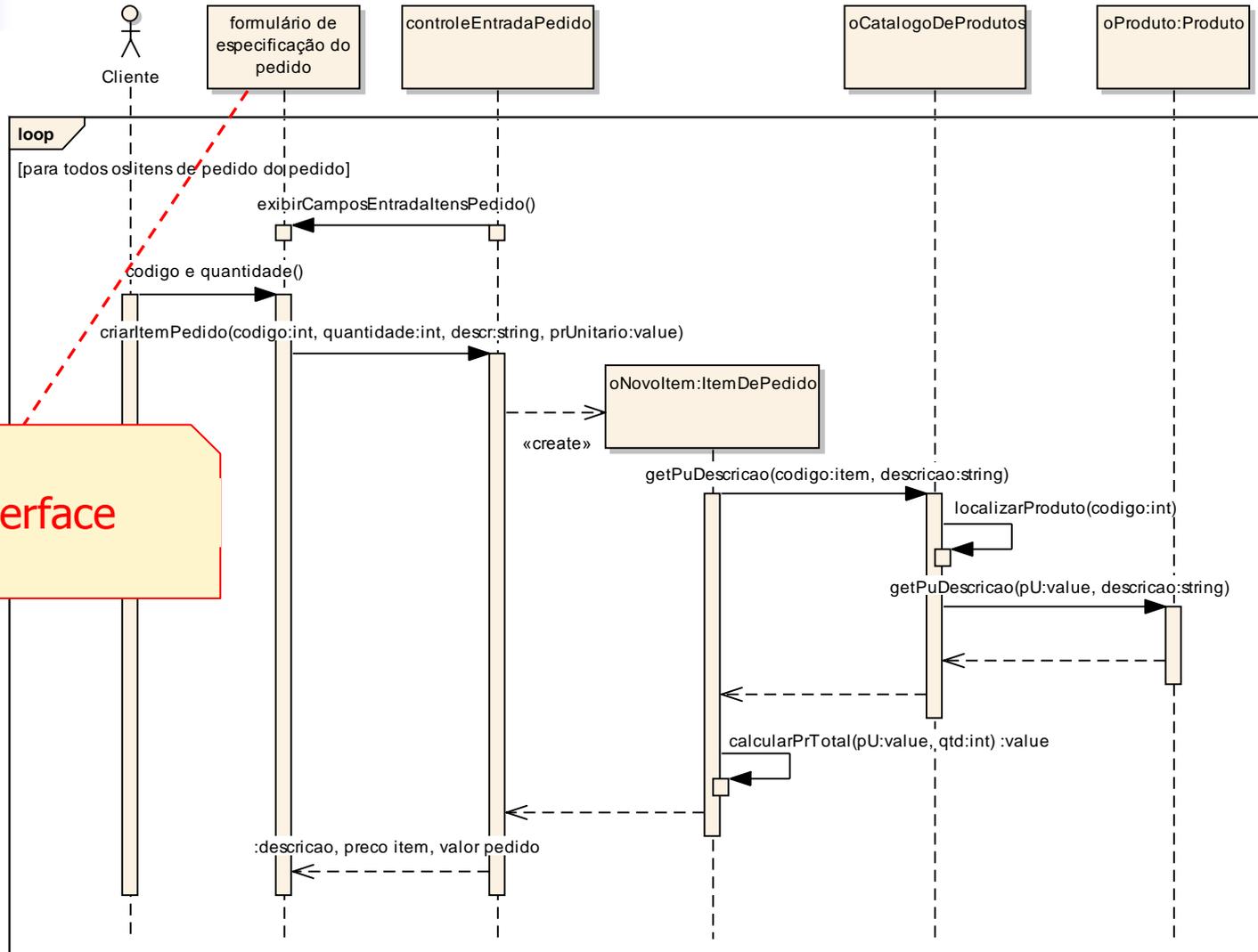
- Atores não interagem diretamente com objetos do sistema. Os *formulários* cumprem essa missão. Eles são:
 - Telas das aplicações web (os navegadores ou browsers), ou
 - Telas das aplicações desktop (formulários Visual Basic, Delphi etc.).

Interfaces

- Objetos instanciados dessas classes armazenam os dados fornecidos pelos usuários e os repassam aos objetos do sistema.
- Esses objetos também recebem os eventos gerados pelos usuários e os repassam aos objetos do sistema em um DS.

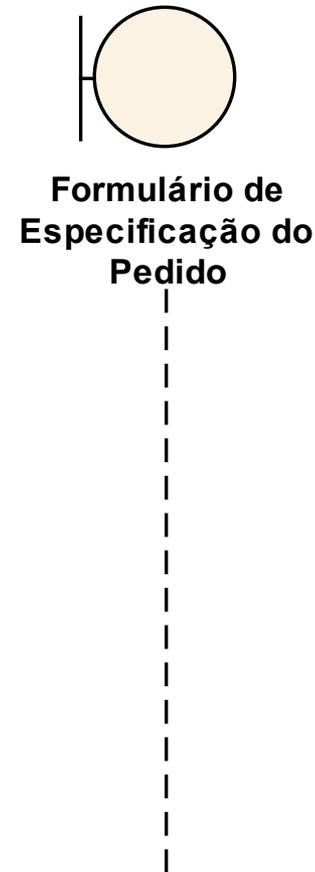


Interfaces



Interfaces

- Também podem ser representadas na forma:



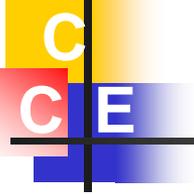
Interfaces

- A boa prática recomenda: formulários não devem ter muita “inteligência”
 - Não devem implementar regras de negócio;
 - Não devem coordenar interações com os objetos do sistema;
 - Não devem verificar consistências de dados;
 - Podem implementar críticas simples (verificação de dígitos verificadores e de preenchimento de campos obrigatórios, etc.)



Controladores

- São objetos que têm a responsabilidade de controlar o fluxo de mensagens e/ou de zelar pela obediência às regras de negócio e pelas demais questões de consistência;
- Alguns projetistas retiram quase todas as responsabilidades das classes conceituais, que passam a ser atribuídas às classes controladoras.

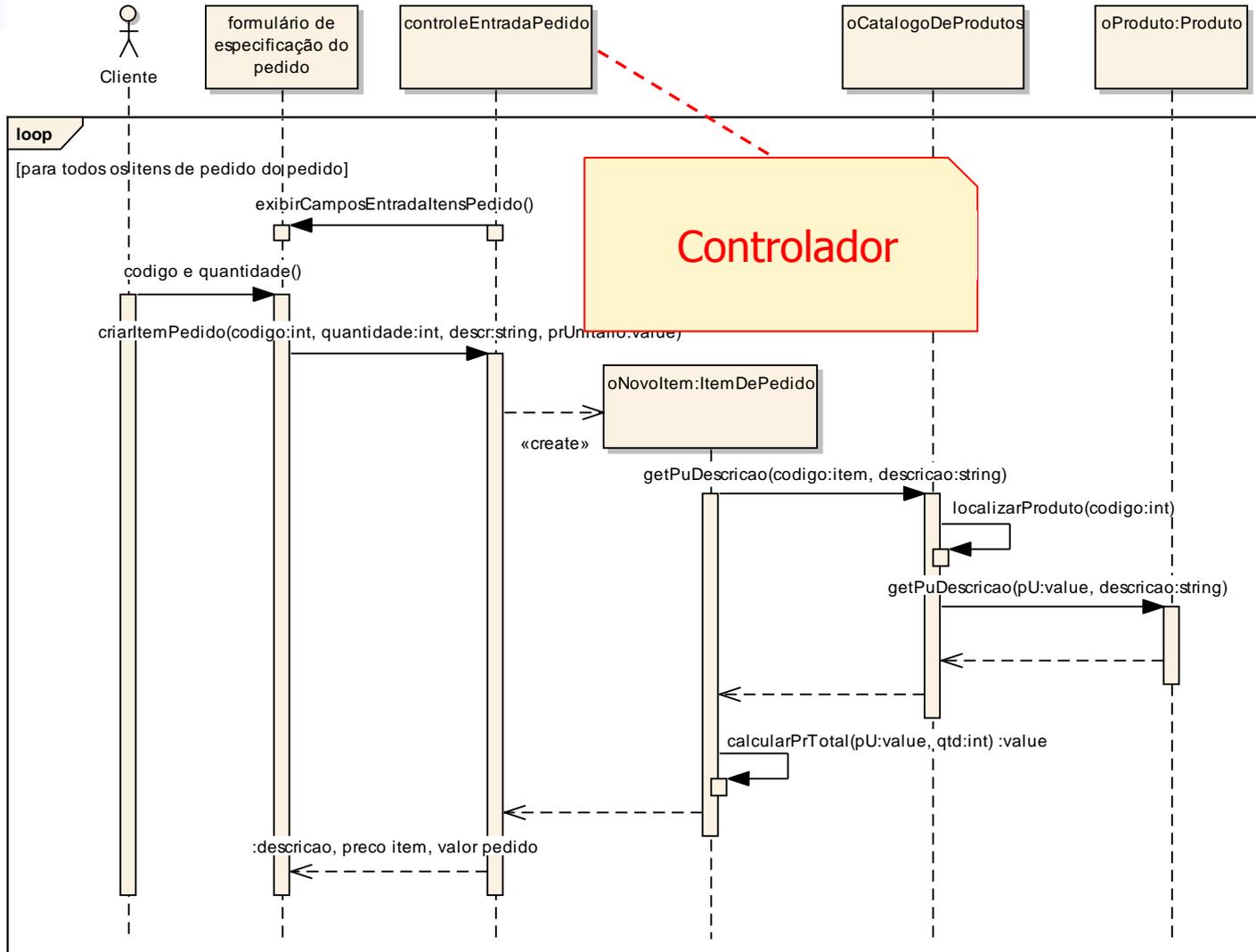


Controladores

- Usualmente é concebida uma classe controladora por caso de uso;
- Podem ser concebidas mais classes no caso de UCs complexos.

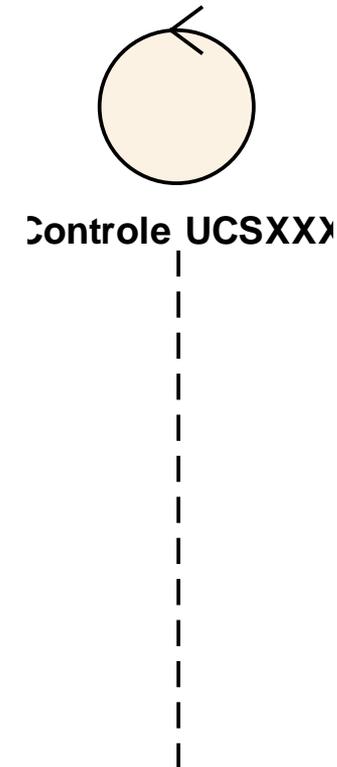


Controladores



Controladores

- Também podem ser representados na forma:





Interfaces e Controladores

- Suas classes devem ser integradas ao modelo de classes;
- A partir desse momento, os diagramas de classes deixam de ser *diagramas de nível conceitual*.

Exercício

- 5.4 – Caixa Eletrônico do Banco Baú
- 5.5 – Cadastrando Pedidos

Quadros de Interação

- DSs são complexos visualmente, especialmente quando modelamos colaborações contendo iterações, condicionalidades (múltiplos cenários) e processamento paralelo, utilizando os elementos de notação das versões anteriores à 2.0.

Quadros de Interação

- A UML 2.0 incorporou o conceito de quadros de interação (*frames*), adicionando recursos importantes que facilitam a especificação de colaborações com essas características.

Quadros de Interação

- Quadros de interação definem uma ou mais regiões onde representamos
 - Iterações,
 - Trechos concorrentes,
 - Trechos opcionais,
 - Trechos alternativos de colaborações,
 - Trechos de interação comuns, fatorados de duas ou mais colaborações,
 - Etc.

Quadros de Interação

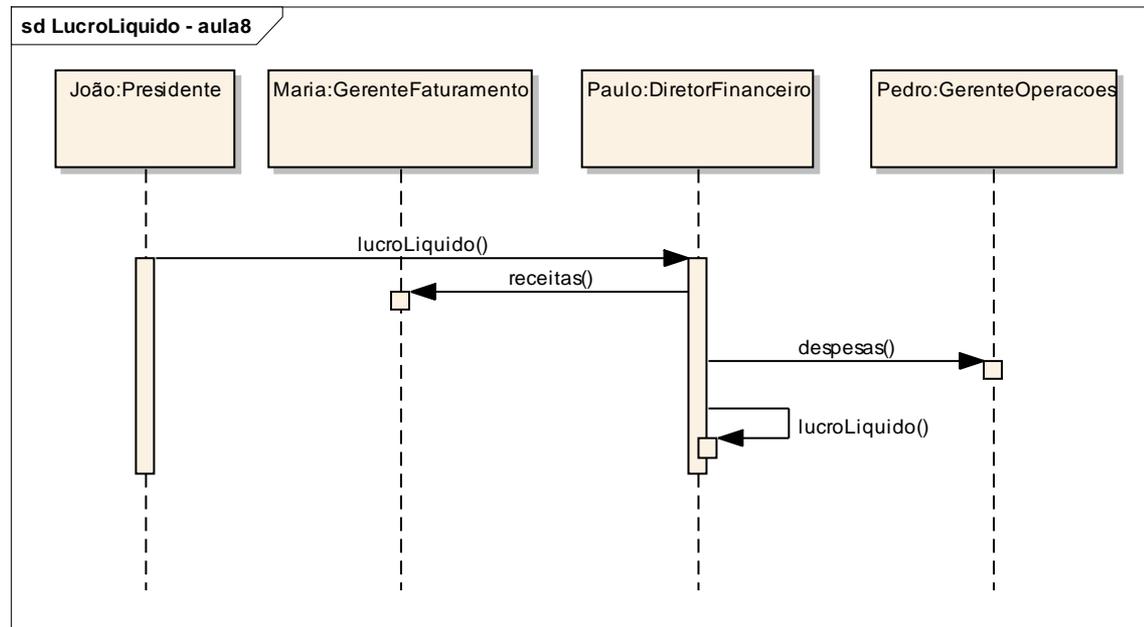
- Quadros de interação tornaram os DS mais expressivos, mais completos (mais de um cenário representado no mesmo diagrama), ao mesmo tempo que mais simples visualmente.

Quadros de Interação

- Os quadros são rotulados com nomes de operadores;
- Rótulos dos operadores são colocados no canto superior esquerdo;
- As regiões que compõem os quadros, quando existe mais de uma, chamam-se fragmentos;
- Fragmentos são separados por linhas tracejadas.

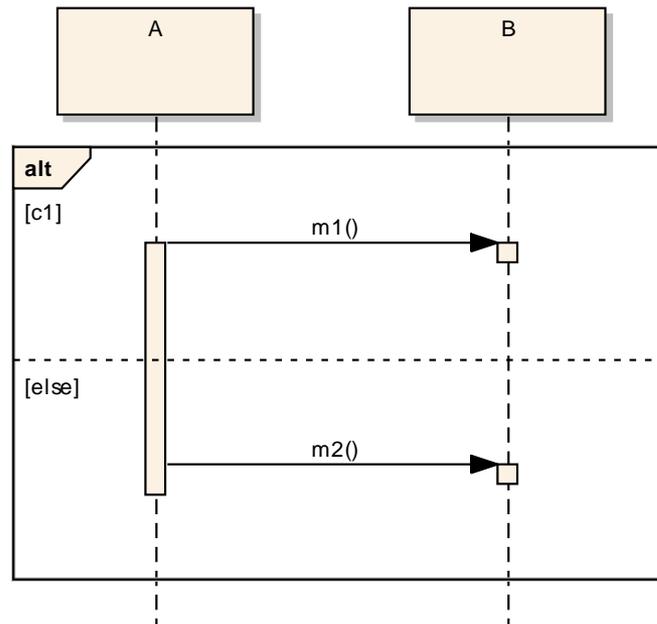
Quadros de Interação

- Operadores mais comuns:
 - **sd**: ("*sequence diagram*"), opcionalmente rotula um quadro que contém totalmente um diagrama de sequência.



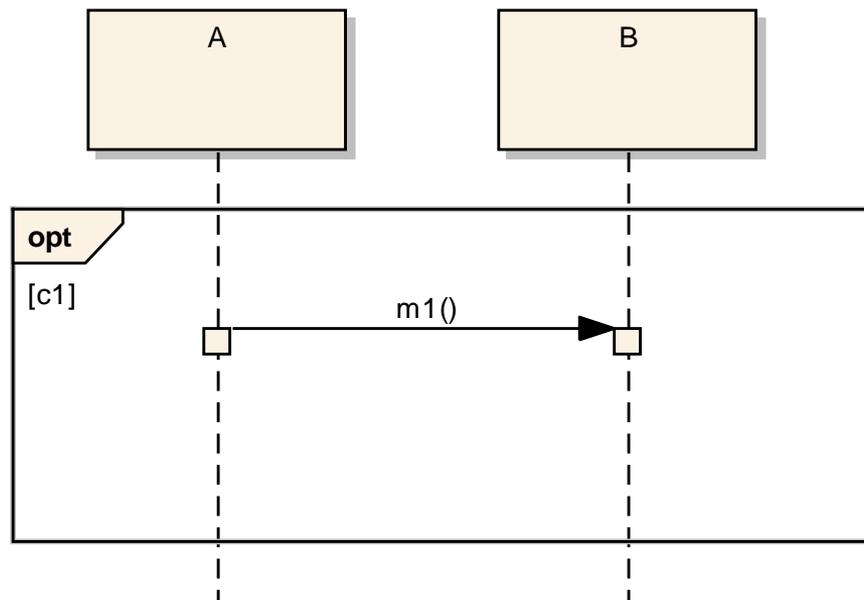
Quadros de Interação

- Operadores mais comuns:
 - **alt**: especifica múltiplos fragmentos alternativos em um quadro. A condição em que um fragmento é executado é colocada no topo do fragmento entre colchetes.



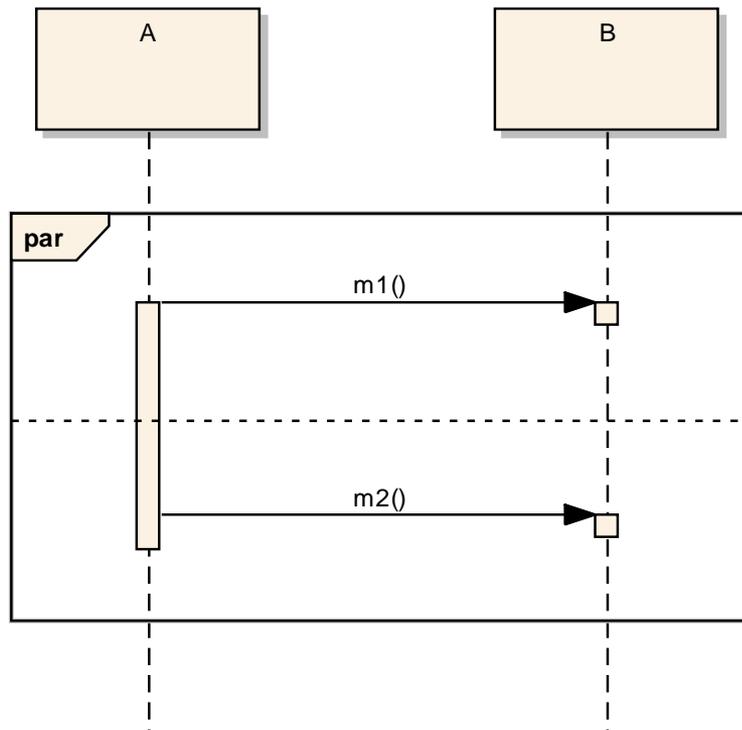
Quadros de Interação

- Operadores mais comuns:
 - **opt**: especifica um quadro executado opcionalmente. A condição em que o fragmento é executado é colocada no topo do quadro, entre colchetes.



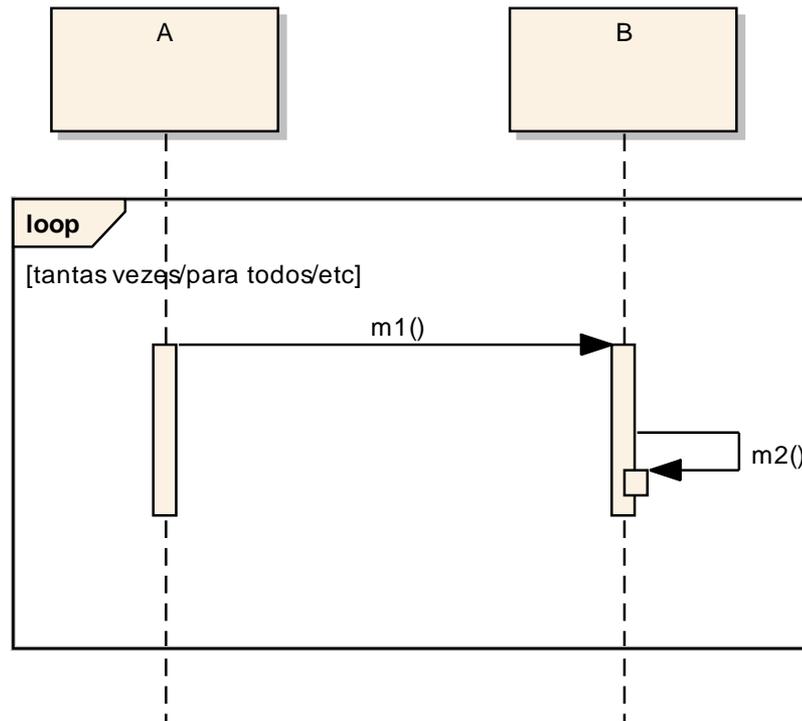
Quadros de Interação

- Operadores mais comuns:
 - **par**: especifica múltiplos fragmentos executados concorrentemente.



Quadros de Interação

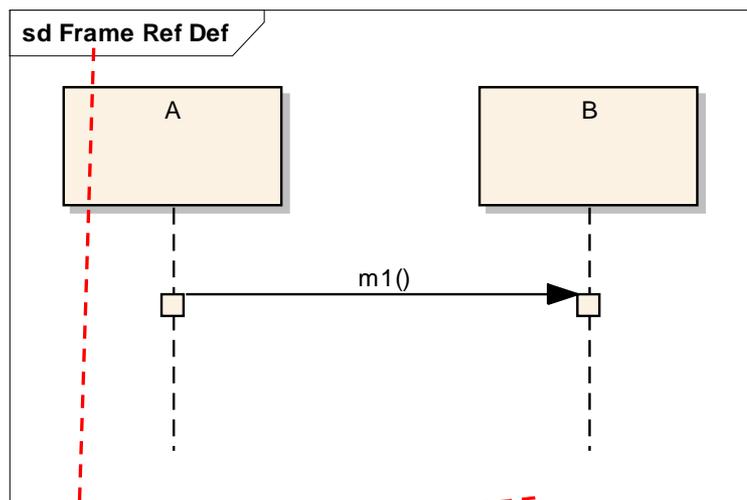
- Operadores mais comuns:
 - **loop**: especifica um quadro (um único fragmento) executado repetidas vezes.



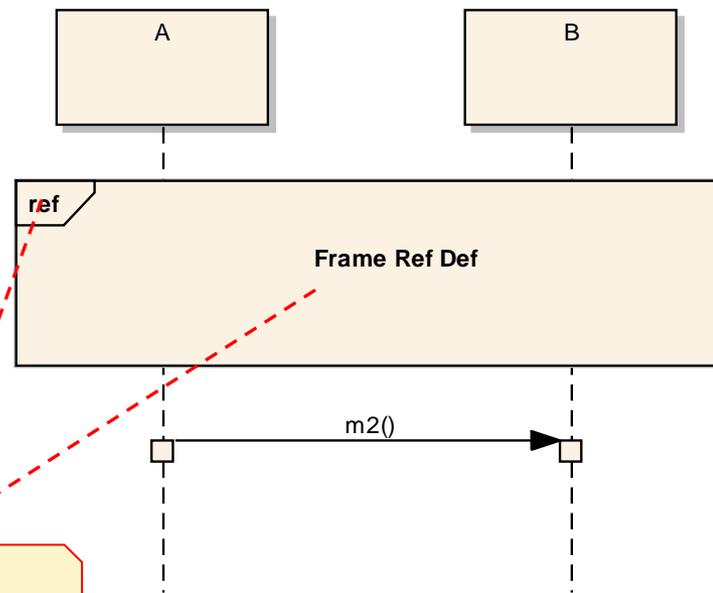


Quadros de Interação

- Operadores mais comuns:
 - **ref**: especifica a chamada a uma colaboração definida em outro lugar ou em outro diagrama.



Define



Chama

Exercícios

- 5.6 – Cadastrando Pedidos Revisitado
- Aglomeratto Classe A