Alguns Exemplos de Aplicação da Modelagem de Interação em Desenvolvimento de Software

1) Usando modelos de interação para refinamento de modelos

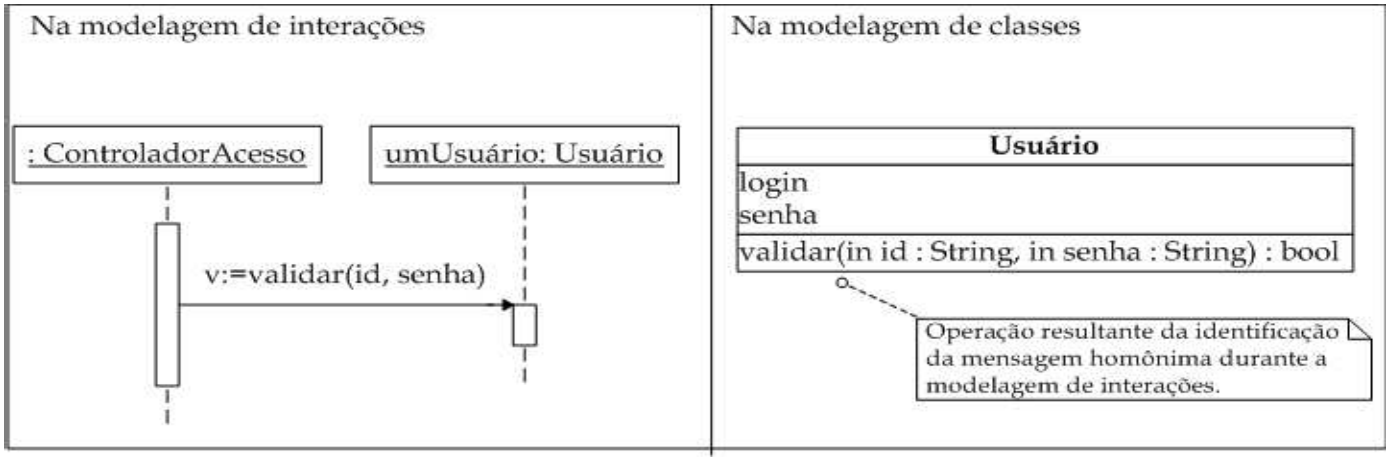
- A modelagem de interação (realizada com os 4 diagramas de interação) pode ser utilizada para se obter informações adicionais para **completar** e **aprimorar** outros modelos

1) Usando modelos de interação para refinamento de modelos

- **Exemplo**: refinando o modelo de classes
  - Durante o desenvolvimento do diagrama de classes surgem dúvidas, tais como:
    - Quais os métodos de uma classe?
    - Para cada método, qual a sua assinatura?
    - Uma classe precisa de mais atributos?

1) Usando modelos de interação para refinamento de modelos

- **Exemplo**: refinando o modelo de classes
  - Ou seja, estamos com dúvidas quanto à responsabilidades de classes
  - O objetivo da modelagem de interações é identificar **mensagens** e, em última análise, **responsabilidades** (atributos e operações).



1) Usando modelos de interação para refinamento de modelos

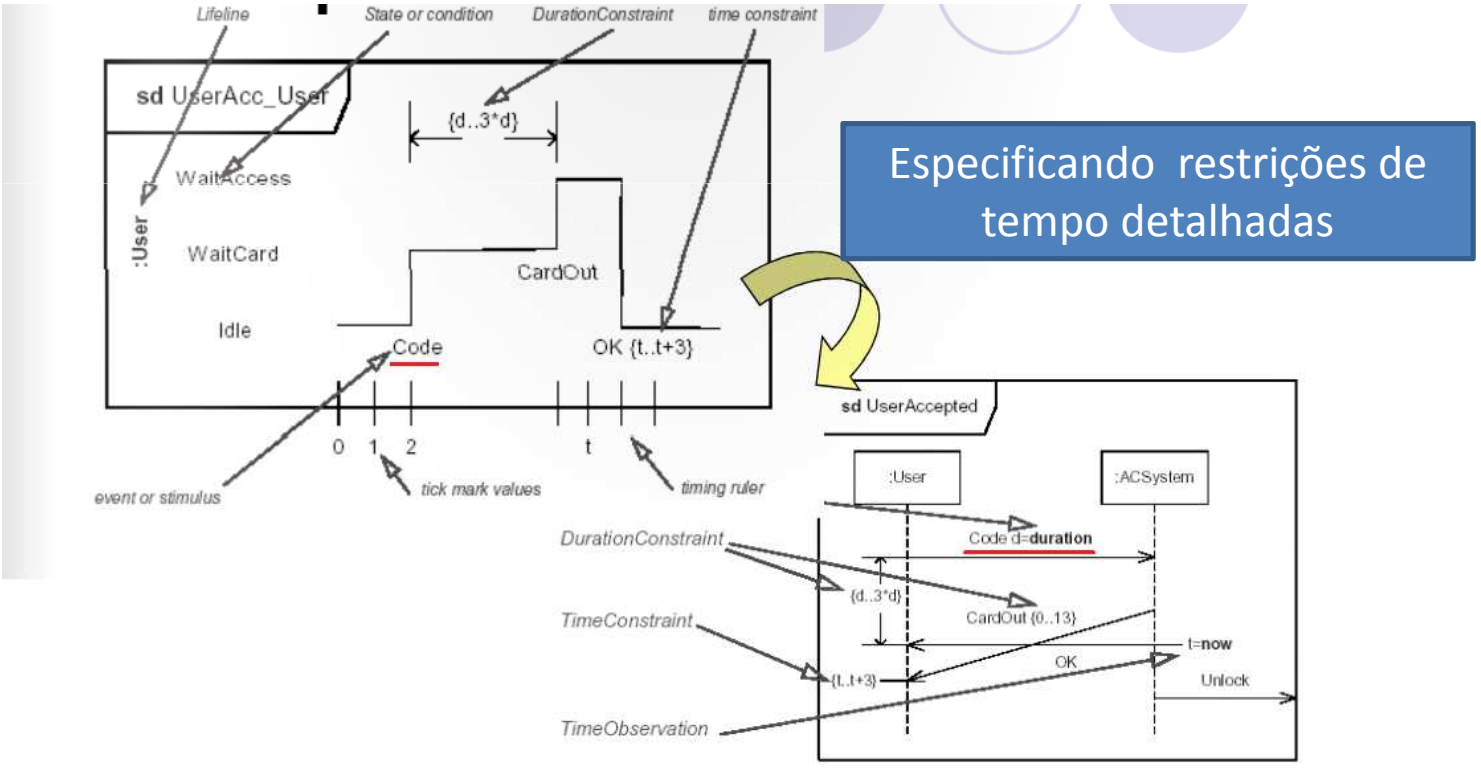
- **Exemplo**: refinando o modelo de classes
  - Assim, durante a construção de diagramas de interação pode-se identificar no diagrama de classes
    - novos atributos
    - novas associações
    - novas operações
    - assinaturas de operações
    - ... e novas classes

1) Usando modelos de interação para refinamento de modelos

- **Exemplo**: refinando o modelo de casos de uso
  - O modelo de interação pode ser utilizado, por exemplo:
    - para validar as interações do usuário com o sistema, e assim, descobrir novos dados que devem ser informados pelo usuário (e não foram percebidos na fase de análise)
    - para aperfeiçoar as descrições de caso de uso (por exemplo, podem existir várias formas de interação do usuário com o sistema, então os modelos de interação podem ajudar a aperfeiçoar o fluxo de eventos nas descrições de casos de uso)

1) Usando modelos de interação para refinamento de modelos

- **Exemplo**: refino entre modelos de interação



Fonte:
https://issuu.com/aldivadyer/docs/5_uml_rev1

Um pouco mais sobre alocação de responsabilidades

- Podemos entender a modelagem de interações como um processo cujo objetivo final é decompor as responsabilidades do sistema e alocá-las à classes
- Dado um conjunto de N responsabilidades, uma possibilidade é criar uma única classe no sistema para assumir com todas as N responsabilidades.
- Outra possibilidade é criar N classes no sistema, a cada um delas sendo atribuída uma das N responsabilidades.
- Certamente, as duas alternativas anteriores são absurdas do ponto de vista prático. Mas, entre as muitas maneiras possíveis de alocar responsabilidades, como podemos saber quais delas são melhores que outras?

Um pouco mais sobre alocação de responsabilidades

- A resposta à pergunta anterior não é nenhuma “receita de bolo”.
- De fato, para construirmos uma bom modelo de interações, devemos lançar mão de diversos princípios de projeto:
- Dois dos principais princípios são o **acoplamento** e a **coesão**.
- A ***coesão*** é uma medida do quão fortemente relacionadas e focalizadas são as responsabilidades de uma classe.
- É extremamente importante assegurar que as responsabilidades atribuídas a cada classe sejam altamente relacionadas.
  - Em outras palavras, o projetista deve definir classes de tal forma que cada uma delas tenha alta coesão.

Um pouco mais sobre alocação de responsabilidades

- O ***acoplamento*** é uma medida de quão fortemente uma classe está conectada a outras classes, tem conhecimento ou depende das mesmas.
- Uma classe com <u>acoplamento fraco</u> (baixo) não depende de muitas outras.
  - Por outro lado, uma classe com <u>acoplamento forte</u> é menos inteligível isoladamente e menos reutilizável.
- Além disso, uma classe com alto acoplamento é mais sensível a mudanças, quando é necessário modificar as classes da qual ela depende.
- **<u>Conclusão</u>**: criar modelos com ***alta coesão*** e ***baixo acoplamento*** deve ser um objetivo de qualquer projetista.

2) Uso de Modelos de Interação em um Processo de Desenvolvimento de Software Iterativo e Incremental

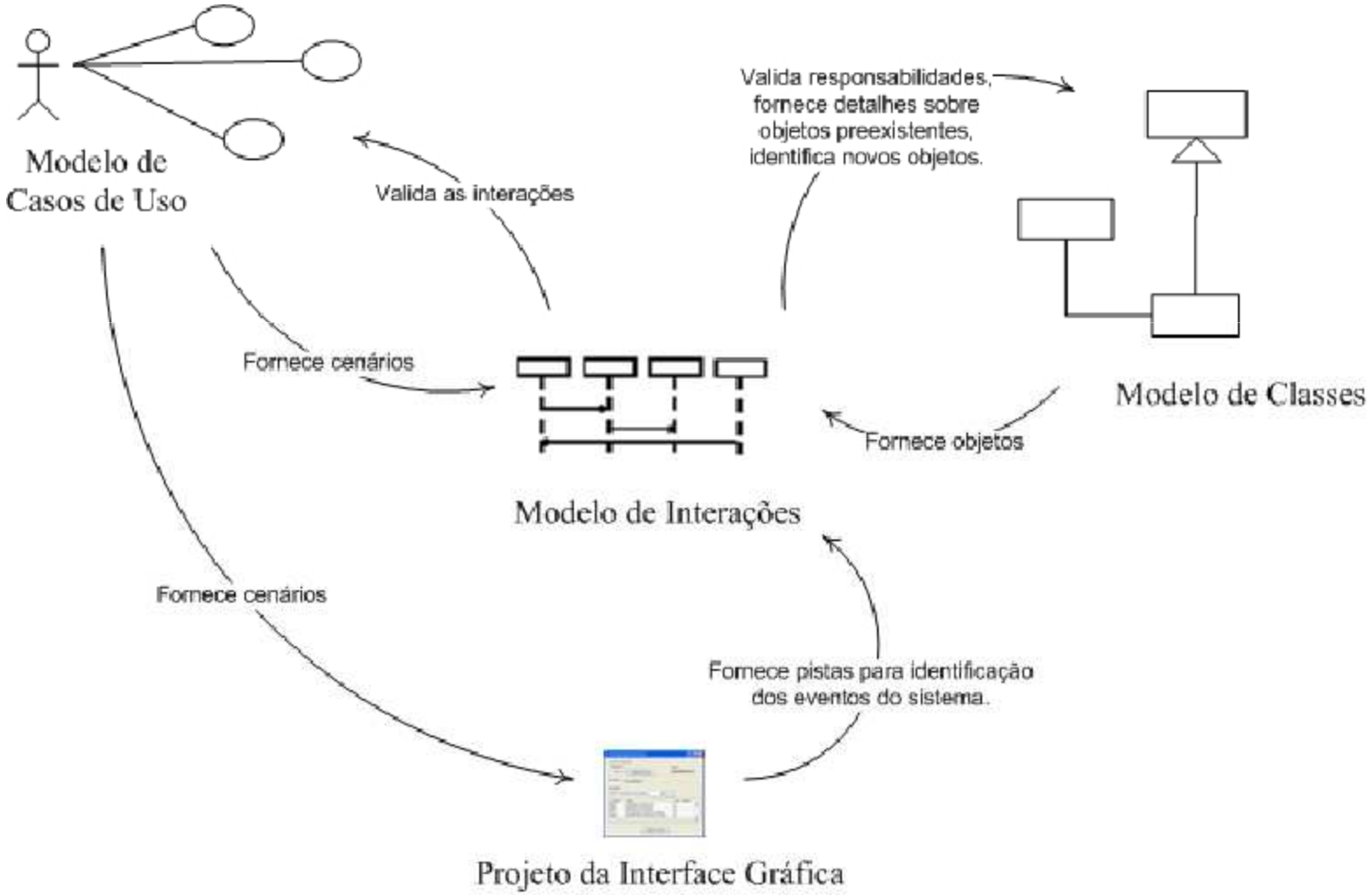
- Modelos de interação são construídos, principalmente, para os cenários de casos de uso.
- Há controvérsias sobre o momento de início da utilização desse modelo (análise vs. projeto).
  - Inicialmente (+análise), pode exibir apenas os objetos participantes e mensagens exibindo somente o nome da operação.
  - Posteriormente (+projeto), pode ser refinado.
    - Criação e destruição de objetos, tipo e assinatura completa de cada mensagem.

2) Uso de Modelos de Interação em um Processo de Desenvolvimento de Software Iterativo e Incremental

- Em processos iterativos e incrementais de desenvolvimento de software é comum ter diagramas em diferentes fases. Por exemplo, modelos de interação podem ser úteis para transformar o modelo de classes de análise no modelo de classes de projeto.

- Assim, o modelo de interação fornece os seguintes itens para refinar o modelo de classes de análise:
  - Detalhamento de métodos
  - Detalhamento de associações
  - Novos métodos
  - Novos atributos
  - Novas classes

2) Uso de Modelos de Interação em um Processo de Desenvolvimento de Software Iterativo e Incremental



3) Uso de Modelos de Interação na Especificação de Casos de Teste de Software

- A modelagem de interação pode ser utilizada no desenvolvimento de casos de teste, principalmente em teste de unidade e em teste de integração

- Por exemplo, um diagrama de comunicação ou de sequência pode mostrar ao analista de testes quais classes e quais mensagens são trocada em um contexto que deve ser verificado – isso pode ajudar na construção de novos casos de teste e na validação de caso de teste criados

3) Uso de Modelos de Interação na Especificação de Casos de Teste de Software

- Atualmente, existem várias pesquisas com propostas de automatização de casos de teste utilizando diagramas de sequência

- U2TP (UML Test Profie): é a extensão da UML para projeto, visualização, especificação, análise, construção e documentação de artefatos de teste de software
    - na U2TP há propostas, por exemplo, de utilização de diagramas de sequência e de visão geral de integração na automatização de testes de software

4) Uso de Modelos de Interação em Desenvolvimento de Software Dirigido por Modelos (DDM)

- O DDM é uma iniciativa da OMG com o intuito de promover toda uma infra-estrutura para desenvolvimento de software

- Almeja solucionar os problemas presentes em desenvolvimento de software (produtividade, interoperabilidade, portabilidade, documentação, etc)

4) Uso de Modelos de Interação em Desenvolvimento de Software Dirigido por Modelos (DDM)

- Atualmente, existem várias pesquisas (e alguns modelos e softwares já utilizados) em Desenvolvimento de Software Dirigido por Modelos
  - Por essa abordagem você vai, por exemplo:
    - Usar uma ferramenta para gerar uma aplicação executável somente informando os modelos diagramas de modelagem do sistema
    - Usar uma ferramenta para gerar (ou validar) um modelo a partir 1 ou mais outros modelos

5) Engenharia de Software Progressiva, Reversa e de Ida e Volta

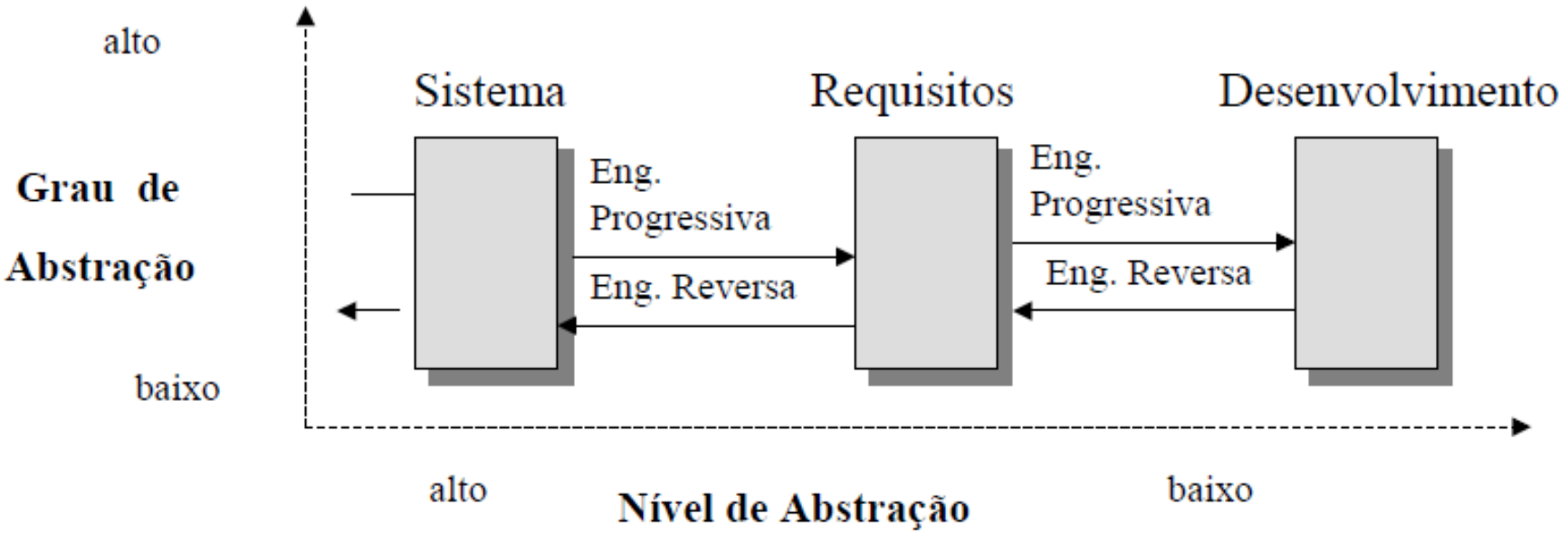
- Ferramentas CASE são ferramentas automatizadas que auxiliam no desenvolvimento de uma ou mais atividades durante o processo de desenvolvimento de software

- No contexto de Ferramentas CASE, temos 3 categorias de engenharia:
  - Engenharia Progressiva
  - Engenharia Reversa
  - Engenharia de Ida de Volta

5) Engenharia de Software
Avante, Reversa e de Ida e Volta

• **Engenharia Progressiva:** Processo tradicional de engenharia de software, caracterizado pelas atividades progressivas do ciclo de vida, que partem de um alto nível de abstração, para um baixo nível de abstração.

•**Engenharia Reversa:** O processo inverso a Engenharia Progressiva, caracterizado pelas atividades retroativas do ciclo de vida, que partem de um baixo nível de abstração para um alto nível de abstração.



5) Engenharia de Software
Avante, Reversa e de Ida e Volta

- Na engenharia progressiva temos, por exemplo, geração de código a partir de diagramas
    - É possível gerar classes Java a partir de um diagrama de classes, ou gerar um script SQL de criação de banco de dados a partir do modelo de implementação do banco de dados

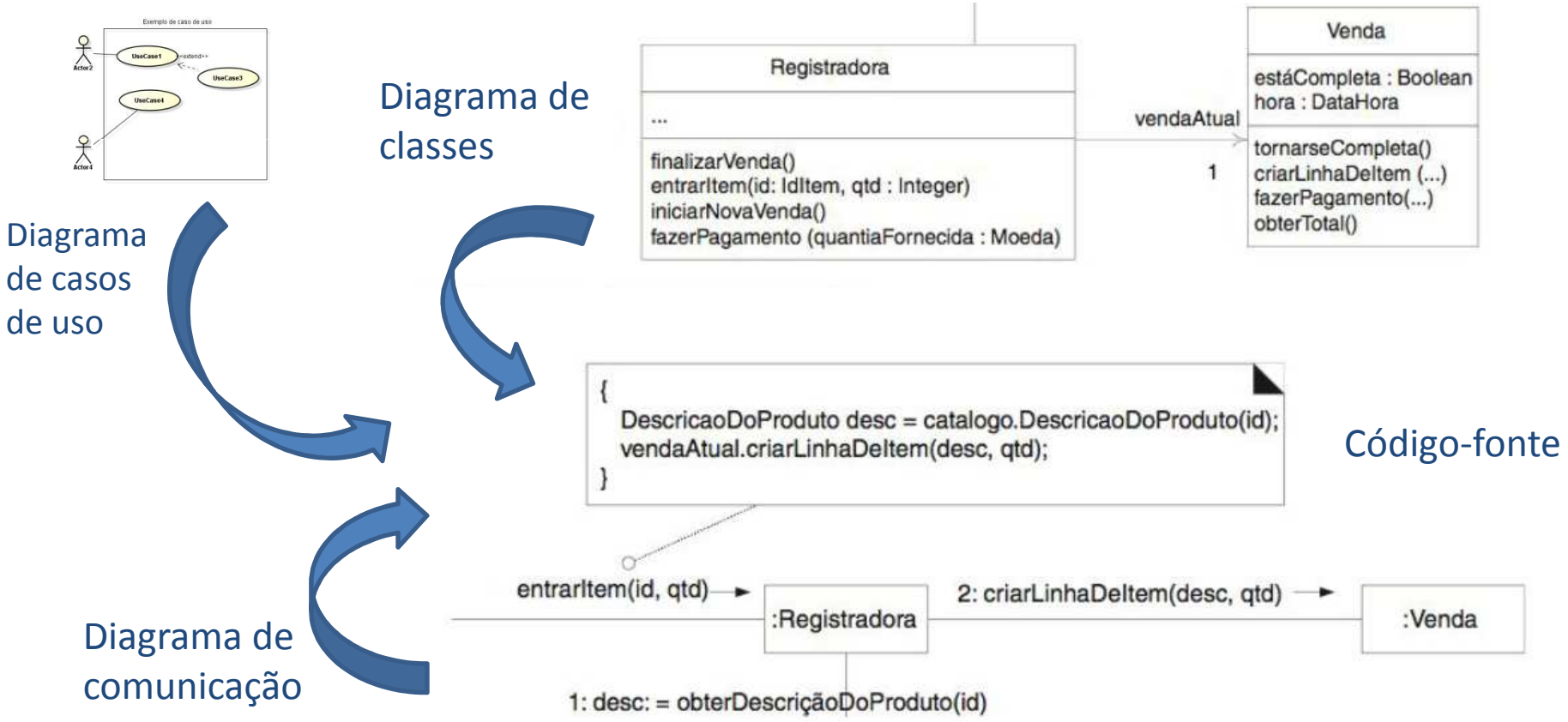
- Na engenharia reversa temos, por exemplo, geração de diagramas de classes e de sequência a partir código-fonte de classes

5) Engenharia de Software Avante, Reversa e de Ida e Volta

- Para completar o círculo, temos também a engenharia de Ida e Volta:
  - Há ferramentas CASE, ainda que limitadas, que conseguem sincronizar alterações progressivas e reversas.
    - Por exemplo, um membro da equipe faz uma alteração no código-fonte, isso se reflete automaticamente nos diagramas de modelagem, e vice-versa
  - É muito comum em projetos de software os diagramas de modelagem e código-fonte ficarem desatualizados com o passar do tempo. Essas ferramentas de Ida e Volta ajudam a minimizar isso

6) Uso tradicional de Modelos de Interação: base para geração de software executável

- O programador constrói o código-fonte da aplicação com base, principalmente, no conjunto de modelos



- Esse material contém partes dos seguintes materiais:

  - Utilizando Padrões de Projeto, Craig Larman, 3ª edição, Bookman, 2007.
  - Princípios de Análise e Projeto de Sistemas com UML - 2ª edição, Eduardo Bezerra, Editora Campus/Elsevier, 2006.
  - http://slideplayer.com.br/slide/3971737/
  - https://bachfatec.files.wordpress.com/2010/09/cap07-modelagem-de-interacoes.ppt
  - http://www.inf.ufpr.br/silvia/ES/reengenharia/reengenharia.pdf