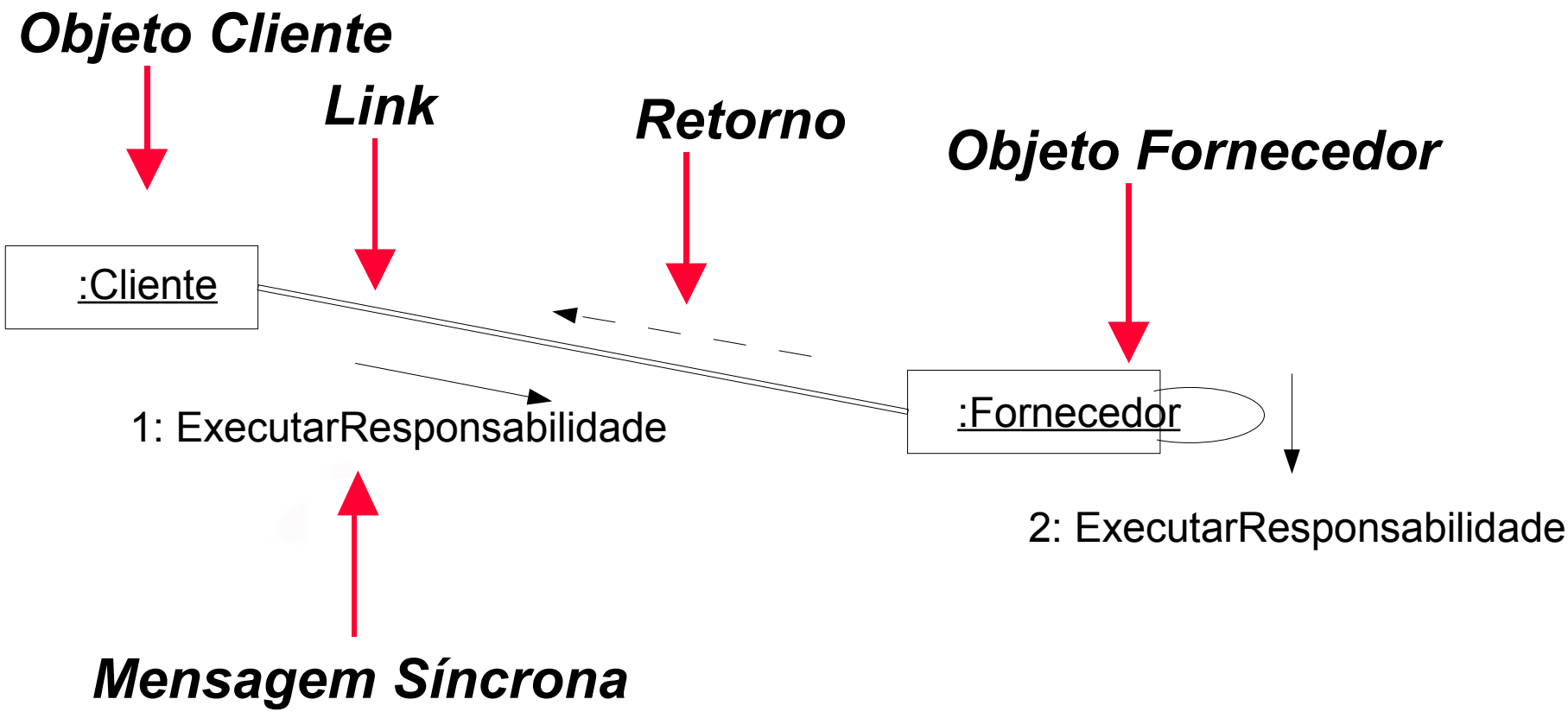
Lição 3

Engenharia de Requisitos – Parte 3

Diagrama de Colaboração

- Seu propósito é modelar um padrão de interação entre objetos
- Modela a participação dos objetos na interação através dos links entre eles e através das mensagens que enviam entre si
- Valida o diagrama de classe já que mostra a necessidade de cada associação
- Modela a mensagem do receptor a qual define uma interface para aquele objeto

Diagrama de Colaboração



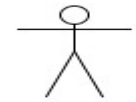
Construindo um Diagrama de Colaboração

- Passo 1: Desenhe os atores e objetos participantes
- Passo 2: Desenhe um link entre atores e objetos
- Passo 3: Escreva as mensagens nos links

Exemplo de cenário:

Atleta envia formulário de aplicação

Primeiro Passo



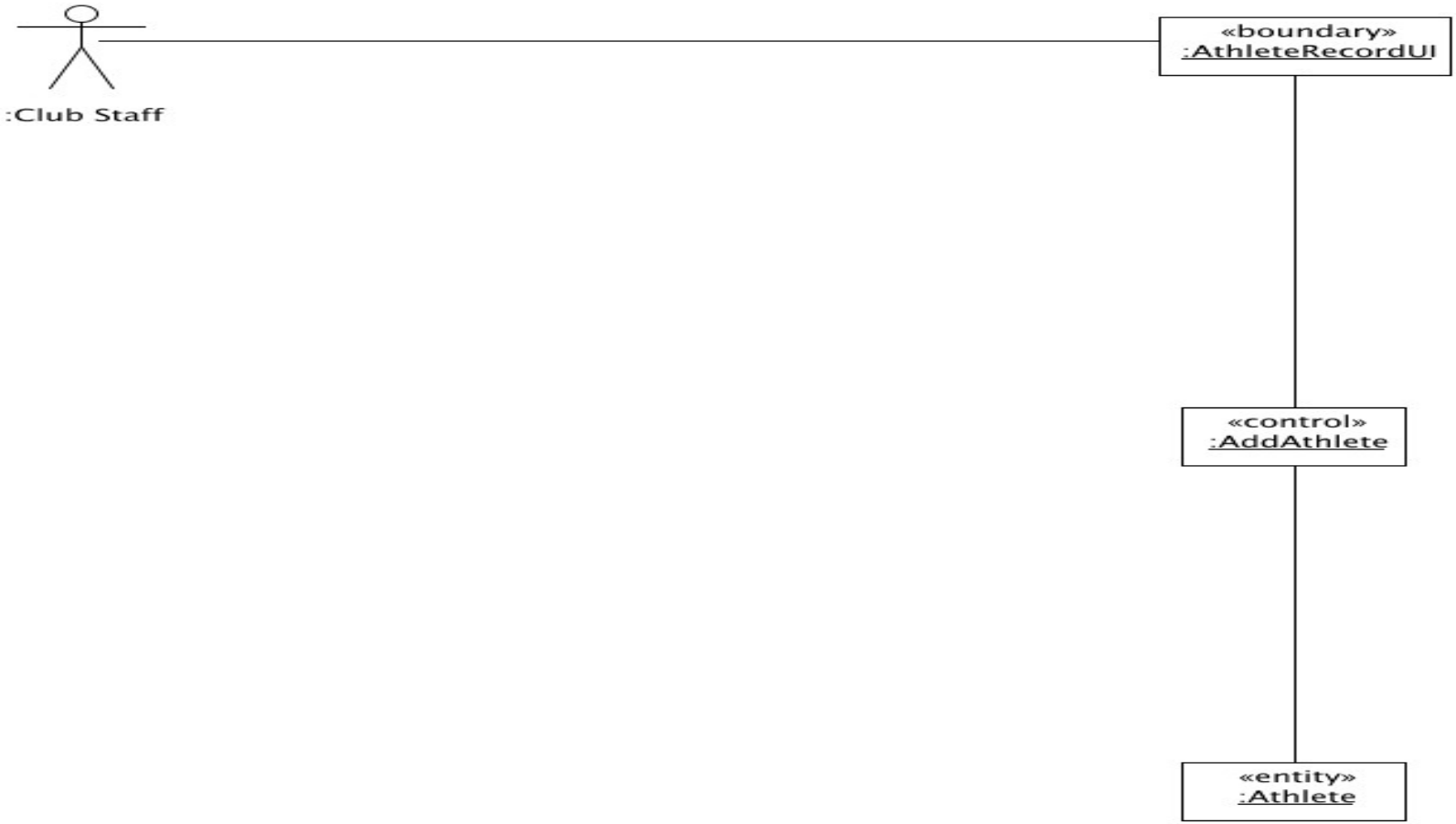
:Club Staff

«boundary»
:AthleteRecordUI

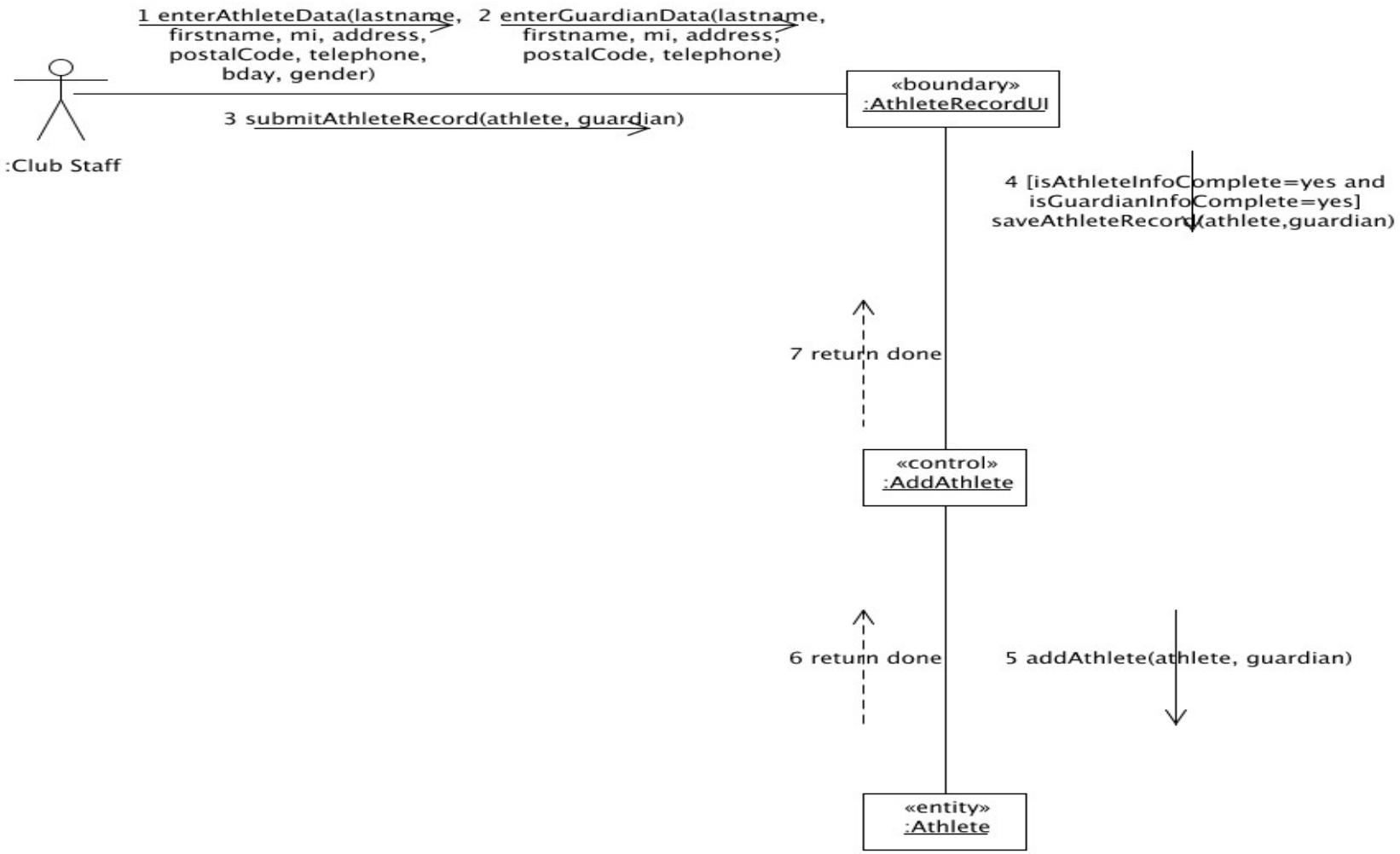
«control»
:AddAthlete

«entity»
:Athlete

Segundo Passo



Terceiro Passo



Passo 4: Para cada classe de análise, identifique responsabilidades

- Esse passo identifica as responsabilidades que cada classe de análise executa ou fornece

Responsabilidades

- Especifica algo que um objeto pode fornecer
- Pode ser:
  - Uma ação que um objeto pode executar
  - Conhecimento que um objeto mantém e fornece a outros objetos
- Pode ser derivada de mensagens obtidas de Diagramas de Eventos ou Interação
- Para cada mensagem, examine a classe do objeto para o qual a mensagem foi enviada; a mensagem pertence ao objeto receptor, logo, a responsabilidade também
- Se a responsabilidade não existe, crie uma que forneça o comportamento
- Verifique também os requesitos não-funcionais para identificar outras responsabilidades

Identificação de Responsabilidades

- Cheque se as classes têm responsabilidade consistente
- Cheque as classes para responsabilidades similares
- Melhor distribuição de responsabilidades pode ser evidente enquanto se trabalha em outro diagrama de interação
- Não há problema se uma classe tem apenas uma responsabilidade, por si, mas ela deve levantar a questão do porque é necessária

Responsabilidades

«boundary» AthleteRecordUI
public void enterAthleteData(String lastname, String firstname, String mi, String address, String postalCode, String telephone, Date bday, Char gender) public void enterGuardianData(String lastname, String firstname, String mi, String address, String postalCode, String telephone) public void submitAthleteRecord(Athlete a, Guardian g)

«control» AddAthlete
public void saveAthleteRecord(Athlete a, Guardian g)

«entity» Athlete
public void addAthlete(Athlete a) public void editAthlete(Athlete a) public void deleteAthlete(Athlete a) public void updateStatus(Athlete a, String newStatus)

«entity» Guardian
public void addGuardian(Guardian g) public void editGuardian(Guardian g) public void deleteGuardian(Guardian g)

Passo 5: Para cada classe de análise, identifique atributos

- O propósito desse passo é identificar a informação que a classe de análise é responsável por manter
- Esse passo identifica os atributos de cada classe de análise

Atributo

- É uma informação que o objeto possui ou conhece sobre si mesmo
- É usado para armazenar informação
- Um atributo tem o seguinte formato:

  atributo : tipo_do_dado = valor_default {observações}

- Exemplo:

  InvoiceNumber : numérico

  Temperature : numérico { 4 decimal precision }

  SequenceNumber : integer = 0 {sequencial automático}

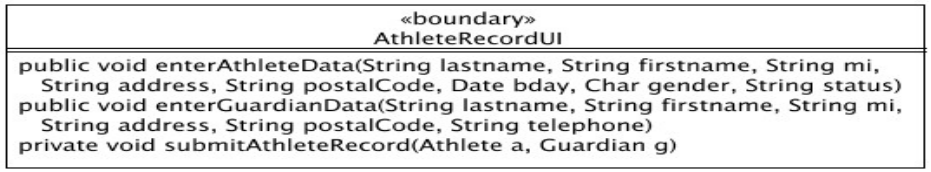
Identificação de Atributos

- Fonte de possíveis atributos são domínio de conhecimento do sistema, requisitos e glossário
- Atributos são usados ao invés de classes quando:
  - Apenas o valor da informação, não sua localização, é importante
  - A informação pertence unicamente ao objeto ao qual ela pertence; nenhum outro objeto referencia a informação
  - A informação é acessada por operações as quais apenas obtém, modificam ou executam simples tranformações na informação; a informação não tem outro comportamento senão prover seu valor
- Se a informação tiver um comportamento complexo, ou é comportilhada por dois ou mais objetos, a informação deve ser modelada como uma classe separada

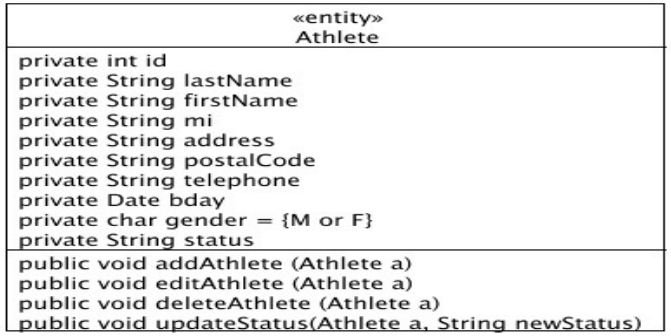
Identificação de Atributos

- Atributos são dependentes do domínio
- Atributos devem apoiar pelo menos um caso de uso

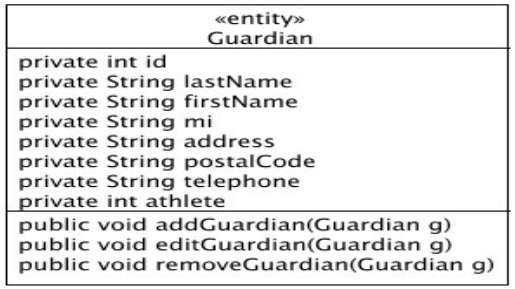
Atributos


«boundary»
AthleteRecordUI
public void enterAthleteData(String lastname, String firstname, String mi, String address, String postalCode, Date bday, Char gender, String status)
public void enterGuardianData(String lastname, String firstname, String mi, String address, String postalCode, String telephone)
private void submitAthleteRecord(Athlete a, Guardian g)


«control»
AddAthlete
public void saveAthleteRecord(Athlete a, Guardian g)


«entity»
Athlete
private int id
private String lastName
private String firstName
private String mi
private String address
private String postalCode
private String telephone
private Date bday
private char gender = {M or F}
private String status
public void addAthlete (Athlete a)
public void editAthlete (Athlete a)
public void deleteAthlete (Athlete a)
public void updateStatus(Athlete a, String newStatus)


«entity»
AthleteStatus
private String code
private String name


«entity»
Guardian
private int id
private String lastName
private String firstName
private String mi
private String address
private String postalCode
private String telephone
private int athlete
public void addGuardian(Guardian g)
public void editGuardian(Guardian g)
public void removeGuardian(Guardian g)

Passo 6: Identificar Associações

- O propósito desse passo é identificar outras classes das quais a classe de análise dependa

- Define o relacionamento estrutural de objetos dentro do conjunto das classes de análise

Associação

- Representa os relacionamentos estruturais entre objetos de diferentes classes
- Conecta instâncias de duas ou mais classes por alguma duração
- Pode ser:
  - Associação unária
  - Associação binária (comum)
  - Associação ternária

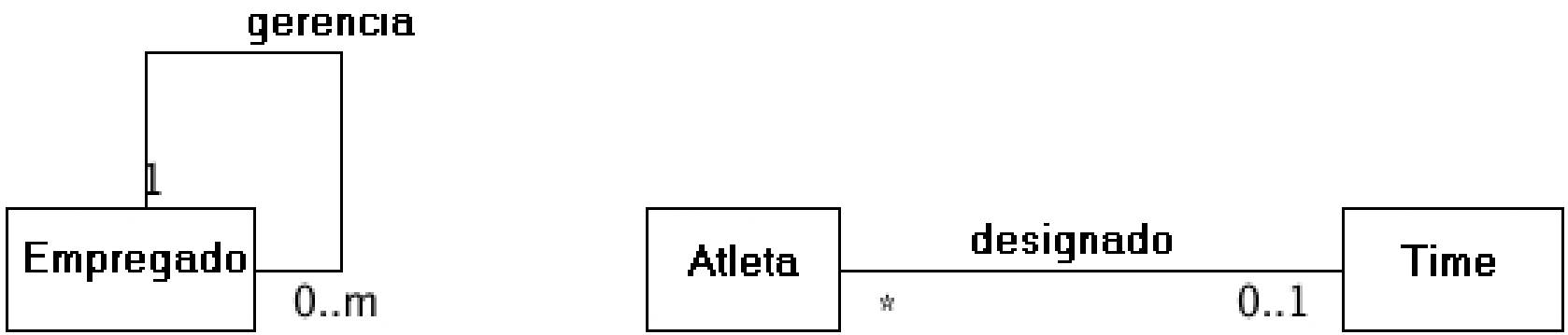
Nomes das Associações

- O nome da associação deve refletir o propósito do relacionamento. Ele pode ser:
  - Uma frase com um verbo
  - Nome de uma função

- Nomes de associações podem ser posicionados sobre ou adjacente ao caminho da associação

- Evite nomes como *tem* ou *contém* pois eles não adicionam informação sobre o que os relacionamentos são entre objetos e classes

Tipos de Associações



Agregação

- É uma forma especial de associação que modela o relacionamento entre um agregado e sua parte

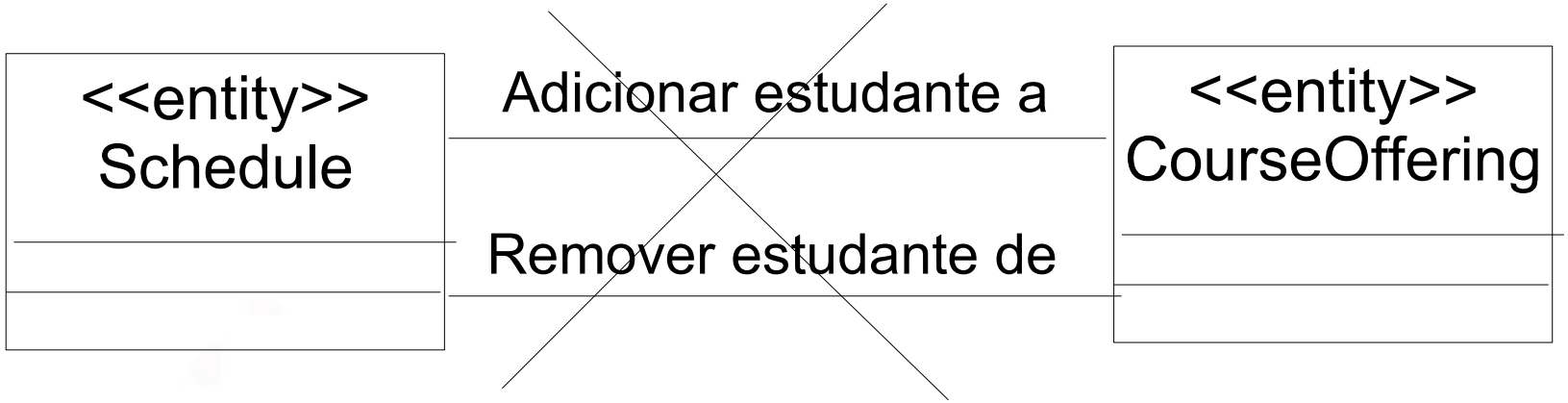
- Algumas situações onde uma agregação é apropriada:
  - Um objeto é fisicamente composto de outros objetos
  - Um objeto é uma coleção lógica de outros objetos
  - Um objeto fisicamente contém outros objetos

Agregação e Funções

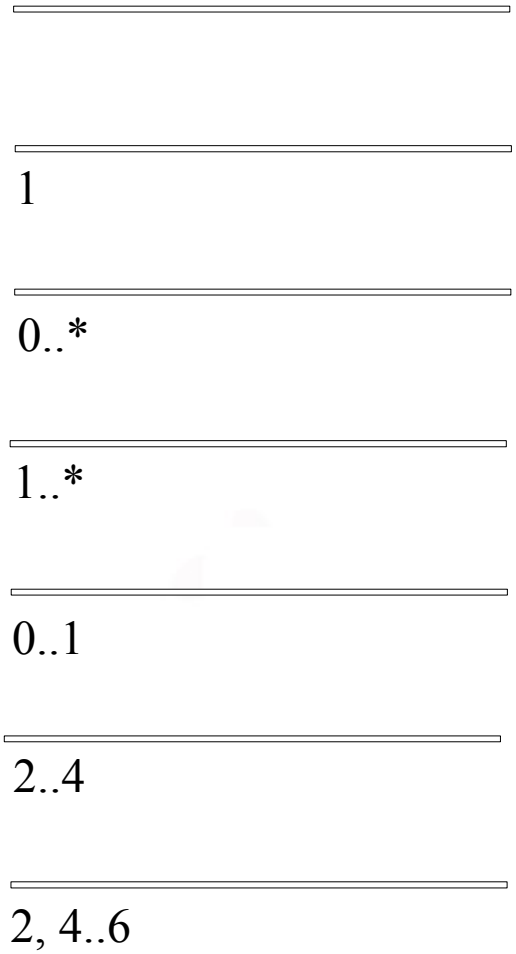
Multiplicidade

- É o número de objetos que uma classe pode associar com outro objeto de outra classe
- É indicada por uma expressão textual na linha de associação
- Responde às seguintes questões:
  - A associação é opcional ou obrigatória?
  - Qual é o número mínimo e máximo de instâncias que podem ser conectar a uma instância?

Multiplicidade

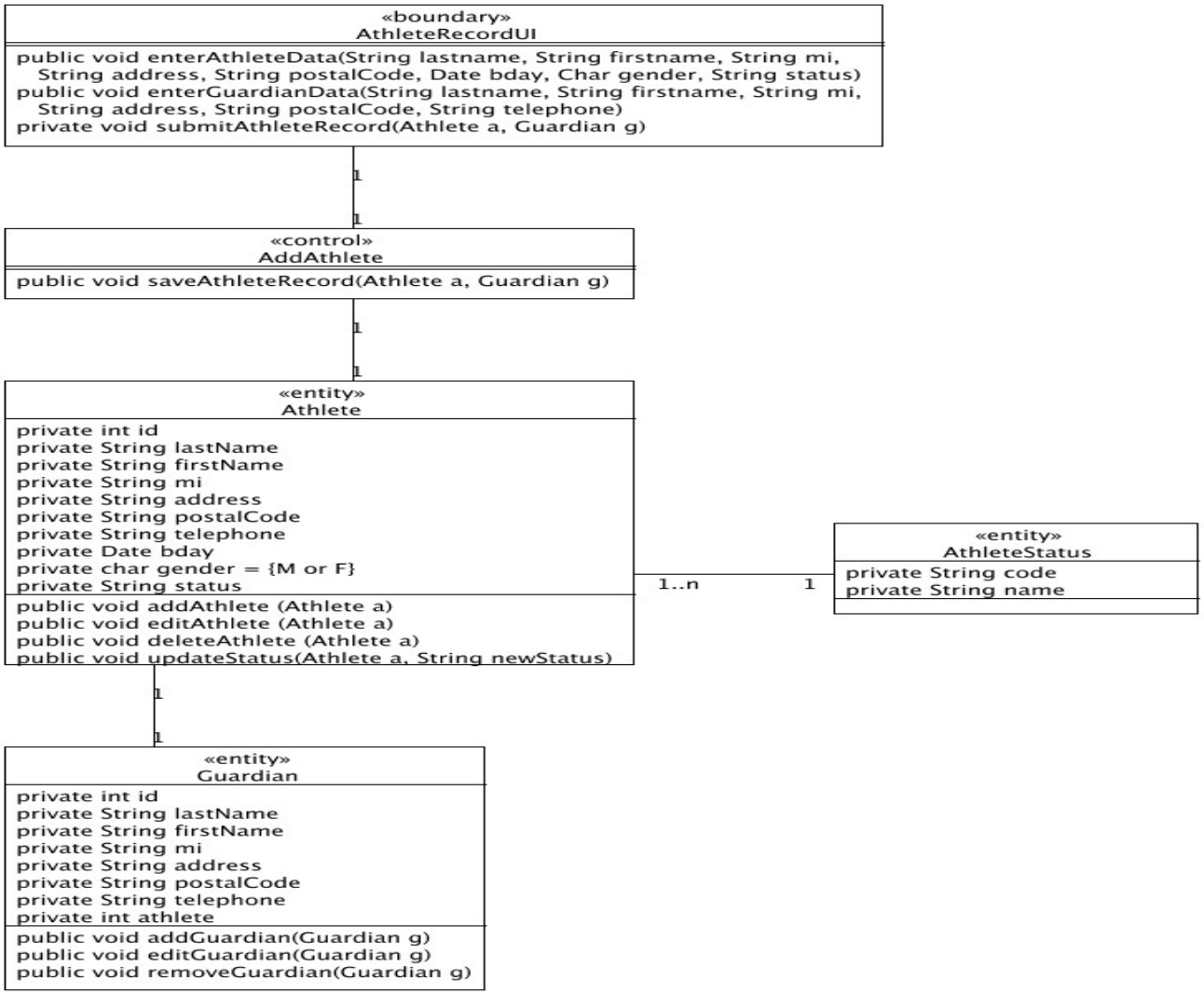


Multiplicidade



- Não especificado
- Exatamente um
- Zero ou mais
- Um ou mais
- Zero ou um
- Intervalo Especificado
- Múltiplos, partes de intervalos

Multiplicidade Ilustrada



Passo 7: Unifique as classes de análise em um diagrama de classe

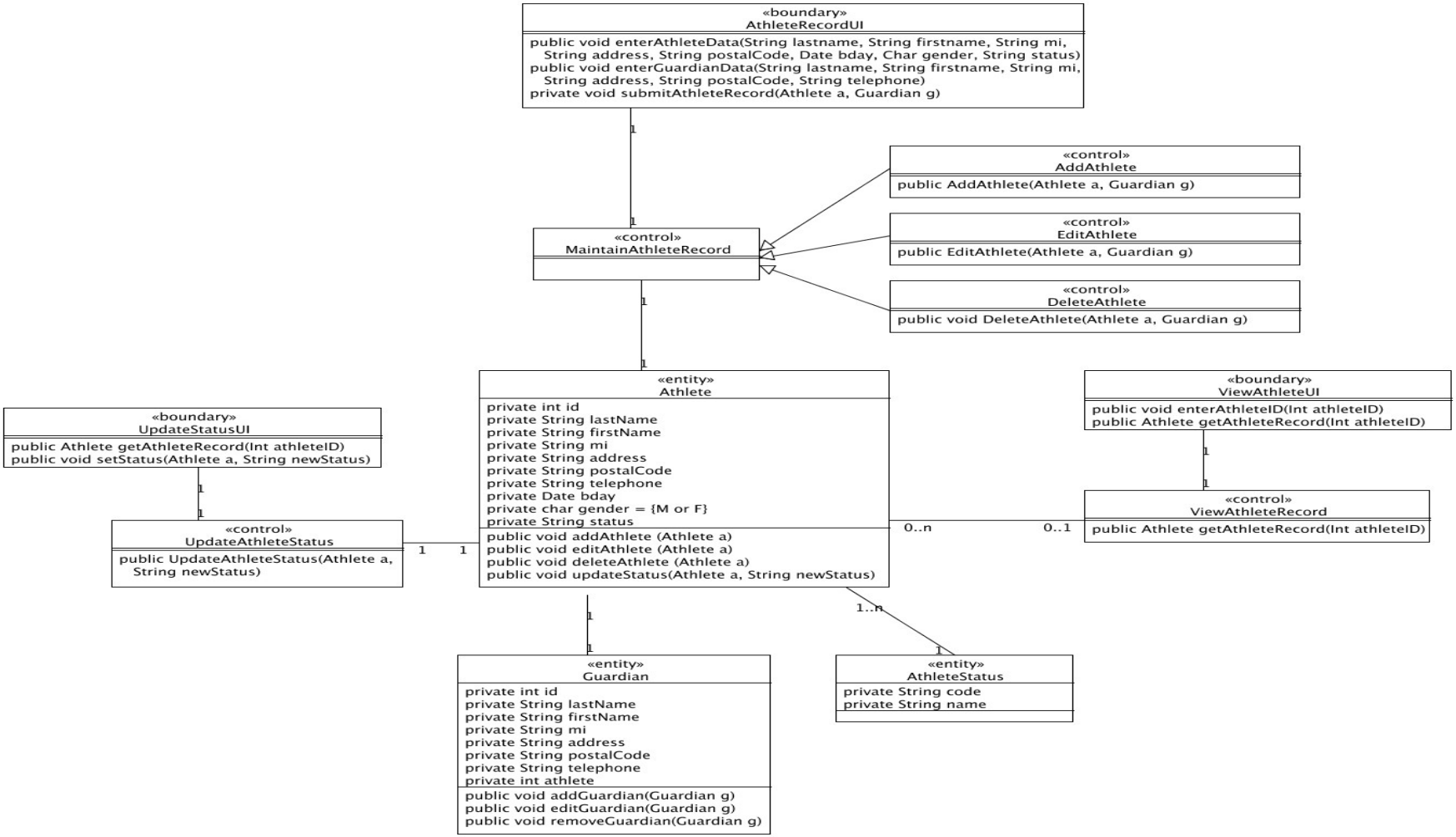
- O propósito desse passo é assegurar que cada classe de análise representa conceitos bem definidos sem sobreposição de responsabilidades

- Filtra as classes de análise para assegurar que um número mínimo de novos conceitos foram criados

Unificação

- O nome da classe de análise deve exatamente capturar a função desempenhada pela classe no sistema. Deve ser único
- Junte classes que possuam comportamento similar
- Una classes de entidade que possuam atributos similares, mesmo que seus comportamentos sejam diferentes; agregue os comportamentos das classes unidas
- Quando atualizar uma classe, você deve atualizar qualquer documentação suplementar necessária
- Avalie o resultado. Tenha certeza que:
  - Verifique que as classes de análise conheçam os requisitos funcionais
  - Verifique que as classes de análise e seus relacionamentos são consistentes com a colaboração que elas fornecem

Manutenção do Modelo de Objeto "Associação ao Clube"



Lista de Validação do Modelo de Análise

- Perguntas para o Modelo de Objeto
- Perguntas para o Modelo Comportamental

Matriz de Rastreabilidade de Requisitos (MRR)

- É a ferramenta usada para gerenciar requerimentos através do processo de desenvolvimento de software
- É a tentativa de desenvolvimento para aproximar a **caixa-branca**
- Seu propósito é capturar requerimentos do usuário-final, e qualquer produto de trabalho desenvolvido deveria estar conectado aos requerimentos

Aproximação da Caixa-Branca

- Refere-se ao desenvolvimento do sistema de requerimentos sem considerar a implementação destes requerimentos

- Usa-se a concepção de ***perspectiva***
  - Conceitual*
  - Especificação*
  - Implementação

Perspectiva Conceitual e de Especificação

- Conceitual – trata de “conceitos no domínio”
- Especificação – trata com as interfaces

Componentes Recomendados para a MRR

- MRR ID
- Requisitos
- Notas
- Requerente
- Data
- Prioridade

Componentes Adicionais da MRR

- MRR ID Relacionado
- Especificação de Trabalho (ET)

Métricas de Requisitos

- Número de Requisitos
  - A medida que o número de requisitos aumenta, dá uma indicação do quão extenso o projeto de software é
  - É uma boa entrada para estimar o esforço do desenvolvimento de software

- Número de Vezes que os Requisitos Modificam-se
  - Um número grande indica um pouco de instabilidade e incerteza em nossa compreensão do que o sistema devia fazer ou como devia se comportar
  - Um número grande significa que tem de refazer a fase de engenharia de requisitos

Teste do Perfil dos Requisitos

- É uma técnica empregada para apontar a facilidades para reorganizar requisitos para o designer e testador

- Se o resultado do teste forem 1 e 2, os requisitos podem ser passados aos dois, designers e testadores. Caso contrário, os requisitos precisam ser reavaliados e reescritos

- A avaliação é subjetiva. Contudo, as pontuações podem prover informações úteis que incentiva uma melhora da qualidade dos requisitos antes de prosseguir o projeto

Escala do Projetista do Sistema

Avaliação do Designer de Sistema	*Descrição*
1	Significa que o projetista entende os requisitos, e deve projetar um requisito semelhante a um dos projetos prévios, e não terá quaisquer problemas com este.
2	Significa que existem aspectos do requisito que são novos para o projetista; porém, não são radicalmente diferentes dos requisitos que tinha realizado com sucesso em projetos anteriores.
3	Significa que existem aspectos dos requisitos que são muito diferentes dos requisitos que o projetista tinha realizado antes; porém, isto é entendido e pode desenvolver um bom projeto.
4	Significa que existem partes deste requisito que não são compreendidas; não é confiante que pode desenvolver um bom projeto.
5	Significa que não entende este requisito e não pode desenvolver um projeto para isto.

Escala do Testador de Sistema

Taxa do Testador de Sistema	*Descrição*
1	Significa que o testador entende os requisitos completamente, e que ele tem que testar um requisito semelhante a um dentre os projetos prévios que ele não terá quaisquer problemas testando este.
2	Significa que existem aspectos do requisito que são novos para o testador; porém, eles não são radicalmente diferentes de requisitos que ele tinha com sucesso, testado nos projetos prévios.
3	Significa que existem aspectos dos requisitos que são muito diferentes dos requisitos que o provador testou antes; porém, ele entende isto e é confiante que ele pode testar isto.
4	Significa que existem partes deste requisito que ele não entende; ele não é confiante que ele pode testar isto.
5	Significa que ele não entende este requisito em nada e ele não pode testar isto.

Sumário

- Engenharia de Requisitos
- Tarefas
  - Concepção, Elucidação, Elaboração, Especificação, Validação, Negociação e Gerenciamento
- Análise de Requisitos
- Modelo de Requisitos
  - Modelo de Caso de Uso
  - Especificações de Caso de Uso
  - Especificação Adicional
  - Glossário ou Definição de Termos
- Lista de Validação de Requisitos

Sumário

- Modelo de Requisitos
- Modelo de Caso de Uso
  - Diagrama de Caso de Uso
  - Especificação do Caso de Uso
- Especificação Adicional
- Glossário
- Modelo de Análise
  - Classes de Análise
  - Diagramas de Classes
- Modelo Comportamental
  - Diagramas de Seqüência
  - Diagramas de Colaboração

Sumário

- Matriz de Rastreabilidade de Requisitos
- Perspectiva de Especificação e Conceitual
- Componentes Recomendados da MRR
- Componentes Adicionais MRR
- Lembretes MRR
- Requisitos Métricos
  - Número de Requisitos
  - Número de Vezes que os Requisitos Modificam-se
- Teste do Perfil dos Requisitos
  - Escala do Projetista de Sistema
  - Escala do Testador do Sistema

Parceiros

- **Os seguintes parceiros tornaram JEDI™ possível em Língua Portuguesa:**