Aula 019

Professores:

Geraldo Xexéo

Geraldo Zimbrão

Conteúdo:

Breve introdução à OCL
(Object Constraint Language)

Introdução

- OCL é a linguagem provida pela UML para a especificação de restrições e derivações
  - Orientada a objetos
  - Declarativa, sem efeitos colaterais
  - Sintaxe simples
  - Fácil implementação
- Vários autores sugerem o seu uso para a modelagem de regras de negócio

Por que OCL?

- Os diagramas da UML não são expressivos o suficiente para conter todos os detalhes
  - Em particular, não são suficientes para modelar as regras de negócio de um sistema
- Usar linguagem natural é a alternativa mais comum
- Porém usar OCL apresenta algumas vantagens
  - Automatização de procedimentos
  - Clareza sem ambiguidades

Estrutura Geral

- Uma restrição em OCL deve estar sempre relacionada a uma classe do modelo
  - Chamamos essa classe de âncora ou contexto
- OCL usa apenas expressões
  - Não há comandos nem estruturas de controle
  - Não é permitido alterar valores de atributos (sem efeitos colaterais)
  - As expressões podem ser derivativas ou restritivas

Expressões Derivativas

- Definem como um valor derivado pode ser calculado ou inferido a partir de outros atributos
  - Fórmulas matemáticas
  - Especificação de métodos
- Retornam um resultado
  - Numérico, cadeias de caracteres, categoria, objetos ou conjunto de objetos

Expressões Restritivas

- Devem retornar sempre um valor lógico: verdadeiro ou falso
- Impedem que uma atualização ocorra quando um valor falso é retornado
  - Violação de uma restrição
  - Notificam o sistema do ocorrido para que as medidas cabíveis sejam tomadas

Uma Derivação em OCL

```
context Contrato::duracao: integer
post:
    result = dataFim - dataInicio
```

- ***context***, ***post*** e ***result*** são palavras reservadas
- Após ***context*** devemos ter o nome da classe, o nome do método e o tipo a ser retornado
- ***post*** indica a expressão que será calculada
- ***result*** é o valor retornado
- ***dataInicio*** e ***dataFim*** devem ser atributos ou métodos da classe ***Contrato***

Um Restrição em OCL

```
context Contrato inv:
    duracao >= 0
```

- Uma restrição possui tipo implícito
  - A palavra reservada ***inv*** é utilizada para indicar restrições
- Após ***context*** devemos ter o nome da classe
- A expressão após ***inv*** é a restrição, e deve ser sempre verdadeira
  - Na expressão podemos usar qualquer atributo ou método da classe ***Contrato***

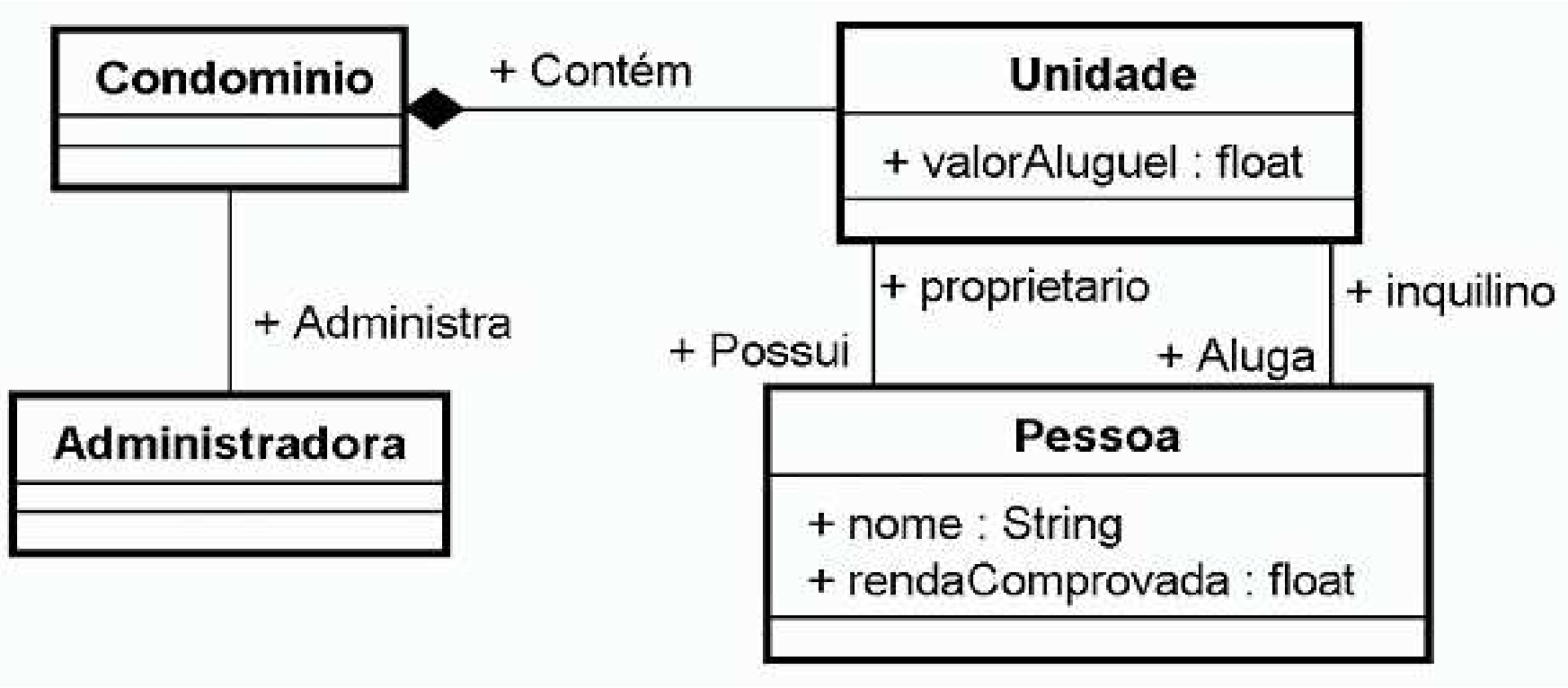
Restrições são Declarativas

- "A duração de um contrato" deve ser maior ou igual a zero dias
- Uma restrição em OCL não especifica quando nem como uma restrição deve ser observada
  - Apenas especifica que ela deve ser sempre verdadeira
  - É possível inferir, nesse exemplo, que sempre que a data de inicio ou a data de fim do contrato for alterada essa restrição deve ser verificada
  - Usando OCL isso pode ser feito automaticamente

Navegação "por ponto"

- OCL utiliza o conceito de navegação "por ponto"
  - Se quisermos utilizar um atributo de uma outra classe do modelo em uma expressão simplesmente devemos acrescentar um caminho através do modelo utilizando "pontos"
  - Esse caminho é formado pelos nomes dos relacionamentos entre as classes separados por pontos

Exemplo - Administradora



Exemplo 1

- Podemos especificar que o proprietário de uma unidade não pode ser o mesmo que a aluga

```
context Unidade inv:
    proprietario.nome <> inquilino.nome
```

- Usamos o nome do relacionamento na direção desejada e podemos entao usar atributos da classe ***Pessoa*** numa restrição ancorada a ***Unidade***
- Não é necessário se preocupar com a cardinalidade

Exemplo 2

- OCL possui uma série de operações predefinidas
  - ***sum*** soma valores de uma coleção

```
context Pessoa::totalAlugueis
post:
    result = possui.valorAlugado->sum()
```

Exemplo 3

Se quisermos especificar que uma pessoa não pode comprometer mais do que 30% da renda comprovada em alugueis...

```
context Pessoa inv:
    0.30 * rendaComprovada >=
    aluga.valorAluguel->sum()
```

Exemplo 4

Na restrição anterior, para sermos mais completos, podemos incluir na renda comprovada os aluguéis que essa pessoa possa vir a ter sendo proprietario de outras unidades

```
context Pessoa inv:
    0.30 * rendaComprovada    +
    totalAlugueis             >=
    aluga.valorAluguel->sum()
```

Verificação das Restrições

- A restrição não especifica quando nem como ela deve ser avaliada
  - No exemplo 4, devemos verificar se a regra foi violada quando:
    - A renda comprovada for alterada
    - A pessoa alugar um imóvel
  - Menos óbvios:
    - A pessoa deixar de ser proprietaria de um imóvel
    - O valor do aluguel de um imovel for alterado

Referências

- OCL é uma linguagem com uma grande quantidade de operações predefinidas
- Mostramos apenas uma breve introdução a OCL
  - Foco em regras de negócio
- Para maiores detalhes, consultar a especificação da linguagem:

  http://www.omg.org/docs/ptc/03-10-14.pdf