Cartões CRC (Class Responsibility Card)

Responsabilidades

- Em sistemas OO, objetos encapsulam tanto dados quanto comportamento.
- O comportamento de um objeto é definido de tal forma que ele possa cumprir com suas ***responsabilidades***.
- Uma responsabilidade é uma obrigação que um objeto tem para com o sistema no qual ele está inserido.
  - Através delas, um objeto colabora (ajuda) com outros para que os objetivos do sistema sejam alcançados.

Responsabilidades

- O que é **Responsabilidade**?
  - Um contrato ou obrigação de um tipo ou classe.
- Responsabilidades definem o comportamento do objeto!
- Há basicamente dois tipos de Responsabilidades:
  - Fazer
    - O objeto faz algo ele próprio.
    - O objeto inicia ações em outros objetos.
    - O objeto controla e coordena atividades em outros objetos.
  - Conhecer
    - O objeto conhece e gerencia os seus dados privados encapsulados.
    - O objeto conhece objetos relacionados.
    - O objeto conhece coisas que ele pode derivar ou calcular.

Responsabilidades

- Na prática, uma responsabilidade é alguma coisa que um objeto *conhece* ou *faz. (sozinho ou não)*.
  - Um objeto Cliente *conhece* seu nome, seu endereço, seu telefone, etc.
  - Um objeto Pedido *conhece* sua data de realização e sabe *fazer* o cálculo do seu total.
- Se um objeto tem uma responsabilidade a qual não pode cumprir sozinho, ele deve requisitar ***colaborações*** de outros objetos.

Colaboração

Modelo CRC

- Composto de um conjunto de cartões com o seguinte formato:

Classe:

Responsabilidade Colaboração

Ex: CRC

Conta Corrente

	Responsabilidade	**Colaboração**
atributos	Saber o seu saldo	Cliente
	Saber seu cliente	Histórico de Transações
	Saber seu número	
métodos	Manter histórico de transações	
	Realizar saques e depósitos	

Técnica CRC

- Técnica originalmente proposta para ensinar OO → efetiva para profissionais com pouca experiência
- Modelo CRC não pertence a UML, mas tem sido bastante utilizado com XP
- A técnica inclui uma sessão CRC que serve para extração (e também modelagem de requisitos) junto a especialistas de domínio e desenvolvedores.
- Uma sessão CRC envolve por volta de meia dúzia de pessoas: especialistas de domínio, projetistas, analistas e um moderador.
- A cada pessoa é entregue um ***cartão CRC***.

Sessão de CRC

- **Distribuição dos cartões**: considera-se as categorias de responsabilidades.
- Para cada cenário de caso de uso típico, pode-se começar com:
  - um objeto de fronteira para cada ator participante do caso de uso;
  - um objeto de controle para todo o caso de uso;
  - normalmente há vários objetos de entidade.

Modelagem CRC

- Configuração inicial:
  - O moderador da sessão pode desempenhar o papel do objeto controlador
  - Outro participante desempenha o papel do objeto de fronteira.
  - Um outro participante pode simular o ator (ou atores, se houver mais de um).
  - Os demais representam objetos de entidade.

Sessão CRC

- **Seleção do conjunto de cenários** de cada caso de uso a ser simulado
- Para cada cenário, uma sessão CRC é realizada.
  - Se o caso de uso não for tão complexo, ele pode ser analisado em uma única sessão.
- Normalmente já existem algumas classes candidatas para um certo cenário (identificada a partir do próprio caso de uso, descrição funcional, etc)

Sessão CRC

- Começa com a simulação do ator primário (aquele que inicia o caso de uso).
- Os demais participantes encenam a colaboração entre objetos para que o objetivo do ator seja alcançado.
- Através dessa encenação, as classes, responsabilidades e colaborações são identificadas.

Procedimento CRC

1. Selecionar um conjunto de cenários de casos de uso.
2. Para um dos cenários:
   a) Examinar a sua seqüência de passos para identificar as responsabilidades do sistema em relação a cada um desses passos.
   b) Identificar classes relevantes que devem cumprir com as responsabilidades.
3. Repetir o passo 2 para o próximo cenário e modificar a alocação de responsabilidades e a definição de classes.

O que considerar

- CRC são bons para discussão mas não documentação
- As responsabilidades devem ser descritas de forma breve e associadas considerando a especialidade da classe.
- Uma classe não deve ter muitas responsabilidades, deve-se distribuir a inteligência do sistema
  - Utilizam-se cartões de tamanho fixo (normalmente com as dimensões aproximadas de 10cm x 15cm). O fato de as dimensões utilizadas serem as mesmas para todos os cartões contribui para uma distribuição mais uniforme das responsabilidades.
  - Cuidado para não criar “God Classes”
- Agrupar as responsabilidades conceitualmente relacionadas

=> *Considerar coesão*

O que considerar

- Evitar responsabilidades redundantes
  - Não se deve registrar responsabilidades já presentes nas superclasses.
  - Não se deve criar classes mágicas, com responsabilidades inúteis ou desconectadas do mundo real.
  - Obs.: Os cartões CRC para superclasses devem conter o nome da classe e os nomes das suas subclasses. Se a classe possuir superclasses, elas também devem ser registradas abaixo do nome da classe.
- As responsabilidades das classes só poderão ser estabelecidas de uma forma mais completa na fase de projeto, com a elaboração dos diagramas de interação, dos quais as operações (e métodos) serão refinados.
- O comportamento das classes é descrito pelos diagramas de estado (e/ou atividades).

Referências

- Bezerra, E. Princípios de Análise e Projeto Orientados a Objetos com UML. Ed. Campus, Cap 5.
- The Object Primer 3rd Edition: Agile Model Driven Development with UML 2, Cambridge University Press, 2004, Cap 8