Conteúdo

1. Introdução
2. Levantamento de Requisitos
3. Análise Orientada a Objetos
4. Projeto Orientado a Objetos
5. UML
6. Métodos Ágeis

Projeto Orientado a Objetos

Projeto Orientado a Objetos

Durante o projeto de objeto, é desenvolvida uma solução lógica baseada no paradigma orientado a objetos.

São definidos:

- Diagramas de Interação: ilustram como os objetos colaboram para satisfazer os requisitos.
- Diagrama de Classes de Projeto: define as classes de software e interfaces que serão implementadas.

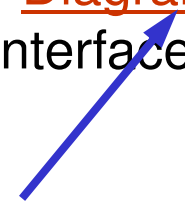


Diagrama de Classes de Projeto

Diagrama de Classes de Projeto

As classes de projeto mostram definições das classes de software e não dos conceitos do mundo real.

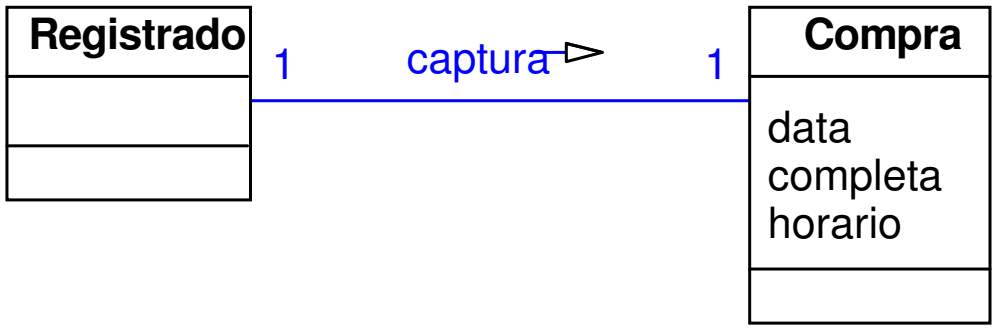
Classe do Modelo de Domínio x Classe do Modelo de Projeto

- Classe do Modelo de Domínio: abstração de um conceito do mundo real tratado pelo sistema.

- Classe do Modelo de Projeto: componente de software.

Diagrama de Classes de Projeto

Classes do Modelo de Domínio x Classes do Modelo de Projeto



Modelo do Domínio

Modelo do Projeto

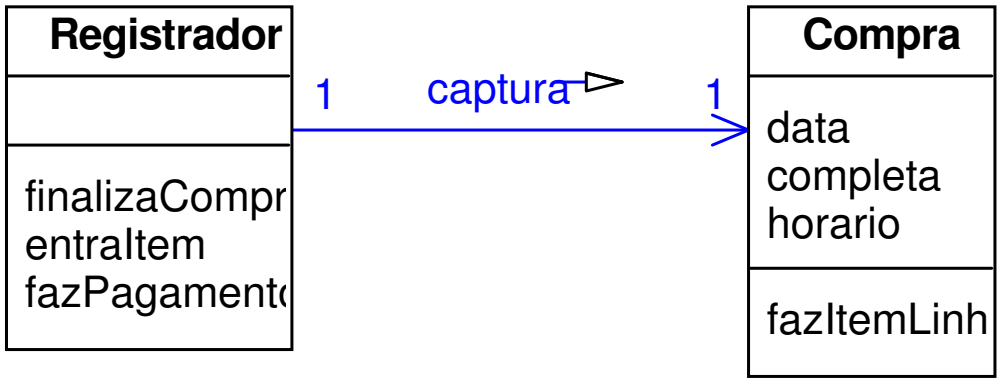


Diagrama de Classes de Projeto

Informações incluídas nos diagramas de classes de projeto:

- Classe, Atributo, Associação, Agregação e Herança
- Operação
- Interface
- Navegabilidade
- Dependência

aparecem no modelo conceitual

Diagrama de Classes de Projeto

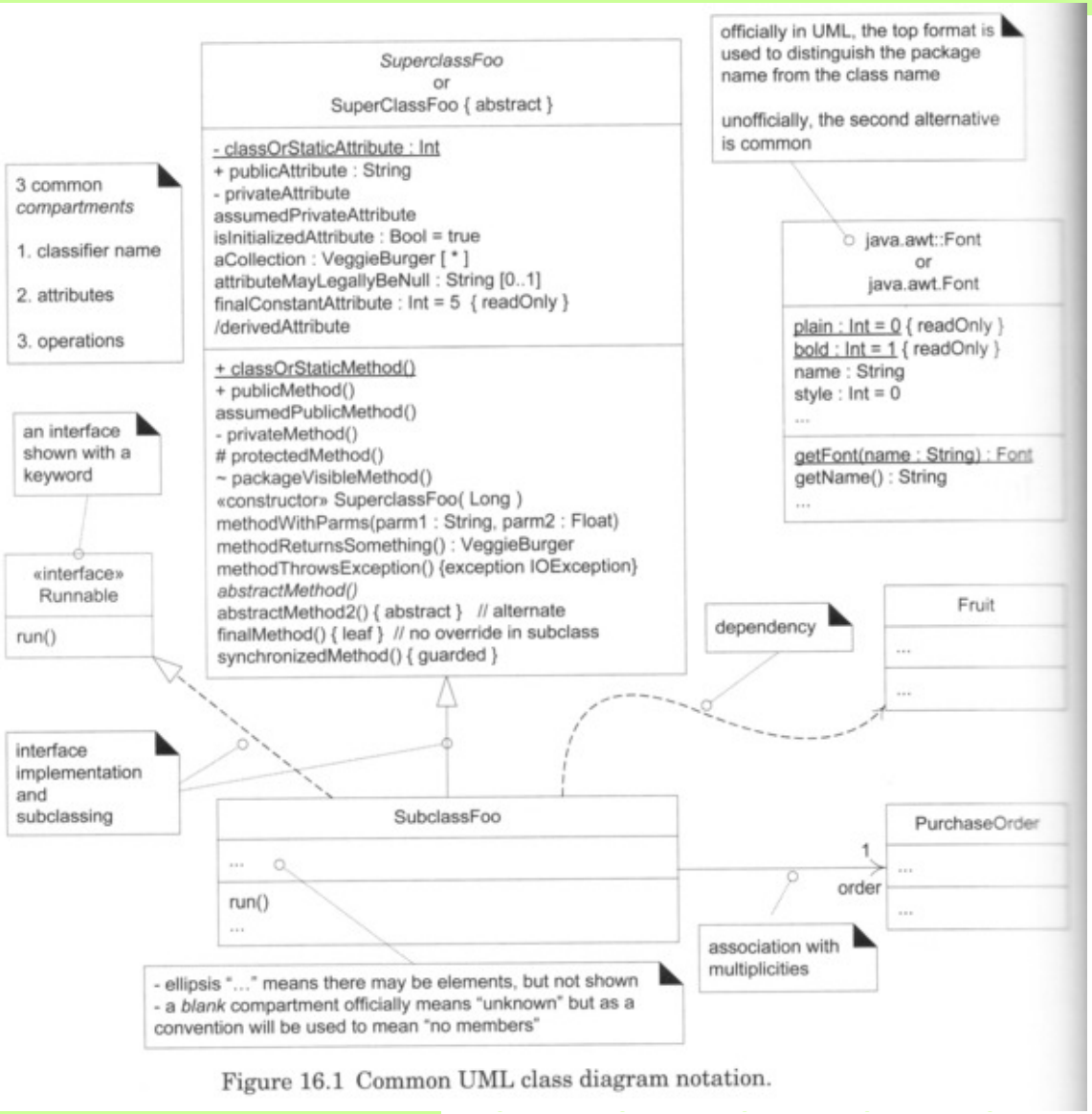


Figure 16.1 Common UML class diagram notation.

Classificador (Classifier)

Um classificador é um elemento do modelo que descreve características comportamentais e estruturais.

Os classificadores são uma generalização de muitos dos elementos da UML, incluindo: classes, interfaces, casos de uso e atores.

Nos diagramas de classes, os dois classificadores mais comuns são as classes e interfaces.

Atributo

Notação textual de um atributo:

visibilidade nome : tipo multiplicidade = default {propriedade}

De acordo com a UML, a sintaxe de qualquer linguagem de programação também pode ser usada na declaração dos atributos.

Se a visibilidade não é apresentada, o atributo é considerado privado.

Visibilidade de Atributos e Operações

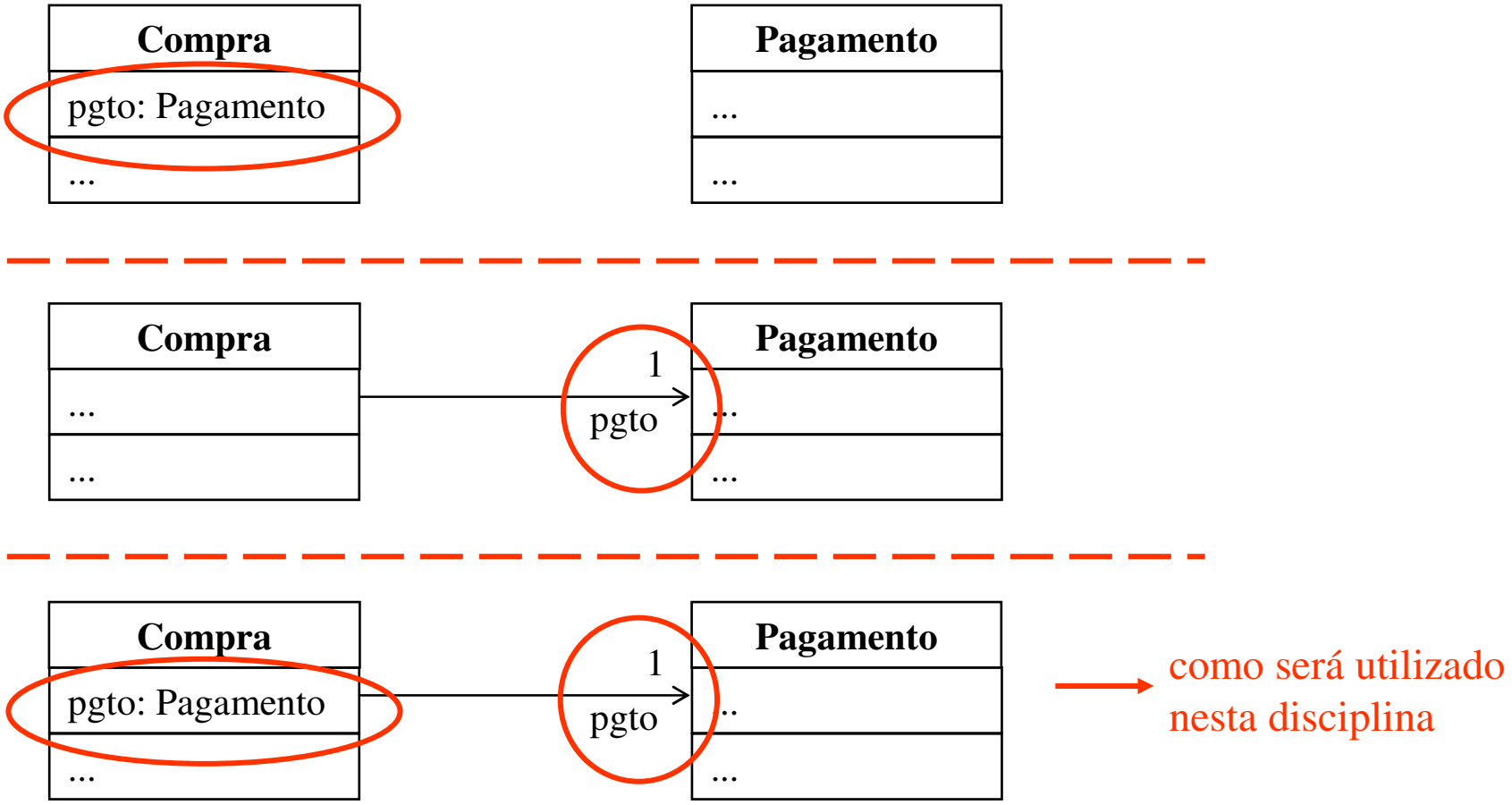
A visibilidade de um atributo ou operação define o nível de acesso que os objetos tem a este atributo ou operação.

A UML define 4 tipos de visibilidade:

+	público	acessível a todos os objetos do sistema
#	protegido	acessível às instâncias da classe em questão e de suas subclasses
-	privado	acessível às instâncias da classe em questão
~	package	acessível às instâncias das classes que estão no mesmo pacote da classe em questão

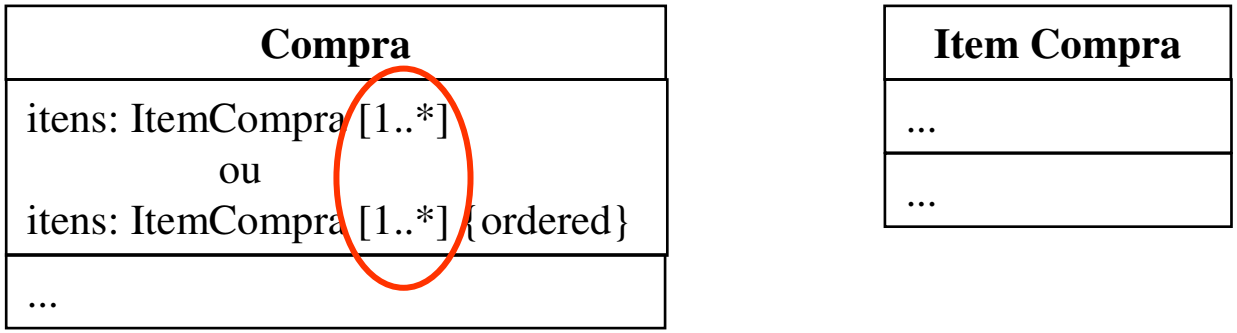
Atributo

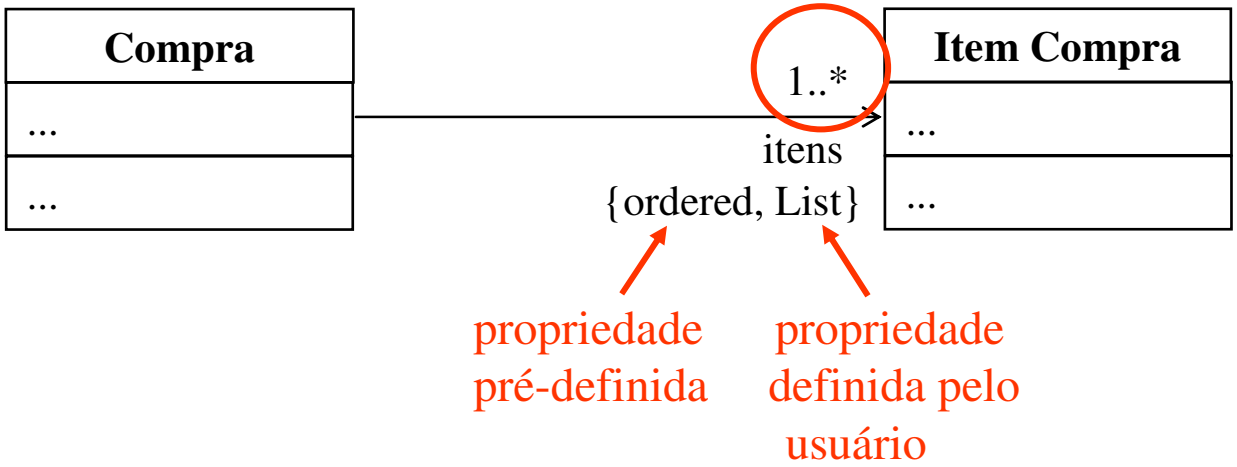
Atributos que representam objetos de outras classes:



Atributo

Atributos que representam coleções de objetos de outras classes:



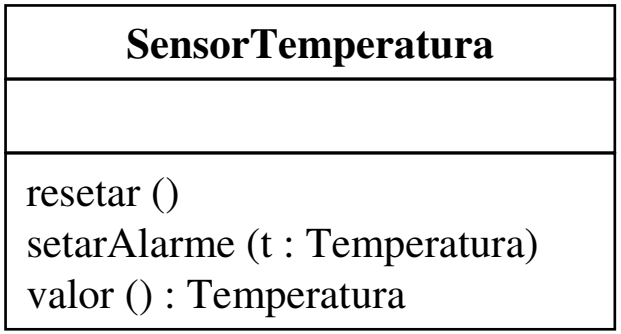


Operação

Uma operação é a implementação de um serviço que pode ser requisitado a qualquer objeto da classe.

Uma classe pode ter zero ou mais operações.

As assinaturas das operações são mostradas no terceiro compartimento do retângulo da classe.



Notação (UML 1.0): nome (lista parâmetros) : tipo-retorno {propriedade}

nome : tipo = valorDefault

A propriedade é uma informação adicional: exceções, abstrato, etc.

Operação

Se a visibilidade não é apresentada, a operação é considerada pública.

Na UML, uma operação não é um método, uma operação é uma declaração.

Na UML, um método é a implementação de uma operação.

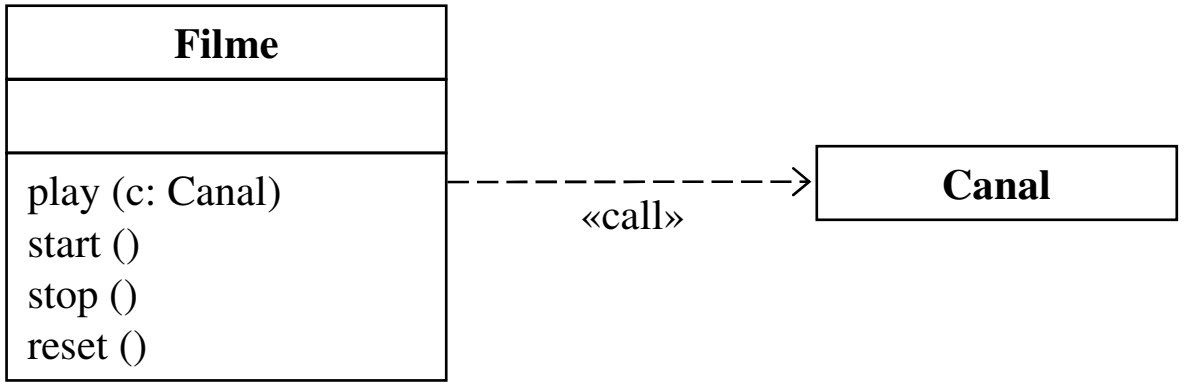
Um método pode ser ilustrado:

- nos diagramas de interação, pelos detalhes e seqüência das mensagens;
- nos diagramas de classes, dentro de um símbolo de nota estereotipada com «method»;
- nos diagramas de atividades.

Operações abstratas, assim como as classes abstratas, são mostradas com {abstract} ou com o nome em itálico.

Dependência

Uma dependência é um relacionamento que indica que um elemento cliente (ex. classe, pacote, caso de uso) tem conhecimento de outro elemento fornecedor e que uma mudança no fornecedor pode afetar o cliente.



Notação de Dependência: é representada como uma linha pontilhada direcionada para o elemento no qual é dependente.

A linha pode ser complementada por um label. Ex: «call», «create».

Dependência

Tipos de dependência entre objetos e classes:

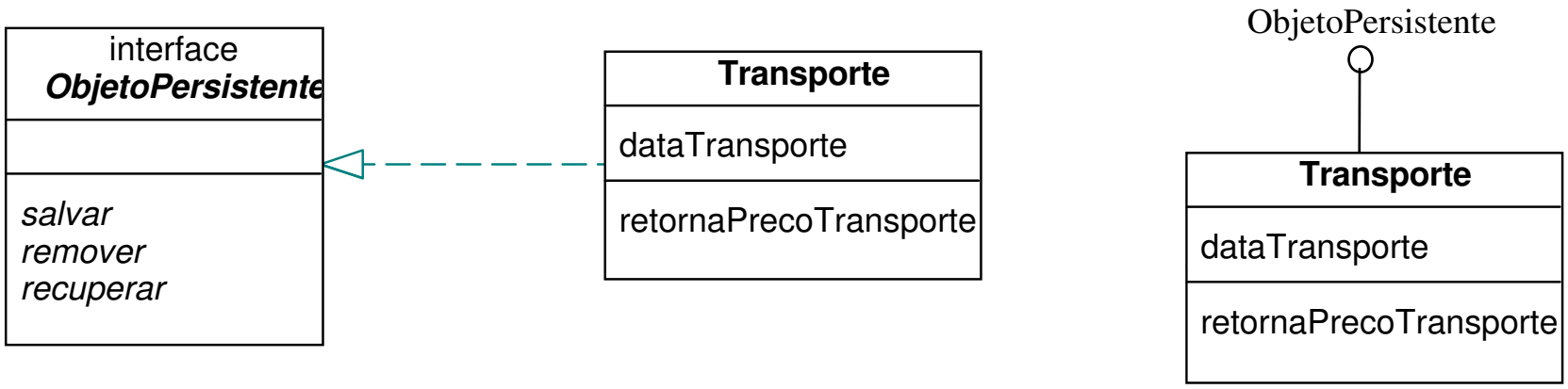
- atributo do tipo do fornecedor
- mensagem enviada a um fornecedor
- parâmetro recebido é do tipo do fornecedor
- o fornecedor é uma superclasse ou interface

➡ Alguns destes tipos de dependência já possuem alguma linha que sugere a dependência.

Interface

Uma interface é uma coleção de assinaturas de operações que definem um conjunto coesivo de comportamentos.

Interfaces são implementadas (ou realizadas) por classes. Para realizar uma interface, uma classe ou componente deve implementar as operações definidas pela interface.



Notação da Interface: igual à da herança, mas com a linha pontilhada. Também pode ser utilizada uma linha com um círculo.

Navegação

A navegação indica que é possível navegar de objetos de um tipo para objetos de outro tipo.

Quando a navegação não é indicada, a associação é bidirecional.

Notação de Navegação: seta junto com a associação.

Definição do Diagrama de Classes de Projeto

Diagrama de Classes de Projeto

Ordem de definição do diagrama de classes de projeto:

➡ O diagrama é definido a partir dos diagramas de interação.

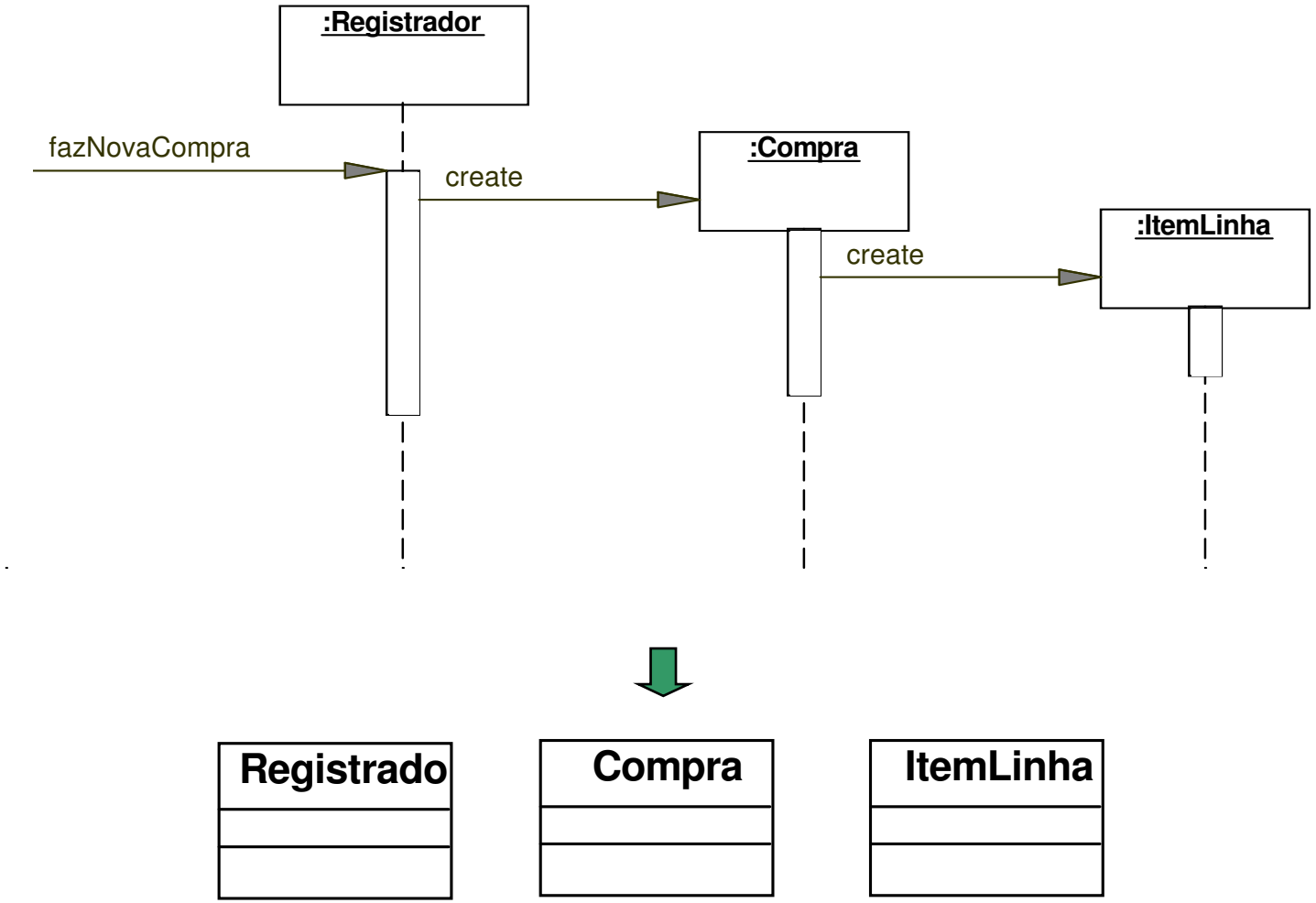
➡ Um diagrama preliminar (classes, atributos e relacionamentos) pode ser definido no início do projeto, a partir do diagrama de classes conceituais.

O diagrama é refinado em paralelo com os diagramas de interação.

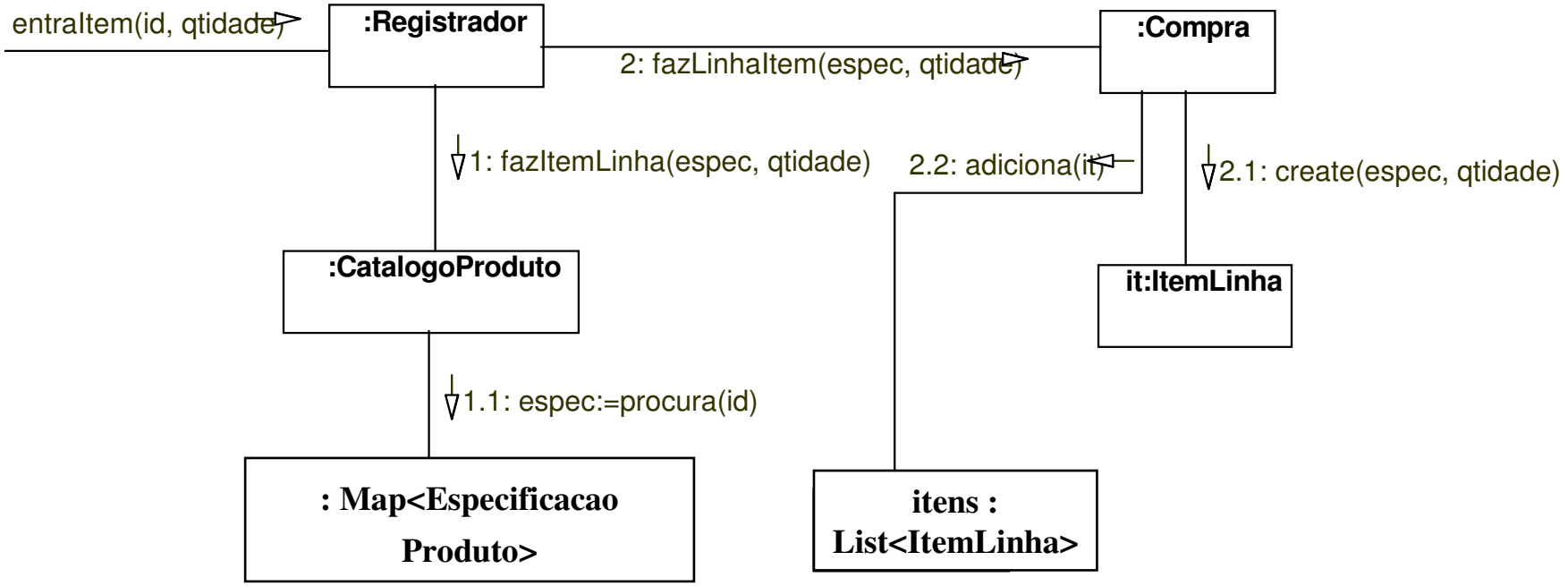
1. Identificação das Classes de Projeto

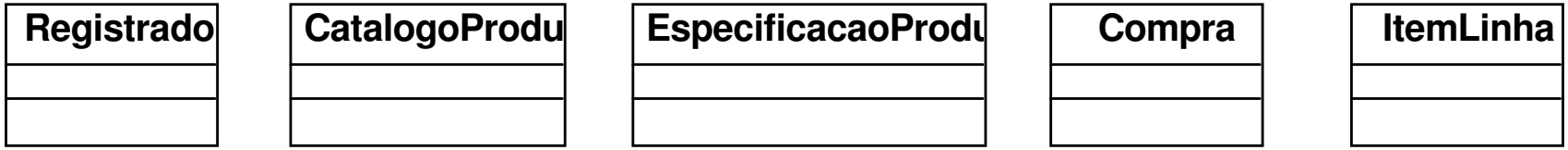
- Identificar as classes que participam da solução de software.
- As classes podem ser encontradas analisando todos os diagramas de interação e listando as que são mencionadas.

Exemplo de Identif. das Classes de Projeto

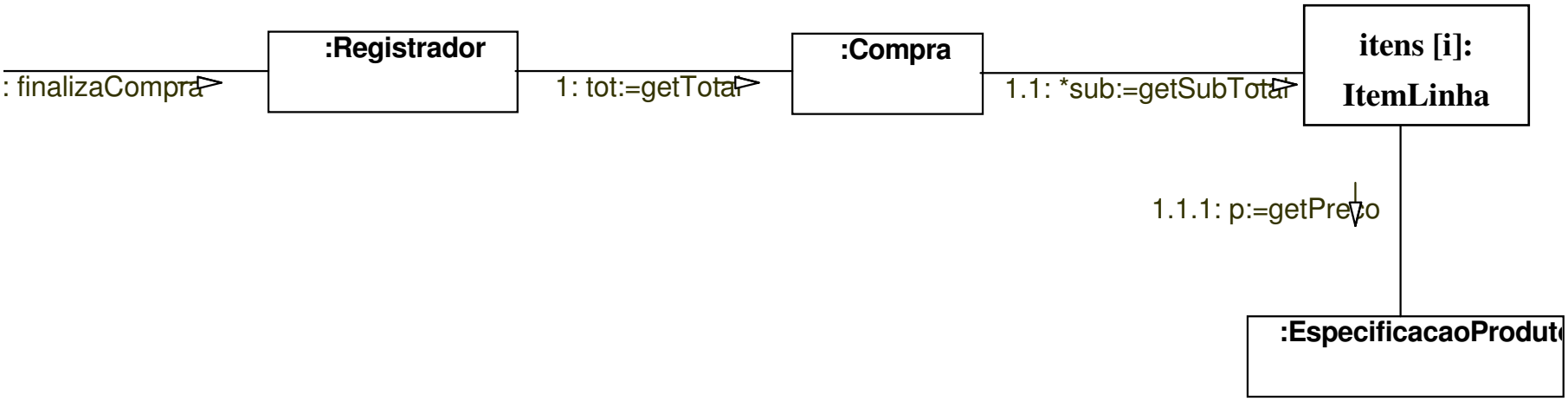


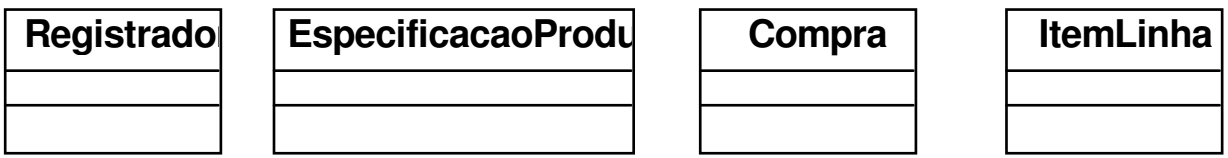
Exemplo de Identif. das Classes de Projeto



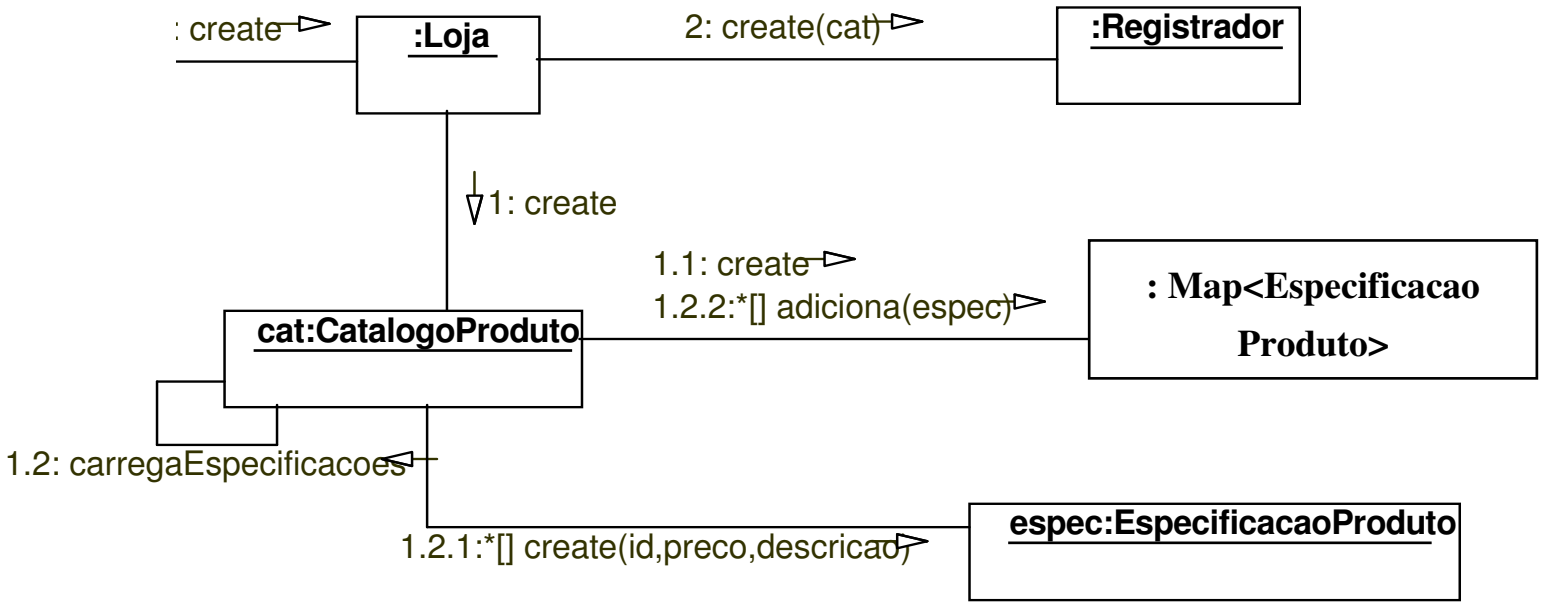


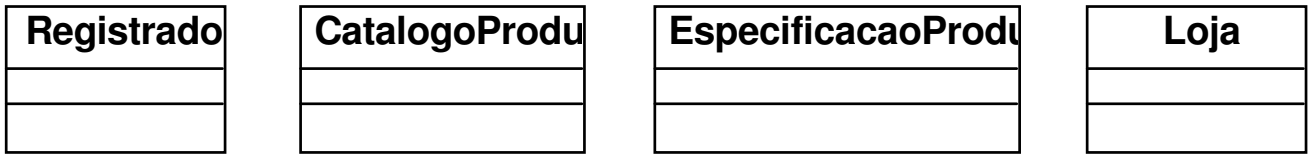
Exemplo de Identif. das Classes de Projeto





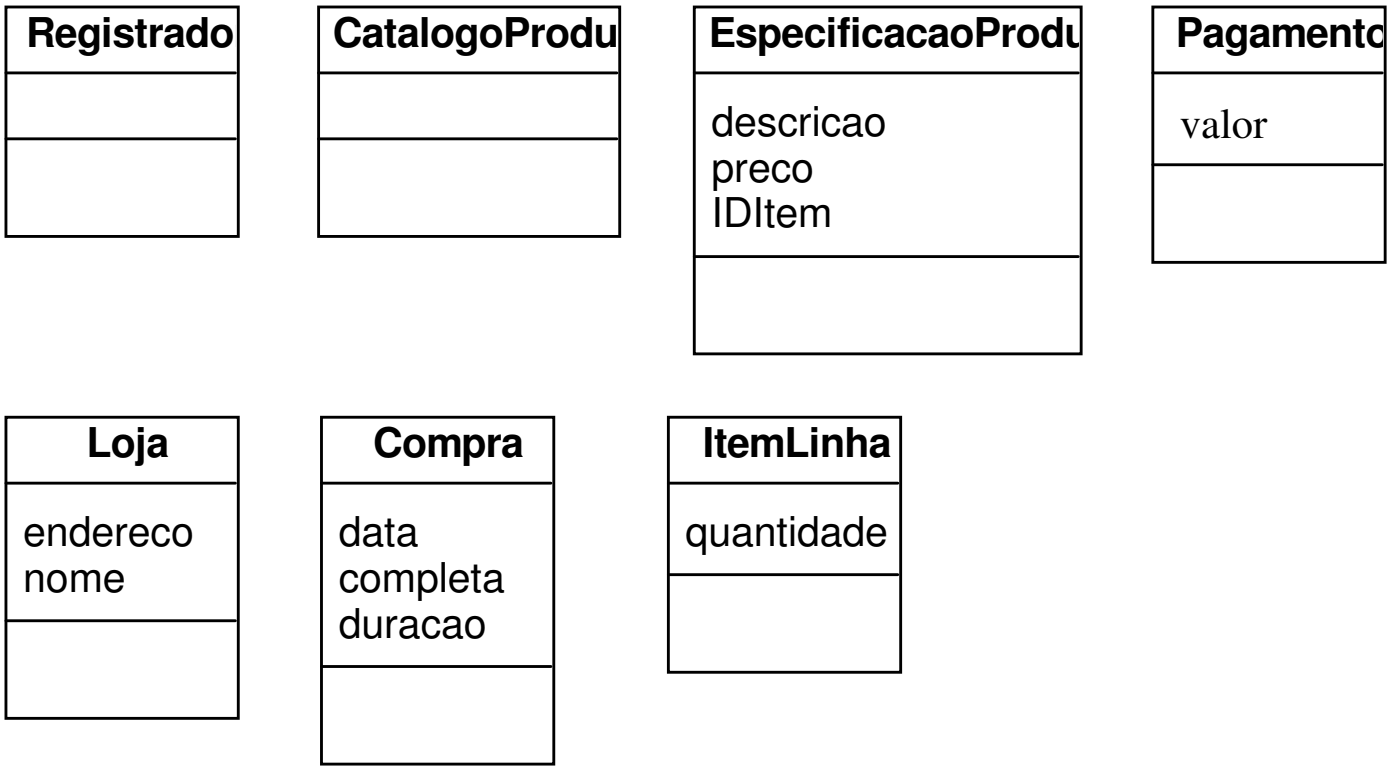
Exemplo de Identif. das Classes de Projeto





Exemplo de Identif. das Classes de Projeto

Diagrama de Classes Preliminar identificado a partir dos diagramas de interação apresentados anteriormente:

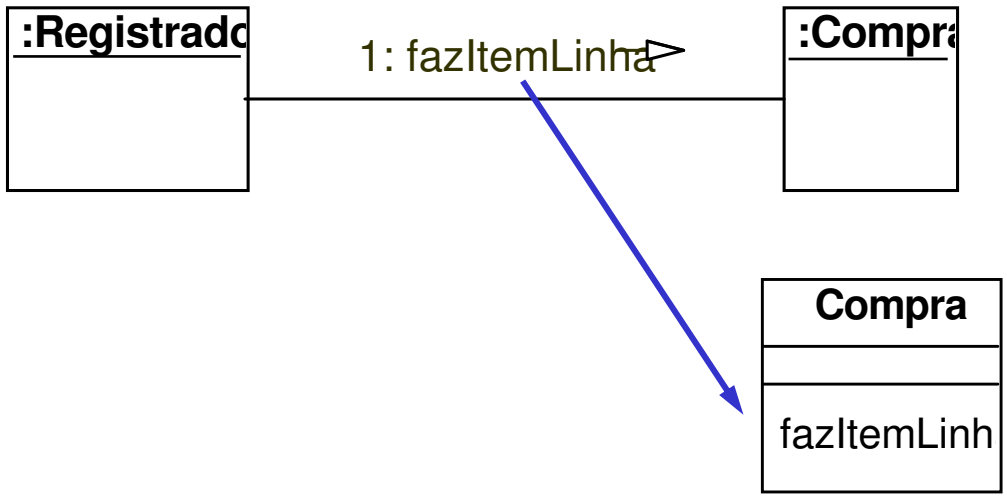


➡ Os atributos identificados durante a análise são incluídos no projeto.

2. Identificação das Operações

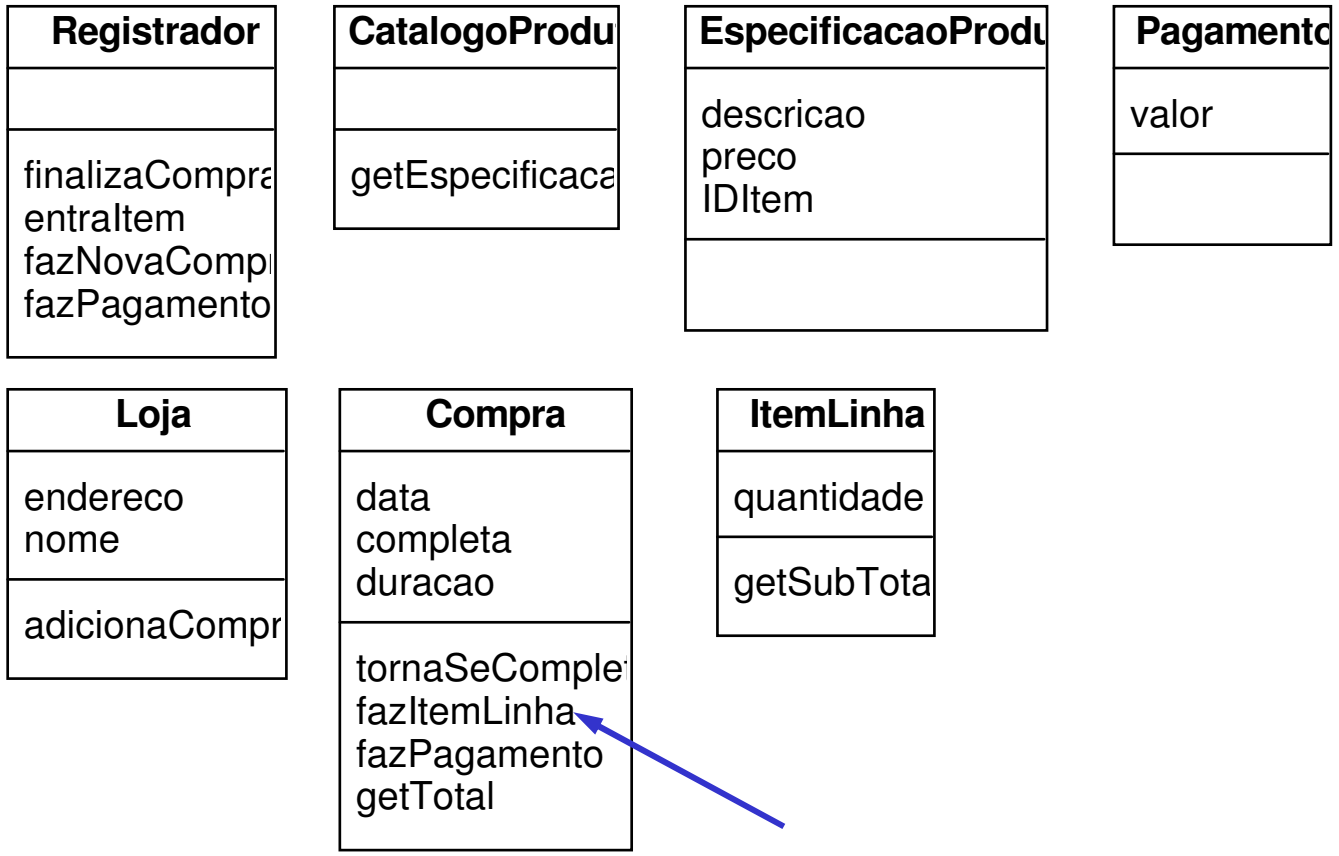
As operações de cada classe são identificadas a partir dos diagramas de interação.

O conjunto de todas as mensagens enviadas aos objetos da classe X nos diagramas de interação indicam operações que a classe X deve definir.



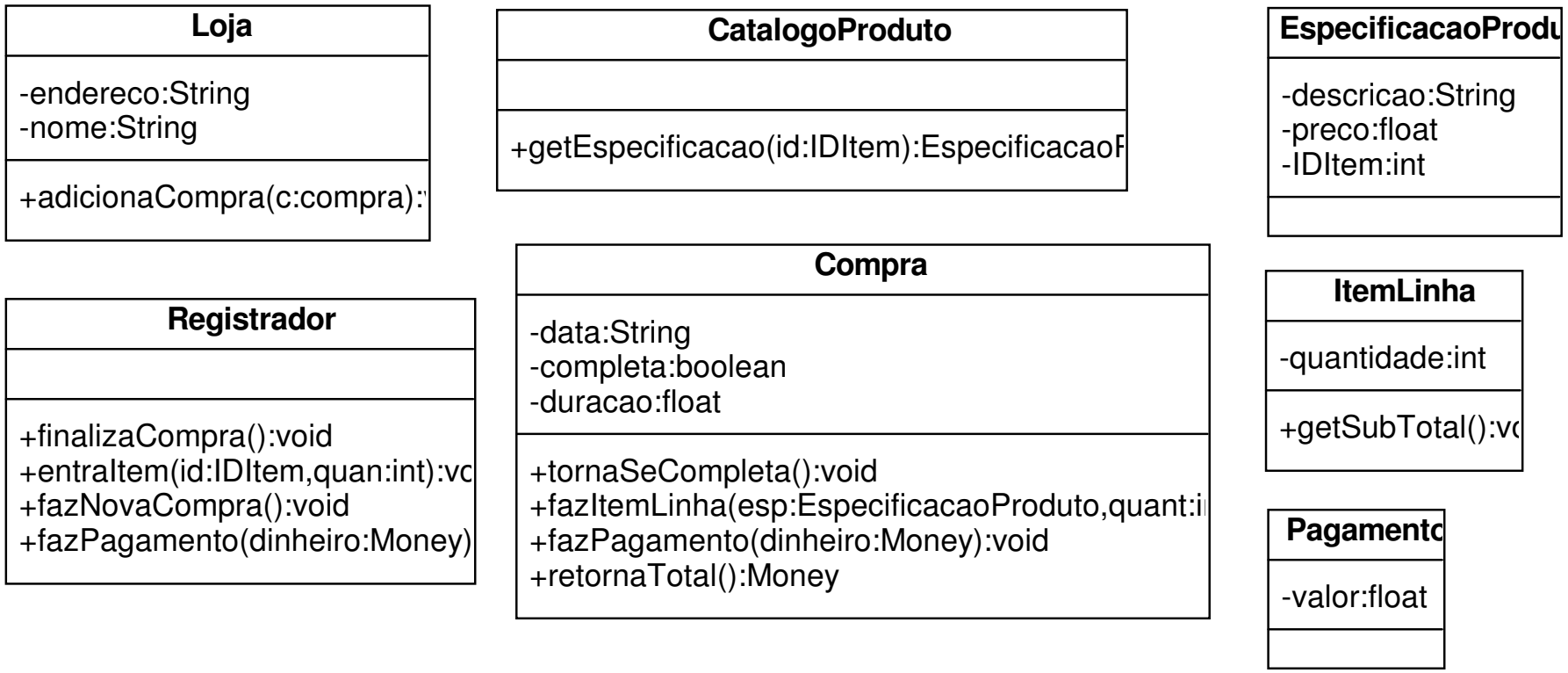
Exemplo de Identificação das Operações

Diagrama de Classes:



Exemplo de Identificação das Operações

Diagrama de Classes com detalhes sobre os atributos e operações



Identificação das Operações

Observações sobre a identificação das operações:

- É comum omitir as operações relacionadas com a criação dos objetos pelo fato da inicialização ser uma atividade bem comum.

- É comum definir operações para acessar e modificar o valor de cada atributo e declarar todos os atributos como privados. Essas operações, geralmente, são excluídas do diagrama de classes para evitar a poluição do diagrama.

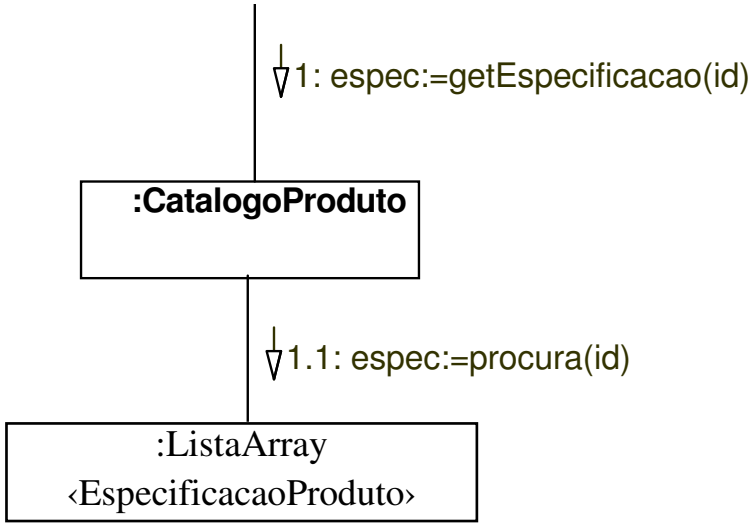
nesta disciplina iremos enumerar todas as operações de criação dos objetos, bem como todas as operações para acessar e modificar o valor de cada atributo

Identificação das Operações

Observações sobre a identificação das operações:

- Mensagens para coleções.

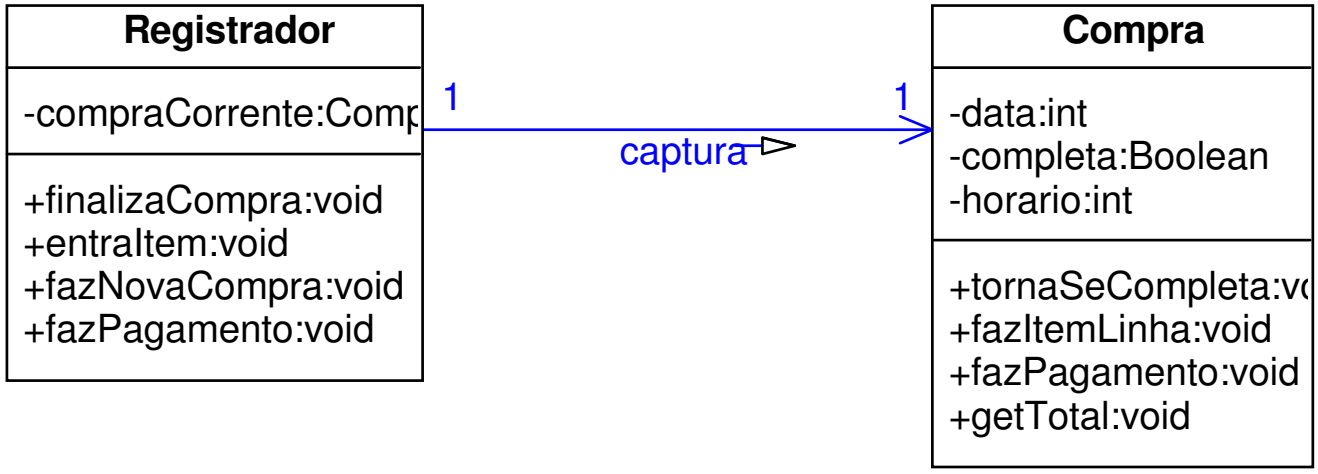
Uma mensagem para uma coleção de objetos é interpretada como uma mensagem para o próprio objeto coleção.



3. Identificação das Associações

Navegabilidade: indica que é possível navegar unidirecionalmente através da associação dos objetos na origem para a classe destino.

A navegabilidade implica visibilidade de atributo da classe origem para a classe destino.



Identificação das Associações

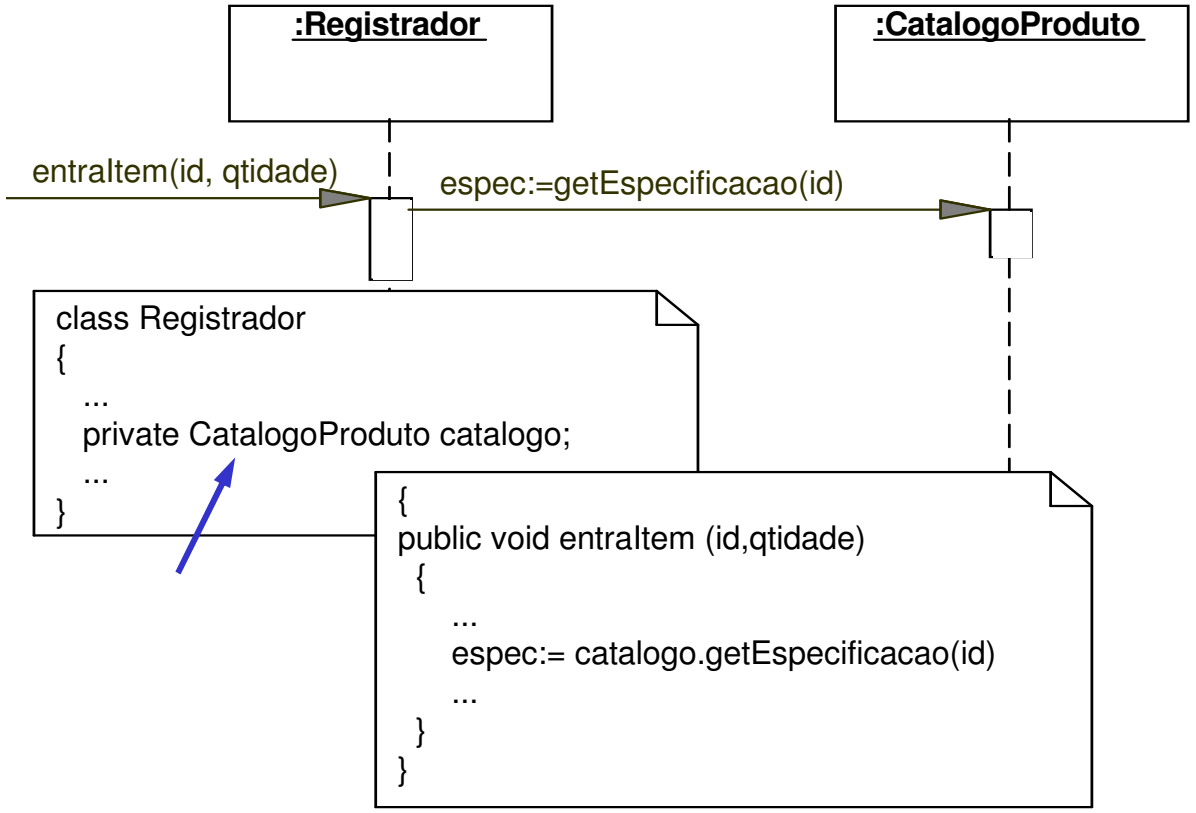
➡ As associações do diagrama de classes de projeto devem ser adornadas com as setas de navegabilidade necessárias.

➡ Quais associações são incluídas no diagrama de classes de projeto? Aquelas que são necessárias para satisfazer a visibilidade e necessidades de acesso indicadas pelos diagramas de interação.

Visibilidade de Atributo

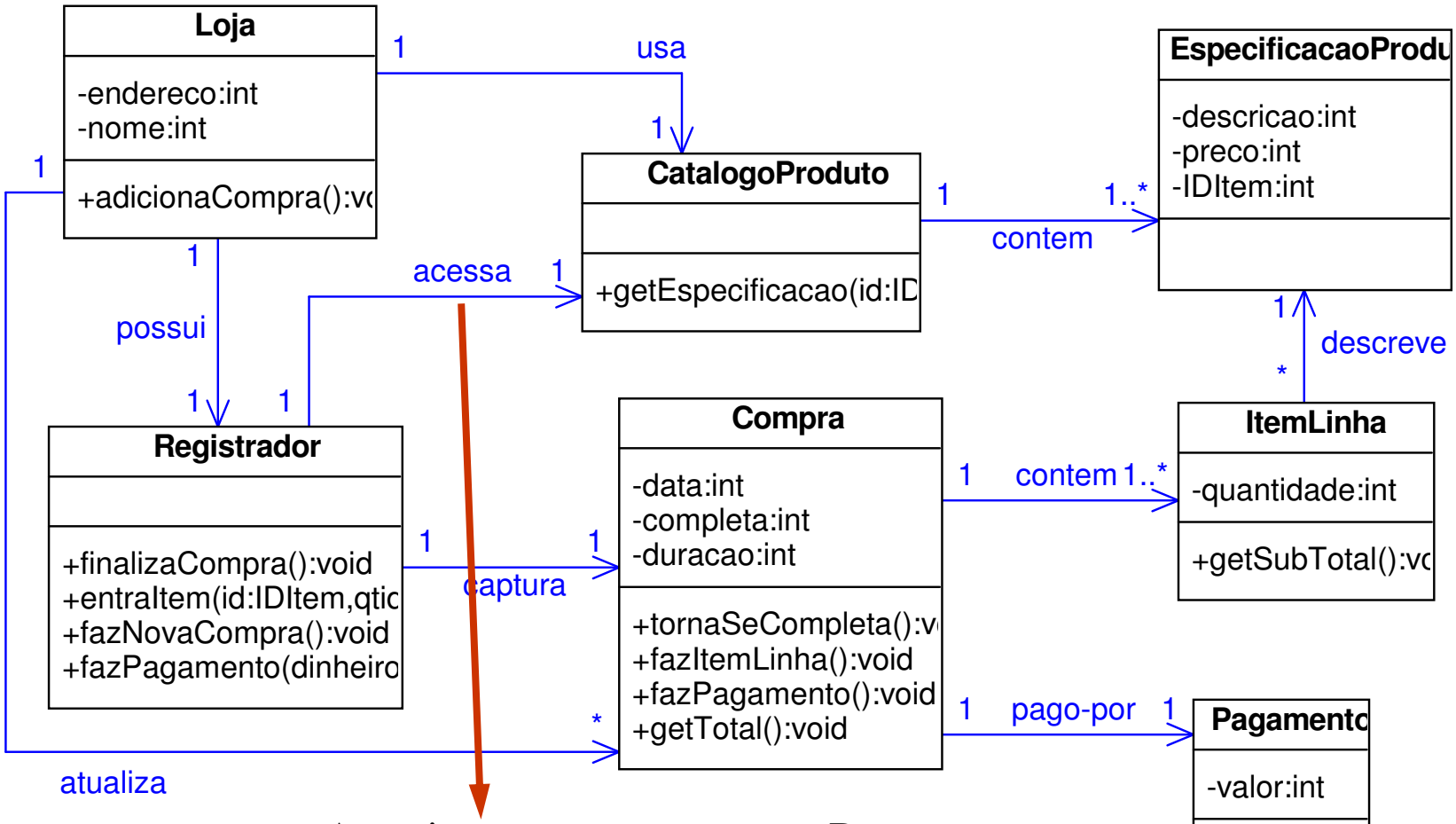
Visibilidade de Atributo de A para B existe quando B é um atributo de A.

É uma visibilidade relativamente permanente porque ela persiste enquanto A e B existem.



Identificação das Associações

Diagrama de Classes de Projeto:

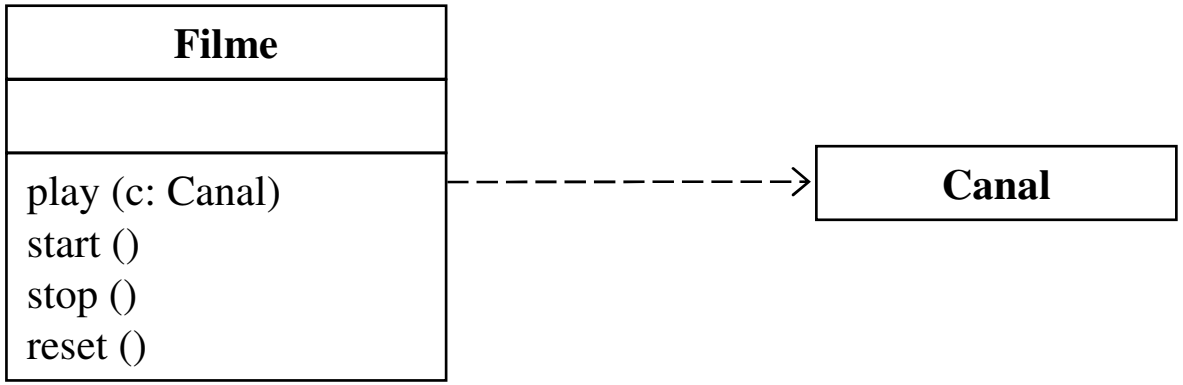


4. Identificação das Dependências

Relacionamento de Dependência: indica que um elemento tem conhecimento sobre outro elemento.

É usado no diagrama de classes de projeto para representar uma visibilidade entre classes que não é do tipo atributo.

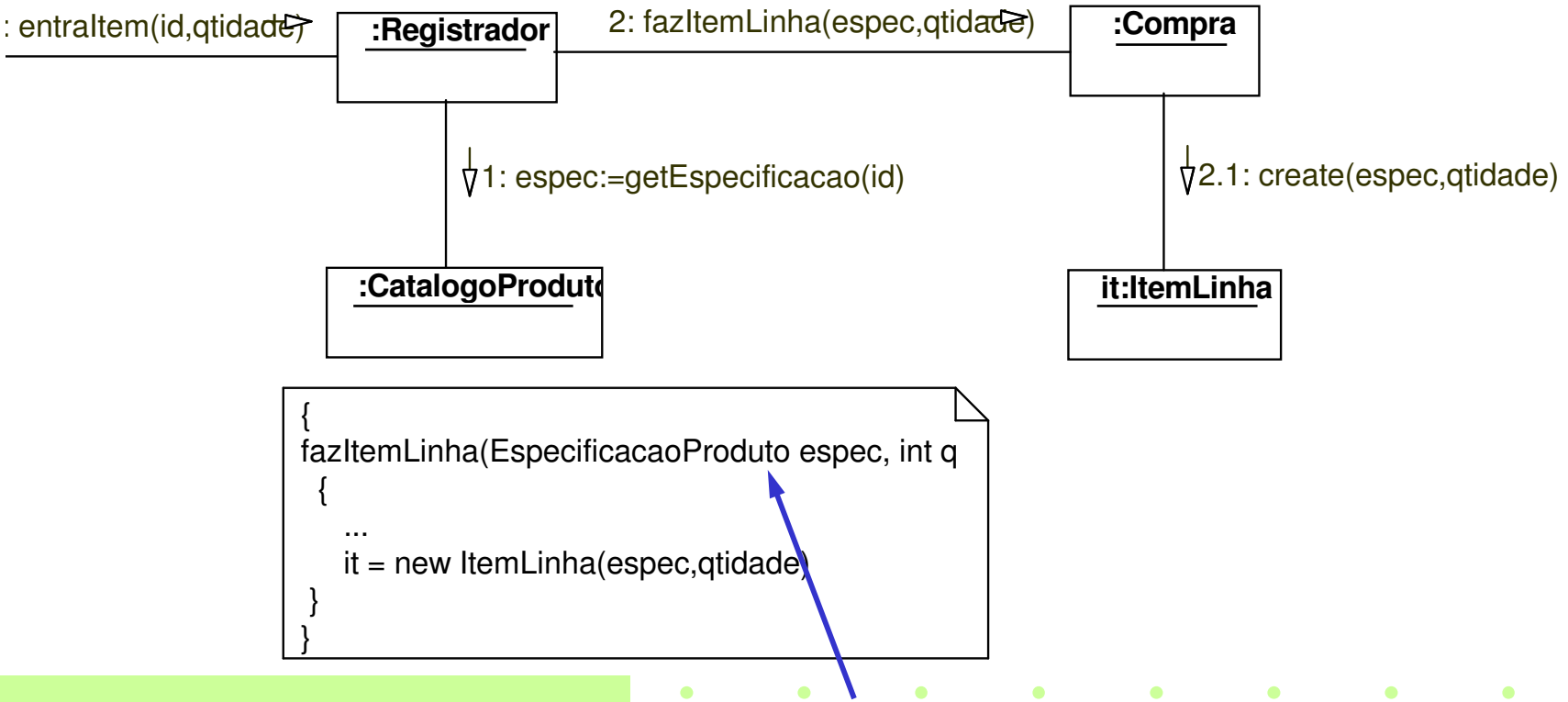
É representado por uma linha pontilhada.



Visibilidade de Parâmetro

Visibilidade de Parâmetro de A para B existe quando B é passado como parâmetro de um método de A.

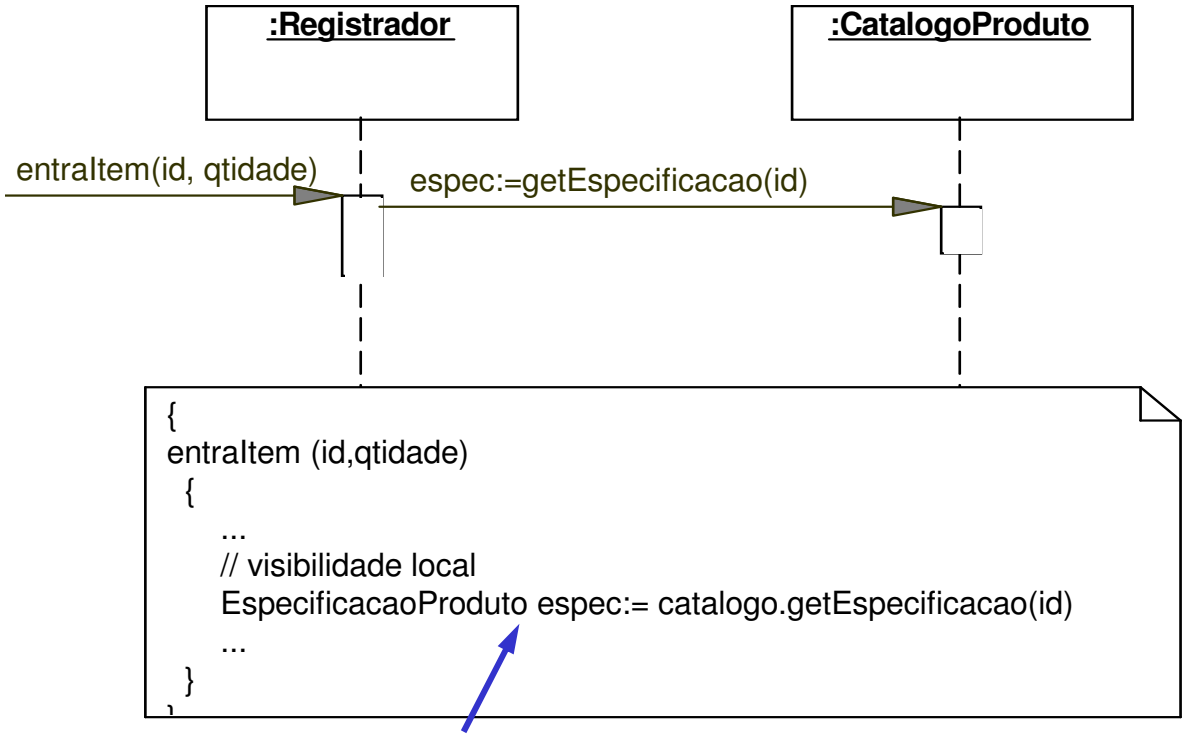
É uma visibilidade relativamente temporária porque ela persiste somente dentro do escopo do método.



Visibilidade Local

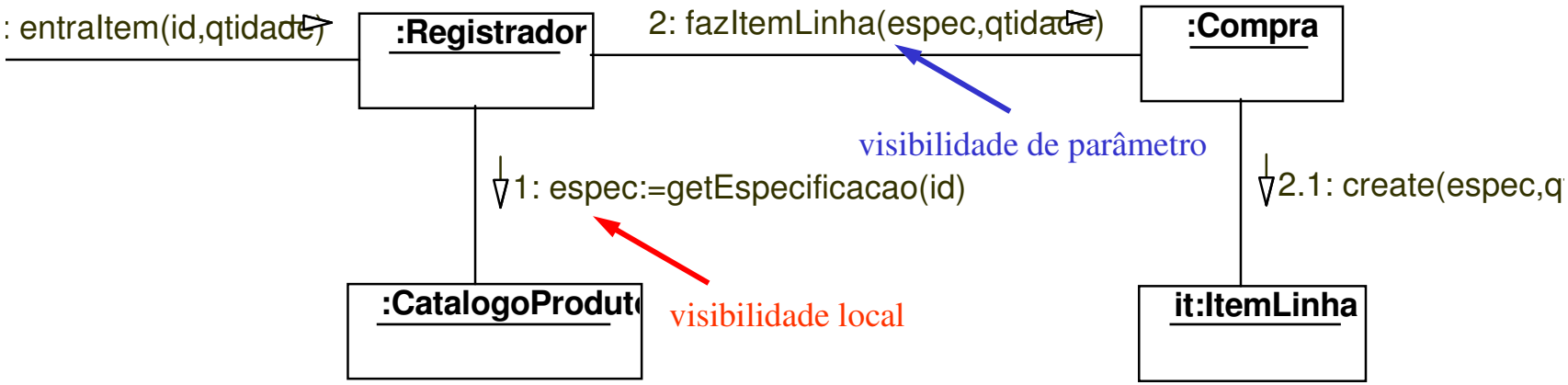
Visibilidade Local de A para B existe quando B é declarado como uma variável local dentro de um método de A.

É uma visibilidade relativamente temporária porque ela persiste somente dentro do escopo do método.



Exemplo de Identificação das Dependências

Considere o seguinte Diagrama de Colaboração:



Exemplo de Identificação das Dependências

Diagrama de Classes de Projeto: