

47006- ANÁLISE E MODELAÇÃO DE SISTEMAS

Desenho por objetos: UML na visualização do código

Ilídio Oliveira

v2020/11/20, TP14a

universidade de aveiro
departamento de eletrónica,
telecomunicações e informática



deti

Objetivos de aprendizagem

Interpretar diagramas de classes (de Código)

Representar construções de código (em Java) nos modelos da UML

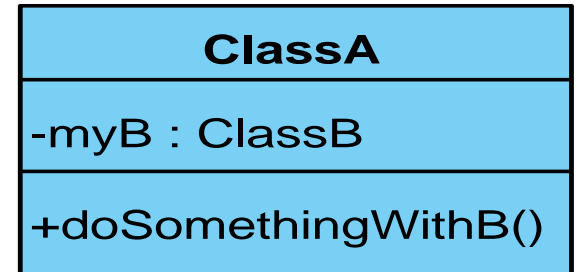
Interpretar diagramas de sequência que modelam colaboração entre objetos)

Visualização do código Java com a UML

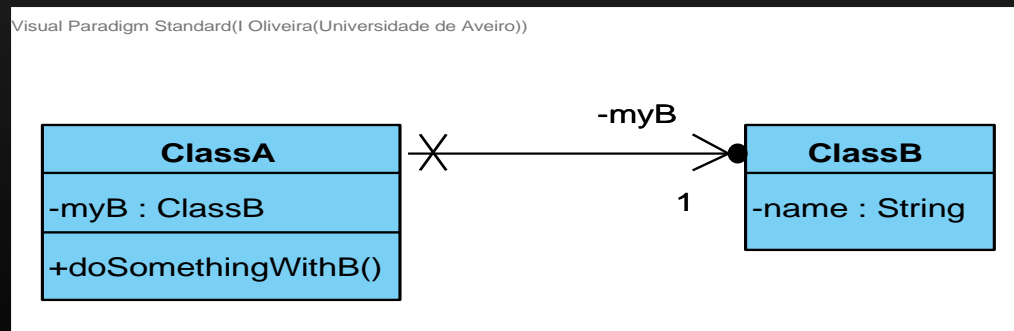
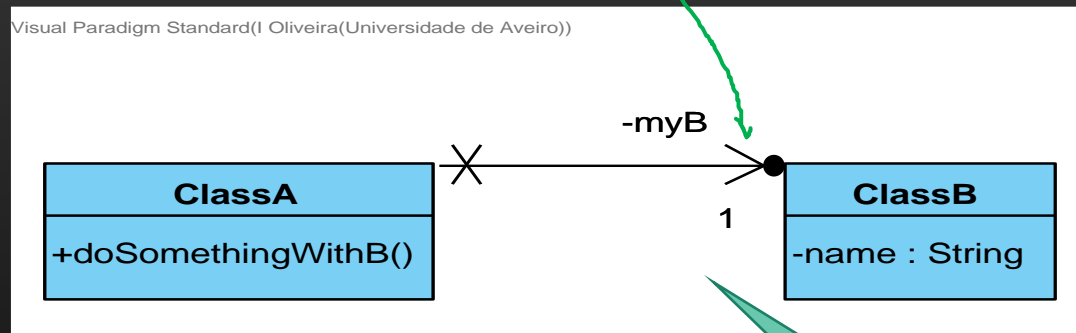
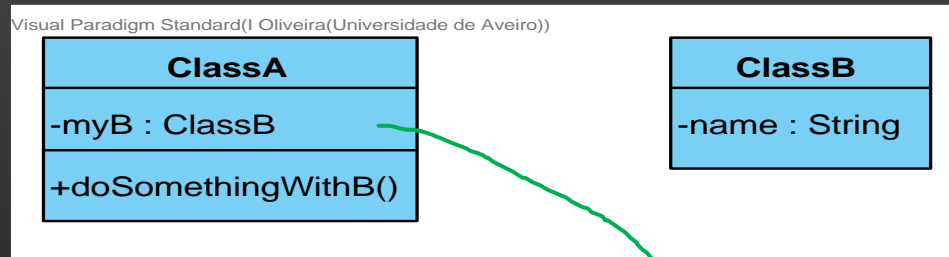
Visualização de Código com classes

```
public class ClassA {  
  
    private ClassB myB;  
  
    public void doSomethingWithB() {  
        // todo;  
    }  
}
```

Visual Paradigm Standard (I Oliveira) (Universidade



Visualização do código com classes



Modelos semanticamente equivalentes.
Mostrar os atributos como associações evidencia os relacionamentos.

```

public class ClientsPortfolio {

    private ArrayList<Client> myClientsList;

    public ClientsPortfolio() {
        myClientsList = new ArrayList<>();
    }

    public void addClient(Client newClient) {
        this.myClientsList.add(newClient);
    }

    public int countClients() {
        return this.myClientsList.size();
    }

}

```

Classe

Atributo (neste caso, é uma lista de objetos do tipo Client)

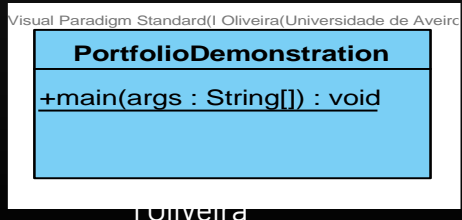
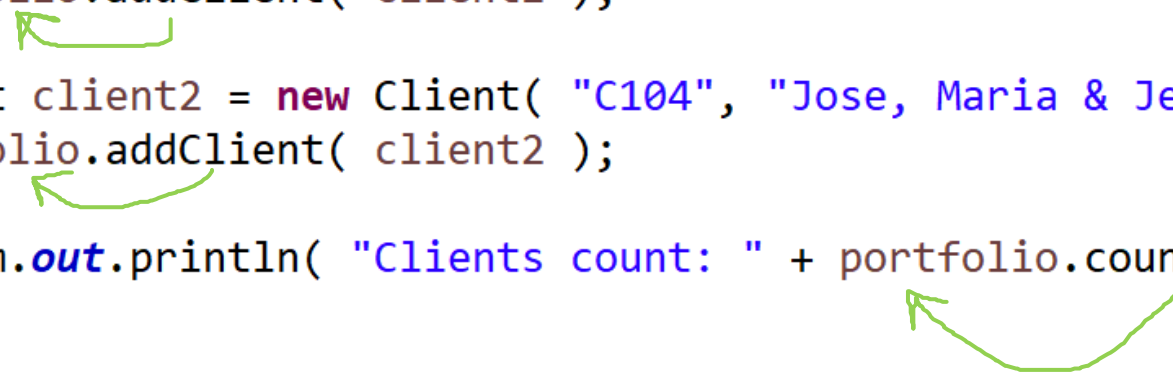
Operações (que podem requerer parâmetros e produzir um valor de retorno)

Visual Paradigm Standard (I Oliveira (Universidade de Aveiro))

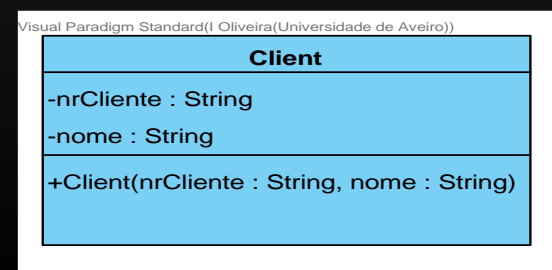


Objetos enviam mensagens

```
public class PortfolioDemonstration {  
    public static void main(String[] args) {  
        // obter um novo objeto da classe ClientsPortfolio  
        ClientsPortfolio portfolio = new ClientsPortfolio();  
  
        // obter um novo objeto da classe Cliente e adicioná-lo ao portfolio  
        Client client1= new Client( "C103", "Logistica Tartaruga");  
        portfolio.addClient( client1 );  
  
        Client client2 = new Client( "C104", "Jose, Maria & Jesus Lda");  
        portfolio.addClient( client2 );  
  
        System.out.println( "Clients count: " + portfolio.countClients() );  
    }  
}
```

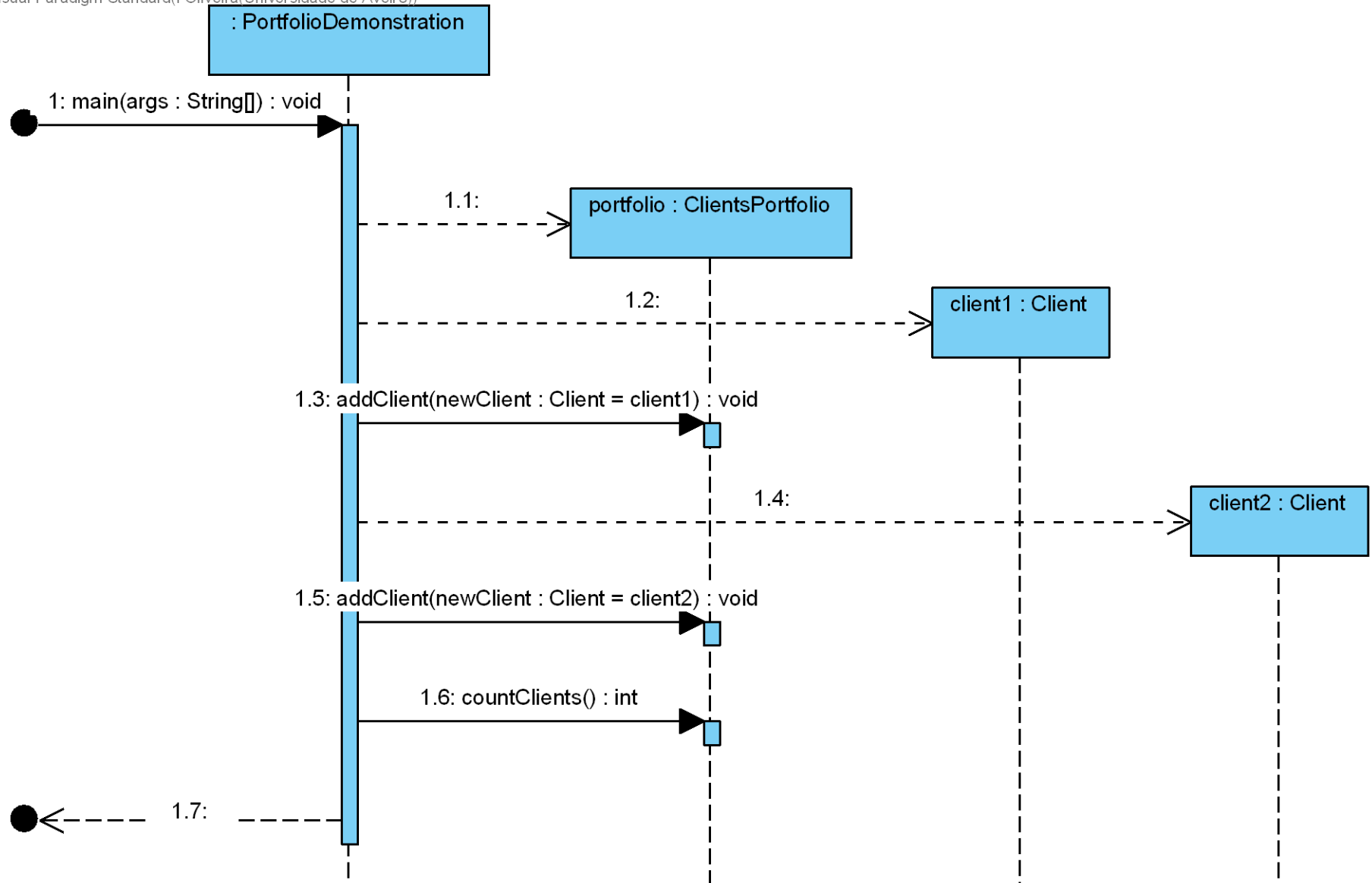


I Oliveira



...que podem ser vistas num modelo dinâmico

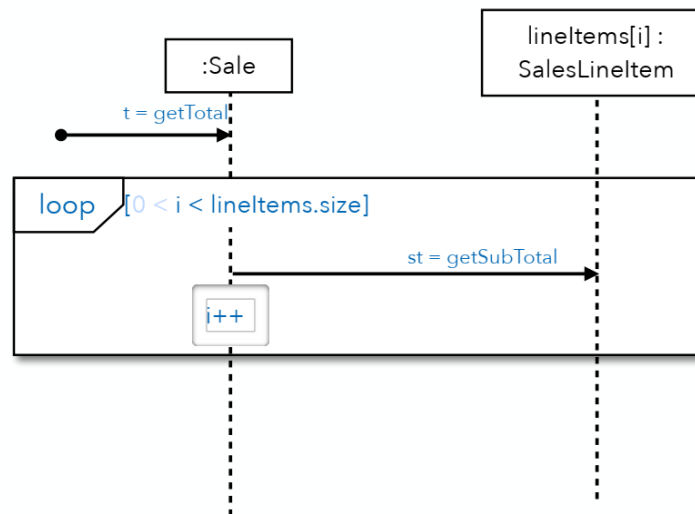
Visual Paradigm Standard (I Oliveira (Universidade de Aveiro))



Alguns exemplos adicionais

Use a **UML loop frame** to iterate over a collection.

UML Sequence Diagrams | 28



Modeling task: Calculate the total of a sale by summing up the sub totals for each sales line item.

Ver pag. 27 a 34

http://stg-tud.github.io/eise/WS18-SE-08-Modeling-dynamic_Part.pdf

UML para “visualizar” o código: estrutura e interação

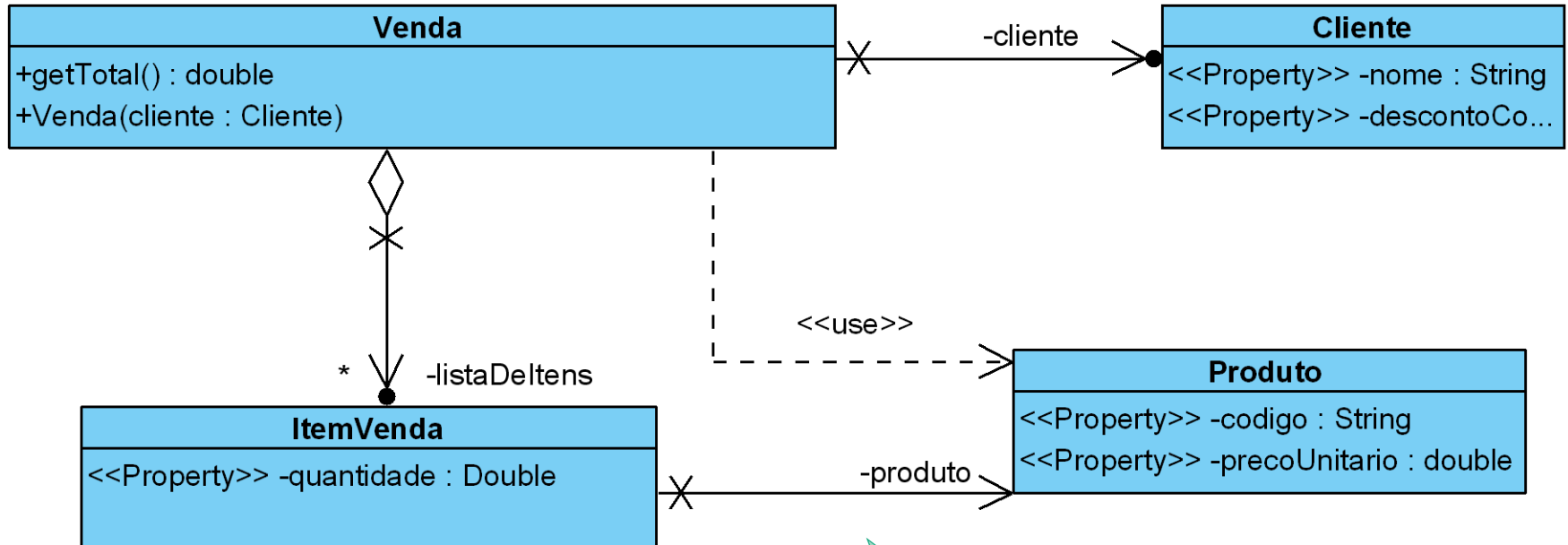
0 objetos Java colaboram para realizar objetivos

```
public class Encomenda {  
  
    private Cliente cliente;  
    private Collection<ItemDaEncomenda> listaDeItens = new ArrayList<>();  
  
    public double getTotal() {  
  
        double total= 0.0;  
  
        Produto produto;  
        for (ItemDaEncomenda item : listaDeItens) {  
            produto = item.getProduto();  
            total += produto.getPrecoUnitario() * item.getQuantidade();  
        }  
        total = total * (1- this.cliente.getDescontoComercial());  
        return total;  
    }  
  
    public Encomenda(Cliente cliente) {  
        super();  
        this.cliente = cliente;  
    }  
  
}
```

Quais são as classes envolvidas?
O que podemos descobrir sobre o seu
"esqueleto" (operações e assinaturas,
atributos)?

Vista estrutural (definição das classes)

Visual Paradigm Standard (I Oliveira (Universidade de Aveiro))

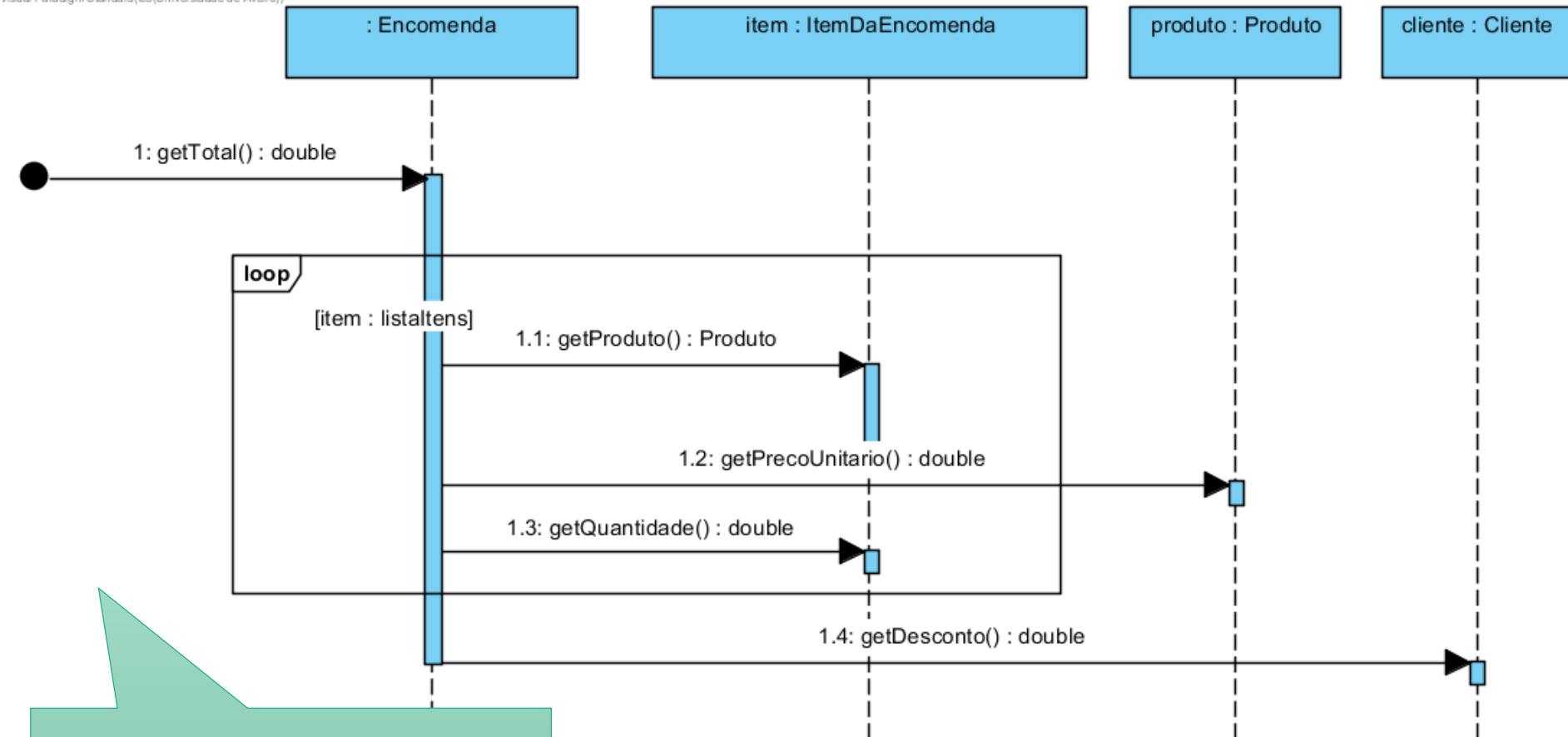


Os atributos que implicam um relacionamento entre classes estão representados como associações.

O esteriótipo `<<Property>>` marca atributos que têm `getter` e `setter`

Vista dinâmica (interações entre objetos)

Visual Paradigm Standard (ico(Universidade de Aveiro))



Qual a colaboração entre objetos necessária para implementar `Encomenda#getTotal()`?

Referências

Core readings	Suggested readings
<ul style="list-style-type: none">• [Dennis15] – Chap. 8	<ul style="list-style-type: none">• [Larman04] – Chap. 17 and 18• Slides by M. Eichberg : SSD and OO-Design