Análise e Programação Orientada a Objetos

Sistemas de Informação

Unidade II – Parte IV

Modelagem de Interações

Projetando o Sistema

Prof. Marciel de Liz Santos

Introdução

Somente após a construção de diagramas de interação para os cenários de um caso de uso, pode-se ter certeza de que todas as responsabilidades que os objetos devem cumprir foram identificadas.

Ivar Jacobson, 1995

Introdução

- Os modelos vistos até agora (MCU e classes) fornecem um entendimento do problema correspondente ao sistema de software a ser desenvolvido.
- MCU – descreve quais os requisitos funcionais do sistema e quais são as entidades (atores) que interagem com o sistema.
  - Também informa quais as ações do sistema percebidas pelo ator e vice-versa, **porém**, não informa quais as operações internas que ocorrem.
- Com o MCU é possível identificar **O QUE** o sistema deve fazer e para quem, mas não **COMO** fazer.
- Desta forma, algumas questões não podem ser respondidas por esse modelo:
  - Quais são as **operações** que devem ser executadas internamente ao sistema?
  - A que **classes** estas operações pertencem?
  - Quais **objetos** participam da realização deste caso de uso?

Introdução

- O modelo de classes de domínio (análise) fornece uma visão estrutural e estática inicial do sistema.
- O modelo apresenta um esboço das classes e de suas responsabilidades.
- Porém, algumas questões não podem ser respondidas por esse modelo:
  - De que forma os objetos **colaboram** para que um determinado caso de uso seja realizado?
  - Em que **ordem** as mensagens são enviadas durante esta realização?
  - Que **informações** precisam ser enviadas em uma mensagem de um objeto a outro?
  - Será que há responsabilidades ou mesmo classes que ainda não foram identificadas?

Modelo de Interação

- Para responder às questões anteriores, o modelo de interações deve ser criado.
- Os objetivos da construção do modelo de interação são:
  - obter informações adicionais para completar e aprimorar outros modelos
    - Quais as **operações** de uma classe?
    - Quais os **objetos** participantes da realização de um caso de uso (ou cenário deste)?
    - Para cada operação, qual a sua **assinatura**?
    - Uma classe precisa de mais **atributos**?
  - fornecer aos programadores uma visão detalhada dos objetos e mensagens envolvidos **na realização dos casos de uso**.
- A interação entre objetos para dar suporte à funcionalidade de um caso de uso denomina-se **realização de um caso**.

Modelo de Interação

- Os diagramas da UML que dão suporte à modelagem de interações são os **diagramas de interação**.

Diagramas de interação representam como o sistema age **internamente** para que um ator atinja seu objetivo na realização de um caso de uso. A modelagem de um sistema de OO normalmente contém diversos diagramas de interação. O conjunto de todos os diagramas de interação de um sistema constitui o seu ***modelo de interações***.

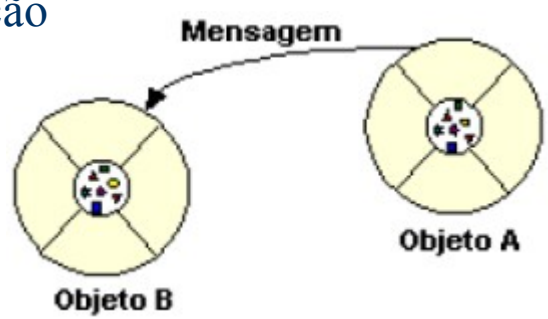
Interações através de mensagens

- **Mensagem** ⟶ conceito básico da interação entre objetos
- O modelo de interações representa **as mensagens** trocadas entre **os objetos** para a execução de **cenários dos casos de uso** do sistema.

> Uma **mensagem** representa a requisição de um **objeto remetente** a um **objeto receptor** para que este último execute alguma **operação** definida para sua classe. Essa mensagem deve conter **informação** suficiente para que a operação do objeto receptor possa ser executada.

Interações através de mensagens

- **Quando** um objeto envia uma mensagem para outro?
- Na construção de diagramas de interação, mensagens de um objeto a outro implicam **operações** que classes devem ter.
- A entrega de uma mensagem a um objeto é concretizada através de uma chamada a um **método**
  - o **objeto** a que se destina a mensagem,
  - o nome da **operação** a ser executada (método)
  - os **parâmetros** necessários para executar a operação
  - Ex: carro.setPlaca(placa)



Tipos de Mensagens

- **mensagem assíncrona:** objeto remetente não espera a resposta para prosseguir com o seu processamento (operação não bloqueante)

- **mensagem síncrona:** indica que o objeto remetente espera até que o objeto receptor processe a mensagem (bloqueante)

- **mensagem de retorno:** indica o término de uma operação.

Sintaxe da UML para mensagens

- Cada mensagem enviada entre objetos possui um rótulo e deve seguir a seguinte sintaxe:

  [[expressão-seqüência] controle:] [v :=] nome [(argumentos)]

  - Elementos delimitados por colchetes são opcionais. Portanto, somente o nome da mensagem é obrigatório.

- Onde:
  - **expressão-seqüência** serve para eliminar ambigüidades acerca de quando uma mensagem foi enviada em relação as demais.
    - **Ex: 1; 1.1; 1.2; 1.3; 2; 2.1 etc.**
    - **Ex2: 1.1a e 1.1b (envio de mensagem paralelas)**
  - **controle** pode expressar uma *condição* ou *interação*
  - **v - variável** – recebe o valor de retorno

Sintaxe da UML para mensagens

- O **valor de retorno** de uma mensagem síncrona pode ser indicado na chamada, com atribuição **:=**, ou na mensagem de retorno
  - Exemplo: livroReservado := reservado()
  - "livroReservado" poderá ser usado em mensagens e condições posteriores
  - Pode-se escrever "livroReservado" na mensagem de retorno
- Uma **condição** é indicada por guarda entre colchetes **[ ]**
  - Exemplo: [x<0] invert(x,color)
  - A mensagem só é enviada se a condição se verificar
  - Condições permitem mostrar várias sequências alternativas num único diagrama
- Uma **interação** é indicada com asterisco *, seguido ou não de uma fórmula de iteração
  - Exemplo: *[i:=1..n] update(i)

Exemplos (Sintaxe da UML para mensagens)

- *[para cada f em F] desenhar()
- *[enquanto x>0] transformar(x)
- [senha é válida] abrirJanelaPrincipal()
- 1: adicionarItem(item)
- 3 [a > b]: trocar(a, b)
- 2 *: desenhar()
- 2 *[i := 1..10]: desenhar(i)
- 1.2.1: x := selecionar(e)

Elementos dos Diagramas de Interações

- Assim como os outros diagramas da UML, os diagramas de interações possuem um conjunto de elementos gráficos.
- O elementos comuns, além das mensagens, a ambos diagramas são:
  - Atores
    - Normalmente o ator primário é responsável por iniciar a interação entre os objetos.
    - Representados da mesma forma que no DCU.
  - Objetos
    - Objetos são representados em um diagrama de interação utilizando-se a mesma notação do diagrama de objetos.
  - Classes
    - A mensagem pode ser endereçada a uma classe (e não para um objeto), então a própria classe deve ser representada no diagrama.
    - Mensagens para uma classe disparam **operações estáticas**
  - Auto-mensagem (mensagem reflexiva) – quando a mensagem é enviada ao próprio objeto.

Elementos dos Diagramas de Interações

- Podem-se representar objetos anônimos ou objetos nomeados, dependendo da situação.
- A sintaxe para objetos é: [nome_objeto[seletor]] : nome_classe
- nome_objeto: nome da instância do objeto.
- Anônimo :ItemPedido
- Nomeado item:ItemPedido
- em uma coleção item[i]:ItemPedido
  - seletor: serve para fazer referência a uma instância de uma classe que está armazenada em uma coleção de objetos, como uma lista.
- A representação de objetos **é** sublinhada e classe **NÃO**

Tipos de diagramas de interação

- Há dois tipos de diagrama de interação:
  - Diagrama de seqüência
  - Diagrama de comunicação
- O diagrama de seqüência e o diagrama de colaboração são equivalentes entre si.

Diagrama de seqüência: enfoque em como as mensagens são enviadas entre os objetos no decorrer do tempo

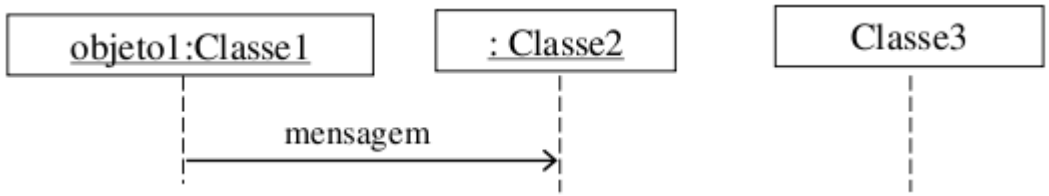
Diagrama de comunicação: enfoque em como as mensagens são enviadas entre objetos que estão relacionados

Diagrama de Sequência

- Elementos básicos em um diagrama de seqüência:
  - Atores
  - Objetos, multiobjetos e classes
  - Linhas de Vida
  - Focos de controle
  - Criação de objetos
  - Destruição de objetos
  - Interações

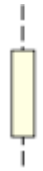
Objetos e linha de vida

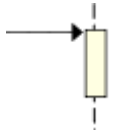
- Cada objeto participante é representado por uma caixa em cima de uma linha vertical tracejada (linha de vida)
- Podem aparecer atores.Normalmente quem inicia as interações.
- O tempo cresce de cima para baixo
  - Note o sublinhado no objeto e não na classe

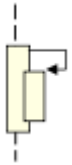


Foco de Controle

- Um **foco de controle (barra de ativação)** mostra o período de tempo durante o qual um objeto está executando uma ação
  - inclui situação em que está á espera de retorno de uma chamada síncrona
  - não inclui situação em que um processo está adormecido à espera de receber uma mensagem assíncrona que o acorde
- Em termos de processos, significa que o objeto tem um processo ou *thread* ativo associado
- Retorno de chamada é implícito no fim do foco de controle
- Chamadas recursivas provocam focos empilhados

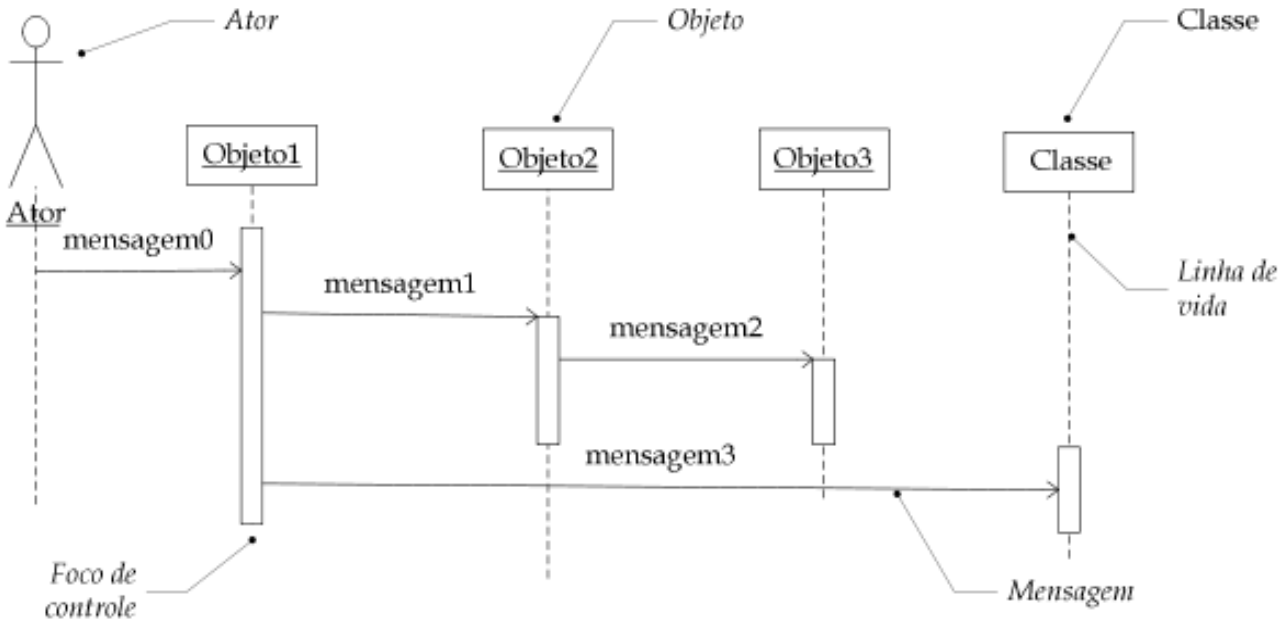






Elementos gráficos de um DS

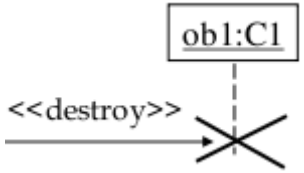
- A posição vertical das mensagens permite deduzir a ordem na qual elas são enviadas.



Criação e destruição de um objeto

- Criação de objeto é representada por mensagem dirigida à própria caixa que representa o objeto (em vez de ser dirigida à linha de vida)
  - Mensagem de criação pode ter estereótipo «create»

- Destruição de objeto é representada por um X no fim da linha de vida do objeto
  - Mensagem de destruição pode ter estereótipo «destroy»
  - Pode ocorrer na recepção de mensagem ou no retorno de chamada
  - Objeto pode auto destruir-se



Criação e destruição de um objeto

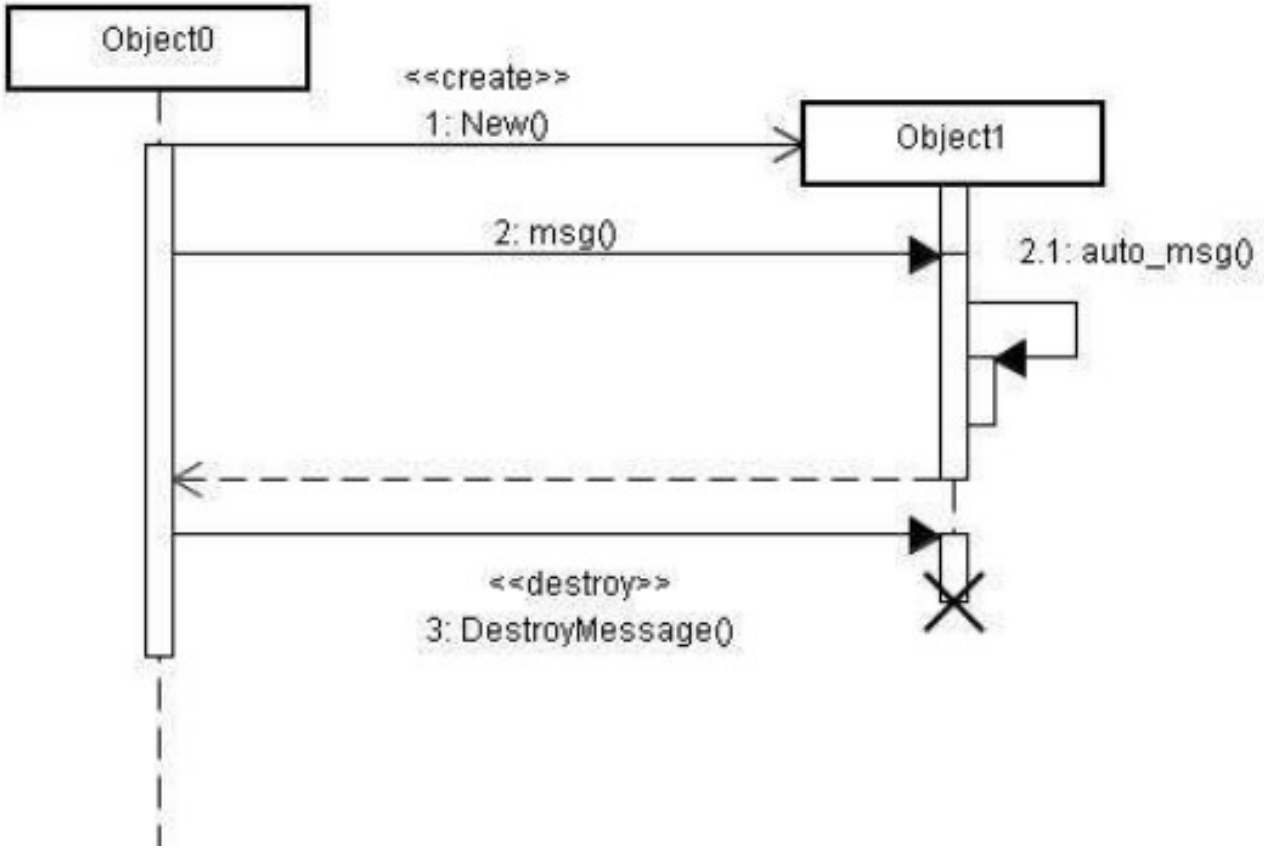


Diagrama de Sequência de Exemplo

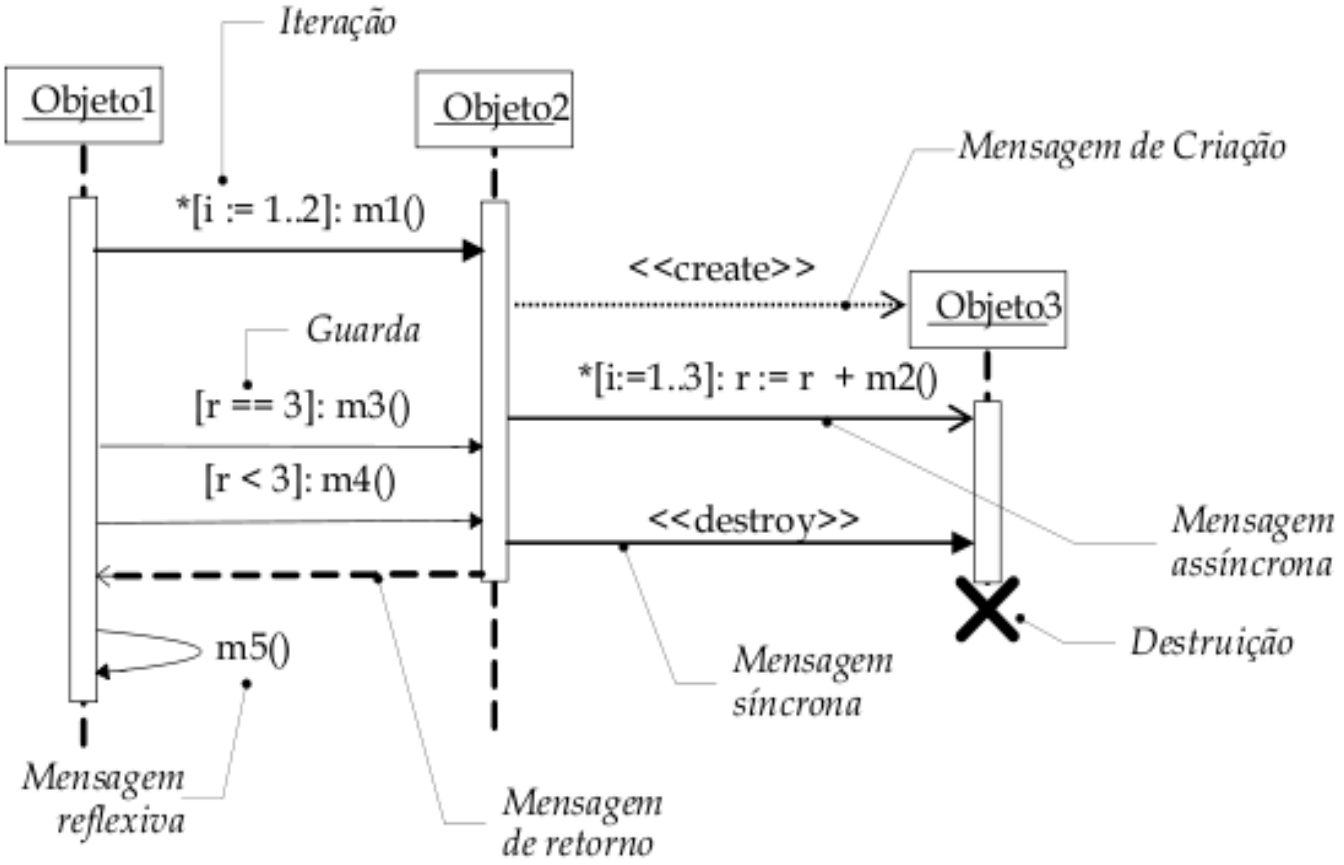


Diagrama de Sequência de Exemplo

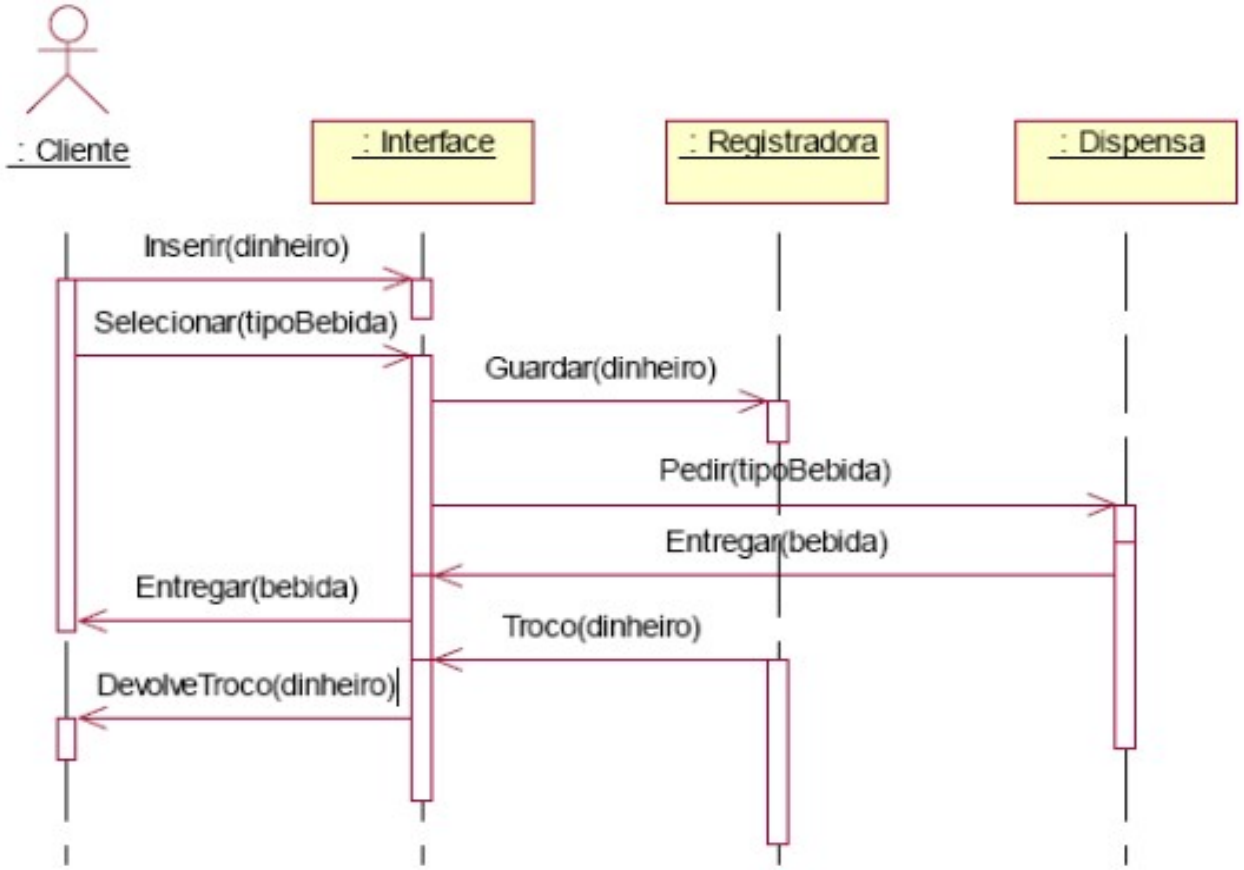


Diagrama de Sequência de Exemplo