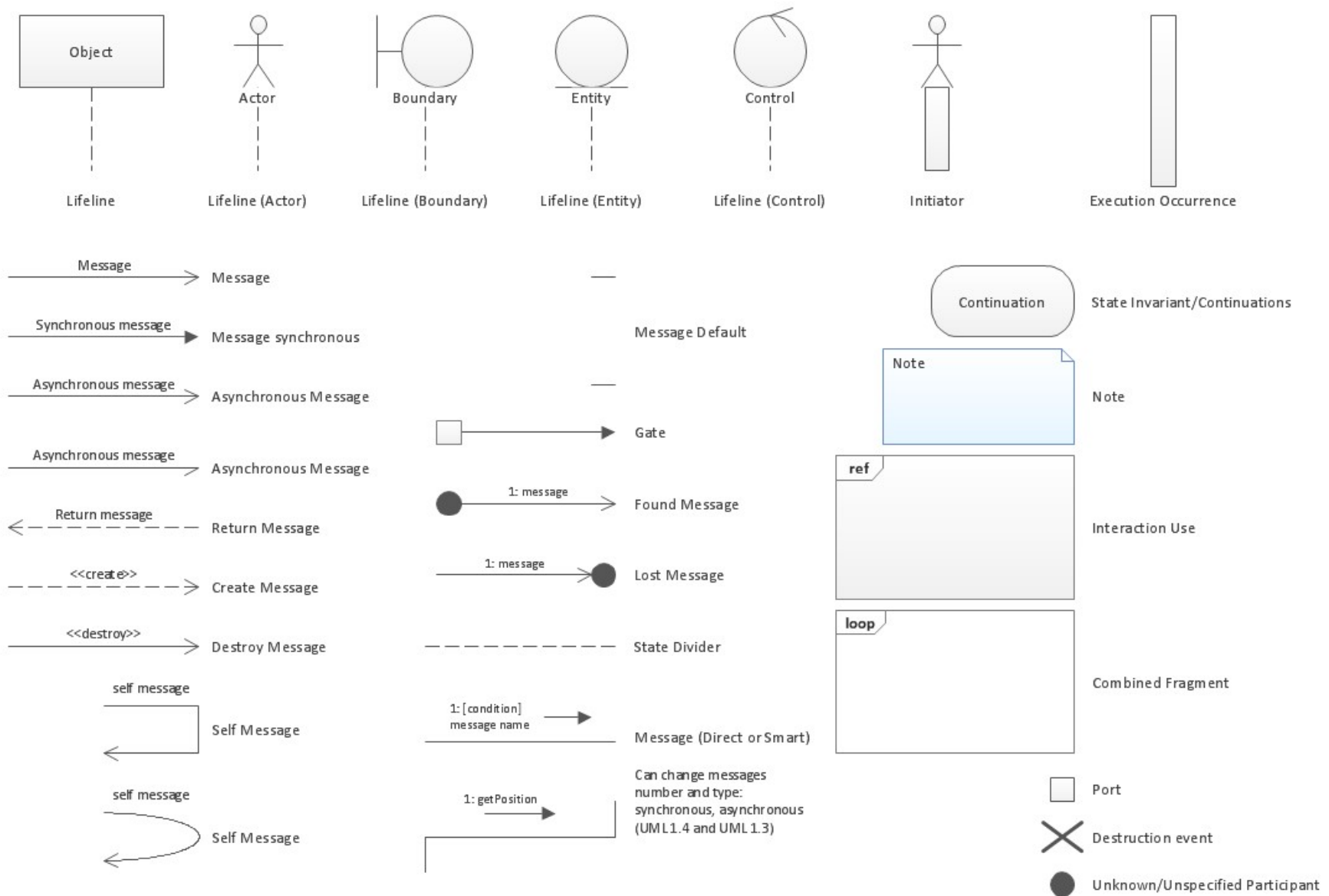


## UML Sequence Diagram




---

*UML*  
*Diagrama de Sequência*

---

## Tipos de Diagramas do UML

UML 2 possui 14 tipos de diagramas divididos em duas categorias:


- **Diagramas Estruturais**  
Enfatizam os elementos que precisam estar presentes no sistema modelado.
- **Diagramas Comportamentais**   
Enfatizam o que precisa acontecer no sistema modelado.

## Tipos de Diagramas do UML

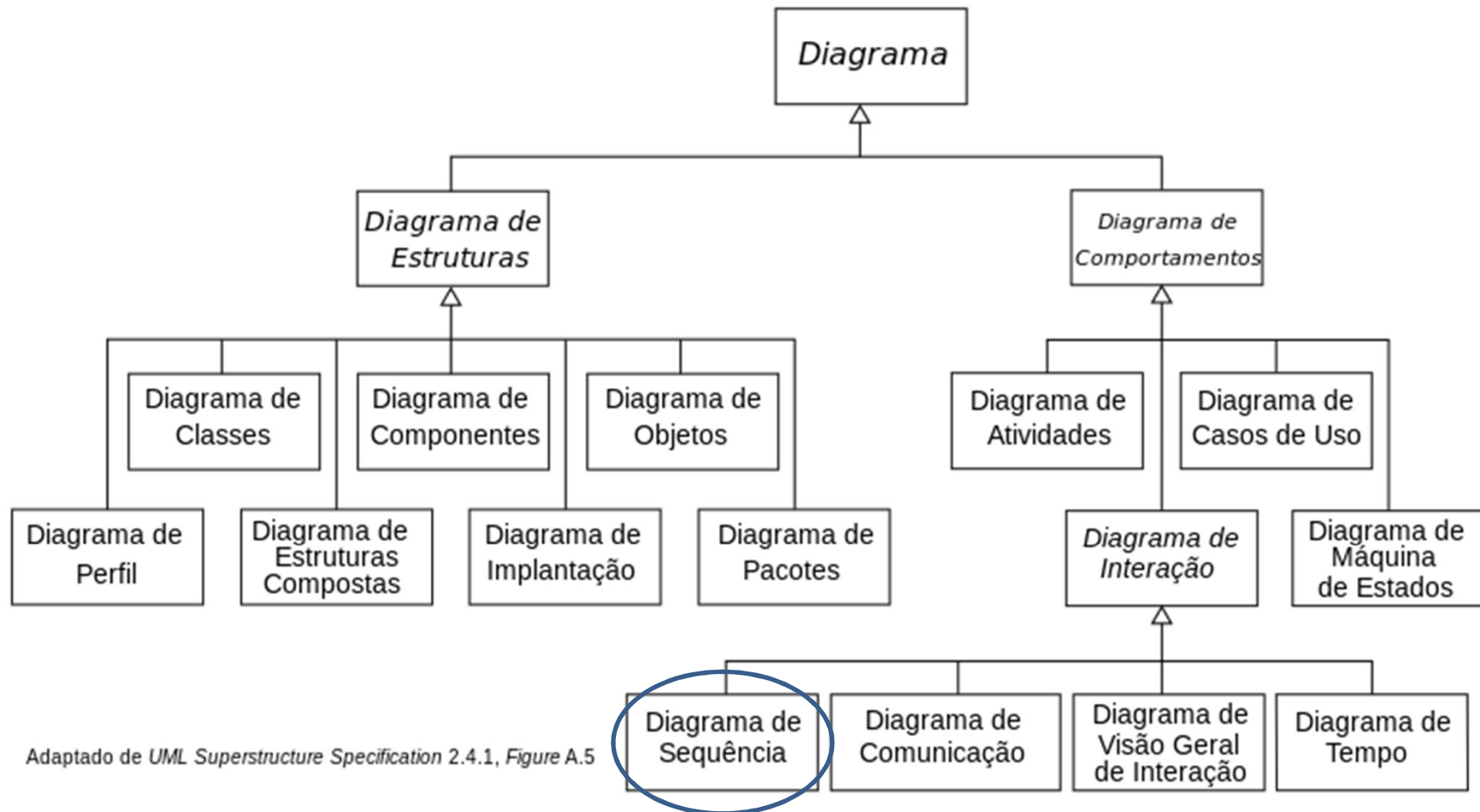
UML 2 possui 14 tipos de diagramas divididos em duas categorias:

- **Diagramas Comportamentais**

Enfatizam o que precisa acontecer no sistema modelado.

- Diagrama de Caso de Uso
- Diagrama de Transição de Estados (ou de Estados)
- Diagrama de Atividade
- Diagrama de Sequência 
- Diagrama Visão Geral de Interação (ou de Interação)
- Diagrama de Colaboração (ou Comunicação)
- Diagrama de Tempo (ou Temporal)

## Tipos de Diagramas do UML





## ***UML: Diagrama de Sequência***

O que são Diagramas de Sequência?

Diagramas de Sequência representam a sequência de mensagens passadas entre objetos num programa de computador.



## ***UML: Diagrama de Sequência***

Qual a importância do Diagrama de Sequência?

Como um projeto pode ter uma grande quantidade de métodos em classes diferentes, pode ser difícil determinar a sequência global do comportamento.



### UML: Diagrama de Sequência

- Descreve a maneira como os grupos de objetos colaboram em algum comportamento ao longo do tempo.
- Registra o **comportamento de um único caso de uso** e exibe os objetos e as mensagens passadas entre esses objetos no caso de uso.
- Dá ênfase a **ordenação temporal** em que as mensagens são trocadas entre os objetos de um sistema.
- **Mensagens** são os serviços solicitados de um objeto a outro, e as respostas desenvolvidas para as solicitações



### UML: Diagrama de Sequência

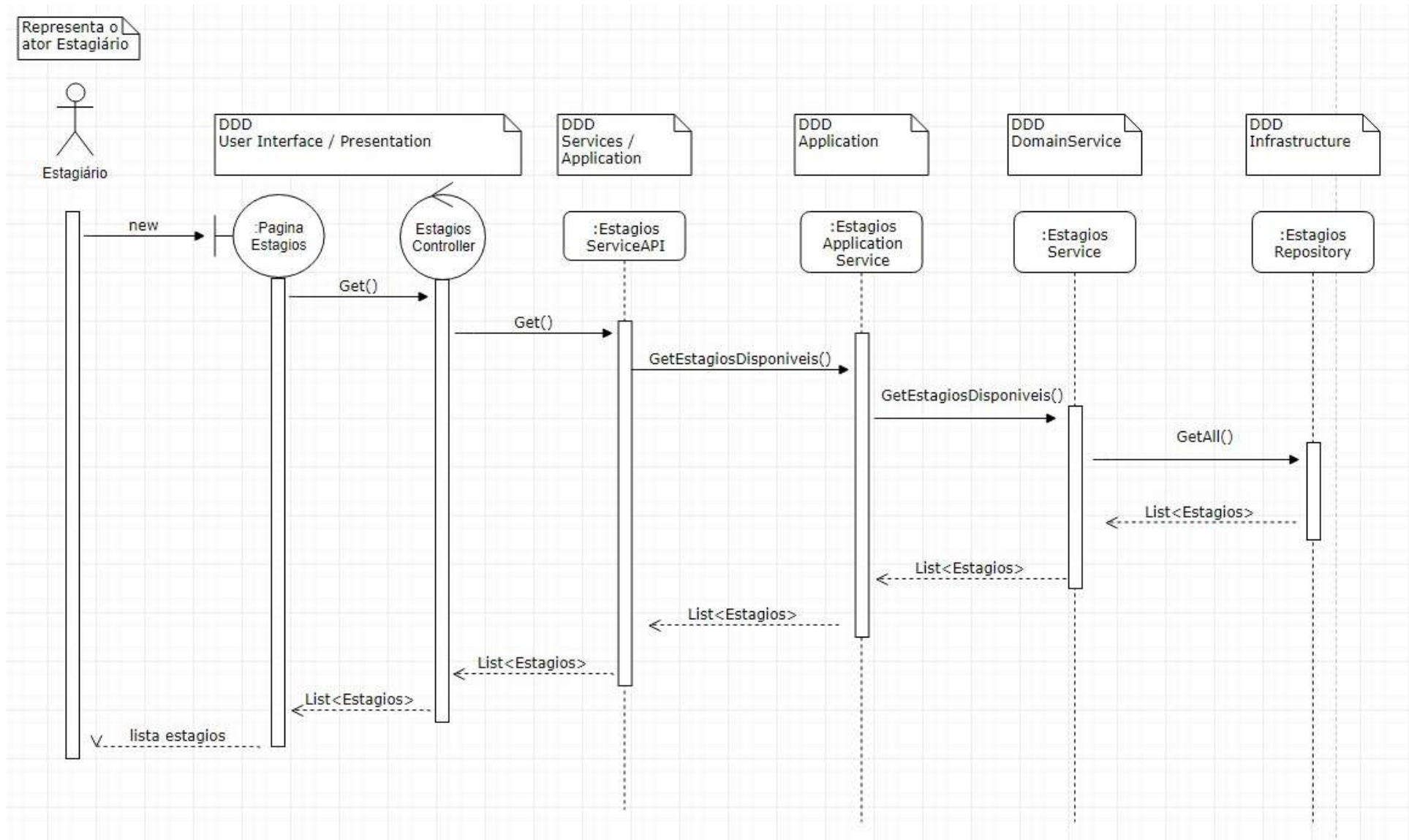
- **Atores**  
São entidades externas que interagem com o sistema e que solicitam serviços.
- **Objetos**  
Representam as instâncias das classes representadas no processo.
- **Gate**  
Indica um ponto em que a mensagem pode ser transmitida para dentro ou para fora.
- **Fragmento de Interação**  
Alt (Alternativa), Opt (Opcional), Break (Parar), Loop (Repetição) etc.
- **Linha de vida**  
Dimensão vertical: Cabeça e cauda.



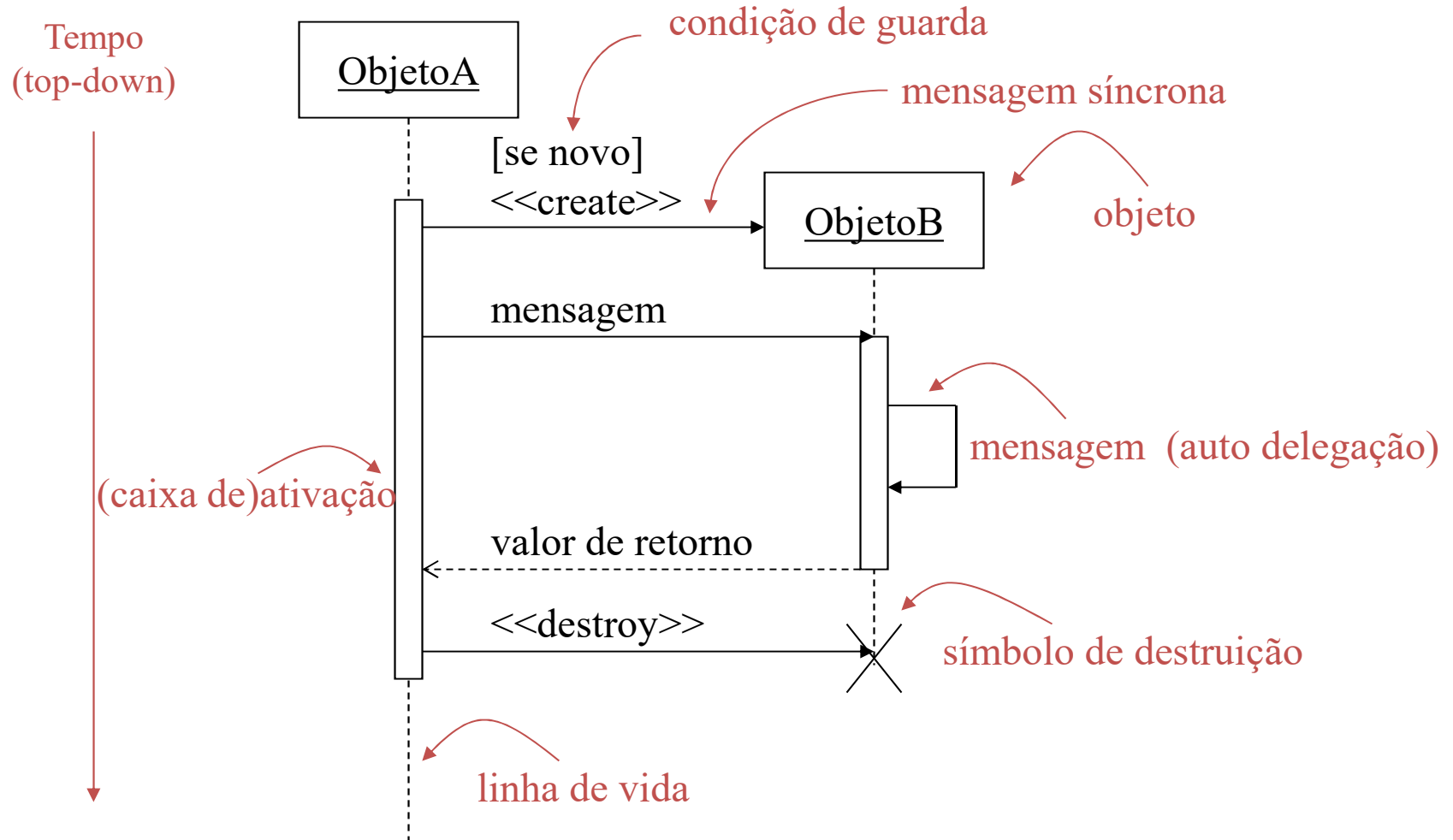
### UML: Diagrama de Sequência

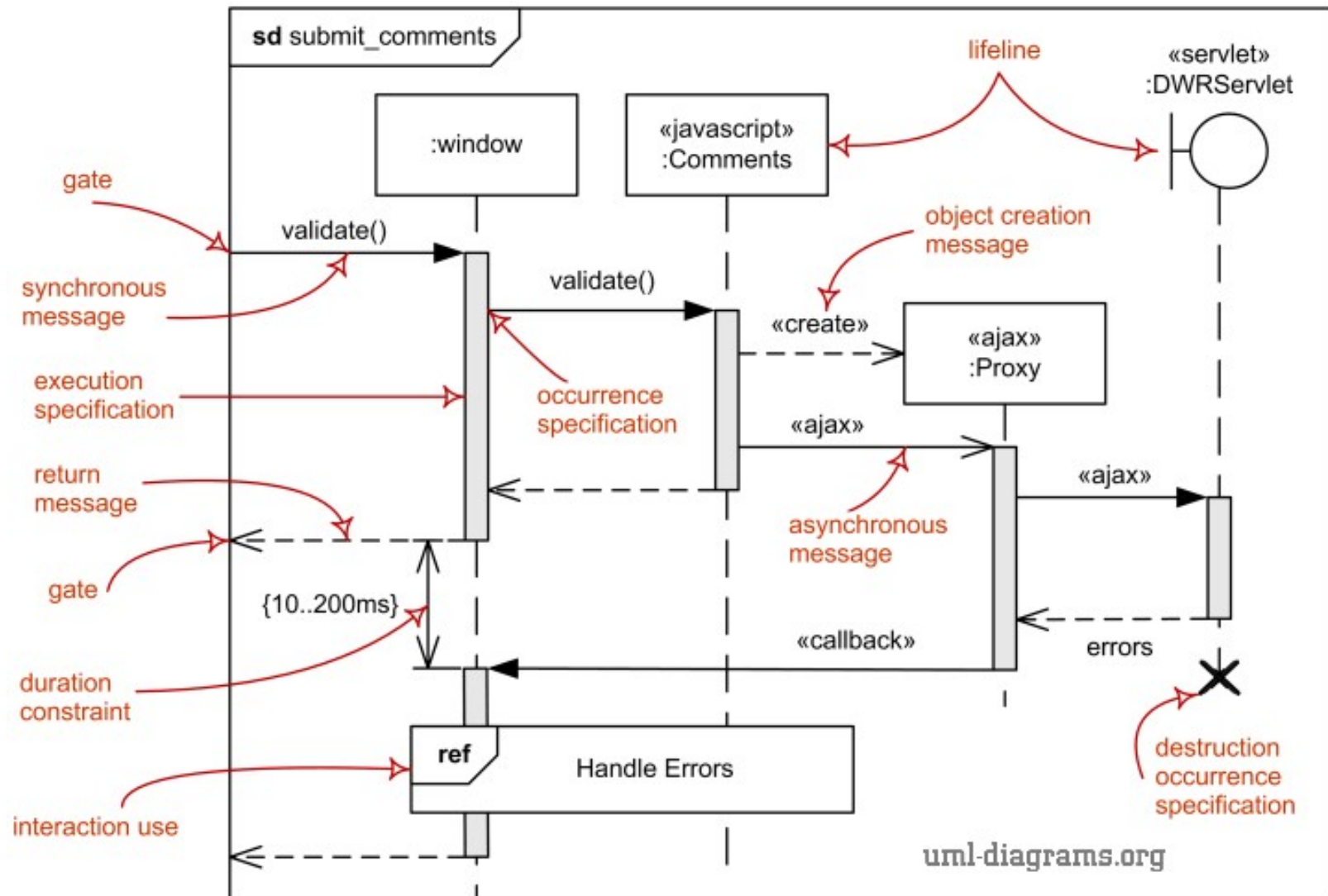
- Objetos podem possuir nomes
  - **obj:Classe**

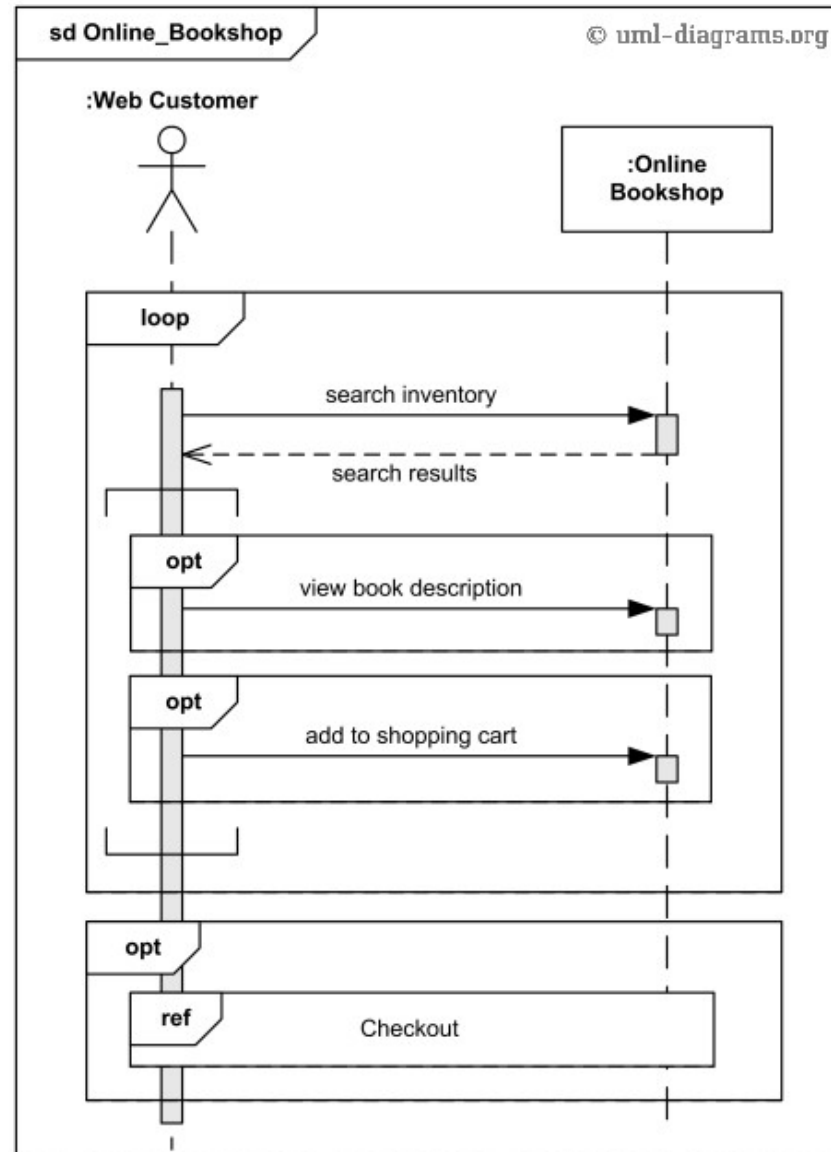
Ex.: carlos : Aluno (objeto do tipo aluno chamado de “carlos”)  
      :Aluno (um objeto Aluno não identificado)



## UML: Diagrama de Sequência







<https://www.uml-diagrams.org/online-shopping-uml-sequence-diagram-example.html?context=seq-examples>



Ler o conteúdo: <https://www.uml-diagrams.org/sequence-diagrams.html>

---

# *Design Pattern*

## *Singleton*

---





## Padrões de Projeto (*Design Patterns*)

**Padrões de projeto** são soluções gerais para problemas que ocorrem com frequência dentro de um determinado contexto no projeto de software.

Não são projetos finalizados que podem ser transformados diretamente em código fonte. **São descrições ou modelos (*templates*) que se propõem a resolver diferentes problemas.**

São práticas formalizadas que o programador pode utilizar para resolver esses problemas comuns.

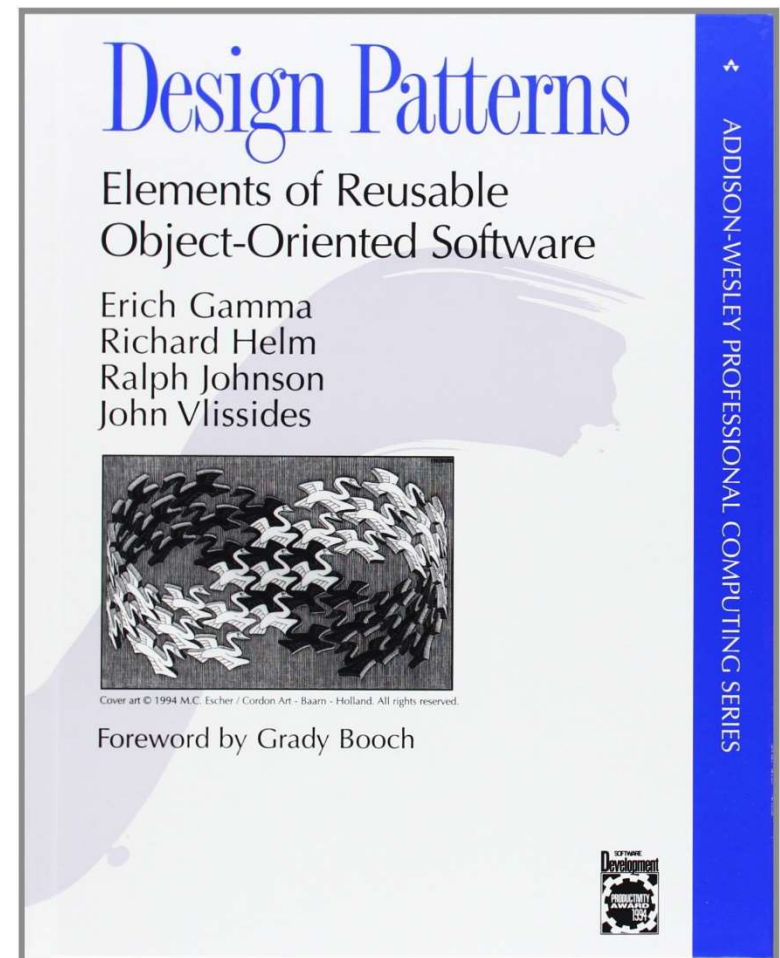


## Padrões de Projeto (*Design Patterns*)

- **Padrões GOF**  
*Gang of Four*
- **Padrões GRASP**  
*General Responsibility Assignment Software Patterns (or Principles)*

### ***Design Patterns: Tipos de Padrões GOF (Gang of Four)***

- Padrões de Criação
- Padrões Estruturais
- Padrões Comportamentais



## Design Patterns: Tipos de Padrões GOF (Gang of Four)

Criação	Estrutural	Comportamental	
Abstract Factory	Adapter	Chain of Responsibility	State
Builder	Bridge	Command	Strategy
Factory Method	Composite	Interpreter	Template Method
Prototype	Decorator	Iterator	Visitor
Singleton	Facade	Mediator	
	Flyweight	Memento	
	Proxy	Observer	

### Design Patterns: **Singleton**

#### Problema

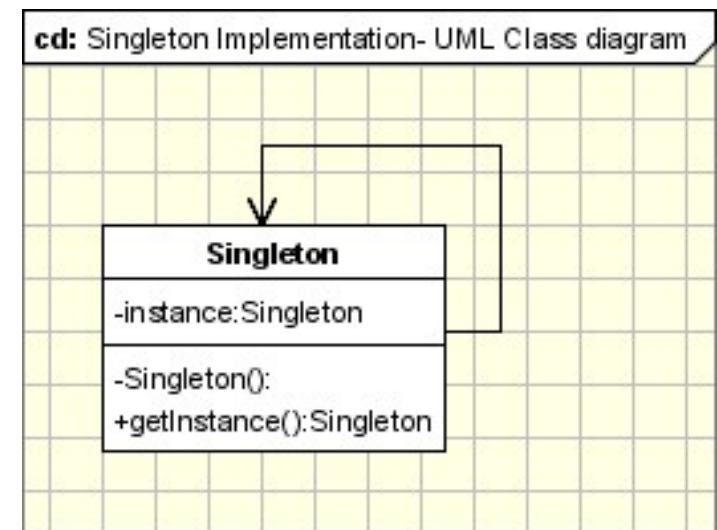
Uma classe precisa ter uma única instância.

#### Solução

Garante que uma classe terá apenas uma instância.

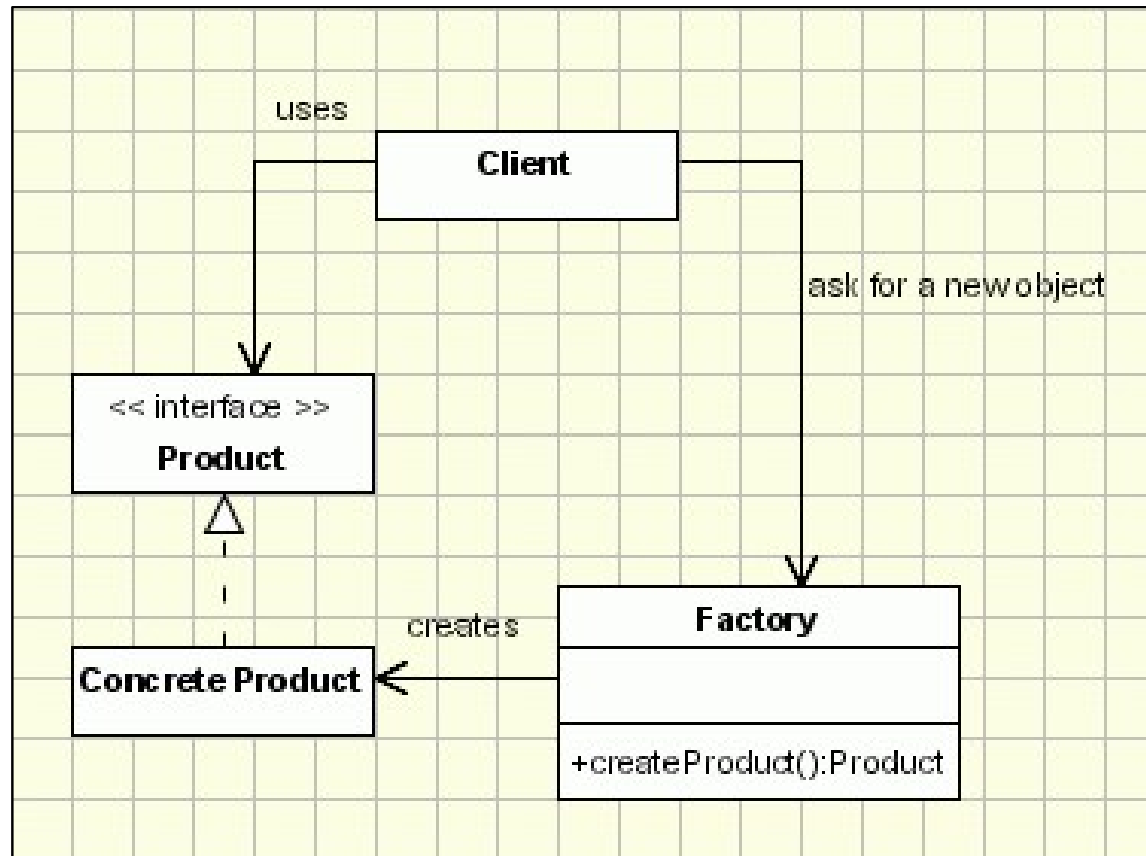
#### Consequência

Fácil acesso a gerência de recursos compartilhados, como variáveis globais.



<http://www.oodeesign.com/singleton-pattern.html>

### Design Patterns: **Factory Method**



<http://www.oodeesign.com/factory-pattern.html>



## ***Design Patterns: Tipos de Padrões GRASP***

### ***General Responsibility Assignment Software Patterns (or Principles)***

Assim como os padrões de projeto do GOF, os padrões GRASP são utilizados para resolução de problemas comuns e bastante típicos de desenvolvimento de software orientado a objeto. Portanto, tais técnicas apenas documentam e normatizam as práticas já consolidadas, testadas e conhecidas no mercado.



**Design Patterns: Tipos de Padrões GRASP**  
*General Responsibility Assignment Software Patterns (or Principles)*

GRASP	
Controller	Polymorphism
Creator	Protected Variations
Indirection	Pure Fabrication
Information Expert	
High Cohesion	
Loose Coupling	





### *Design Patterns*

- *Visam facilitar a reutilização de soluções de desenho – isto é, soluções na fase de projeto do software.*
- *Estabelecem um vocabulário comum de desenho, facilitando comunicação, documentação e aprendizado dos sistemas de software.*