

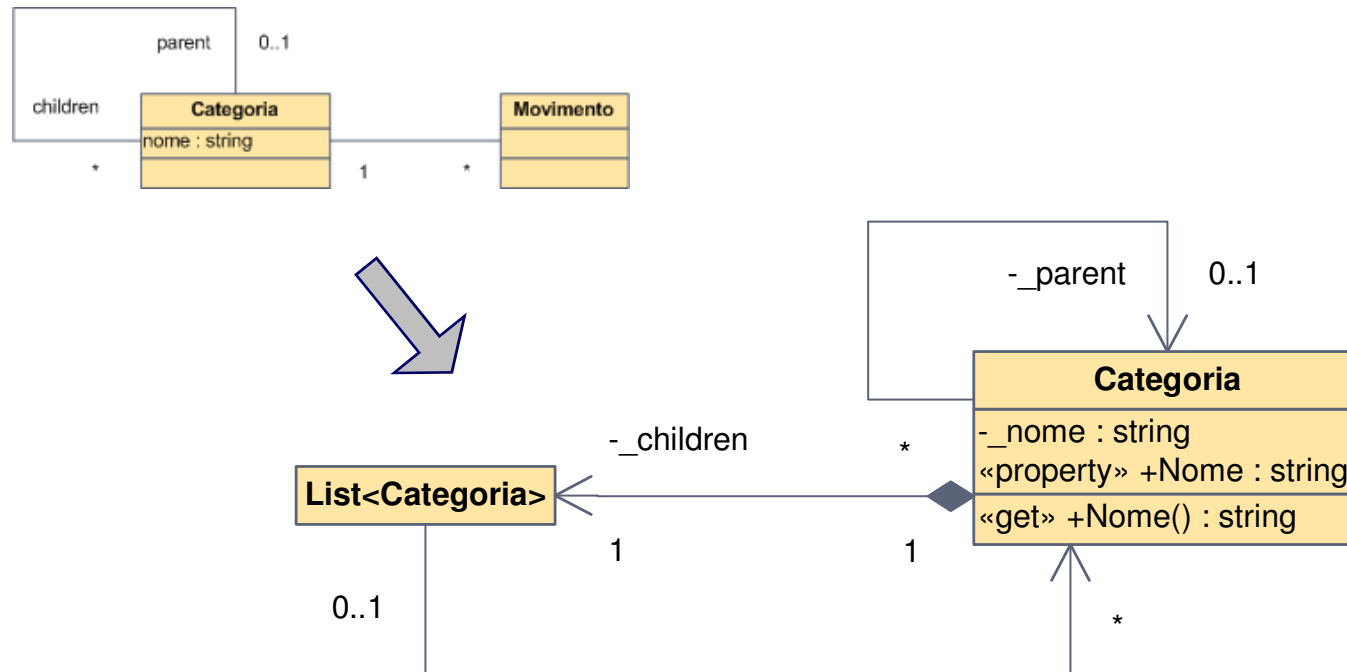
# Progettazione

- ▶ Prevede la specifica completa del sistema
- ▶ Si deve tenere conto
  - del **tipo di applicazione**
    - standalone
    - web
    - ...
  - delle **strategie di persistenza**
    - database relazionale
    - file testo (xml o altro)
    - file binari (serializzazione binaria o altro)
    - ...
  - del **framework** e del **linguaggio di programmazione** (meno possibile)
    - .NET e C# (nel nostro caso)

# Diagramma delle classi di progettazione

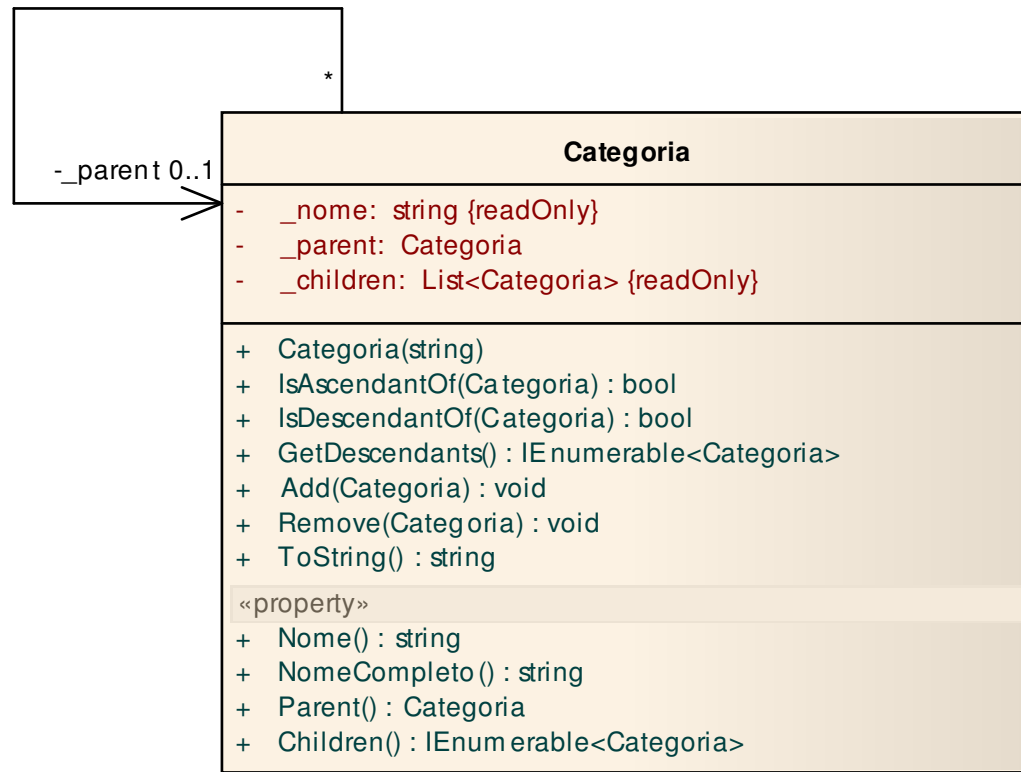
- ▶ Rendere il diagramma delle classi di analisi “implementabile”
  - Trasformare le **classi associazione** in classi normali
  - Determinare la **navigabilità** delle associazioni
  - Aggiungere le **classi contenitore** (elencare solo attributi e/o operazioni non standard)
  - Completare elenco **attributi** e **operazioni**
  - Specificare **tipi di dato** e **visibilità**
  - ...
- ▶ Applicare **principi** e **pattern di progettazione**

# Classi di progettazione Categorie<sub>1</sub>

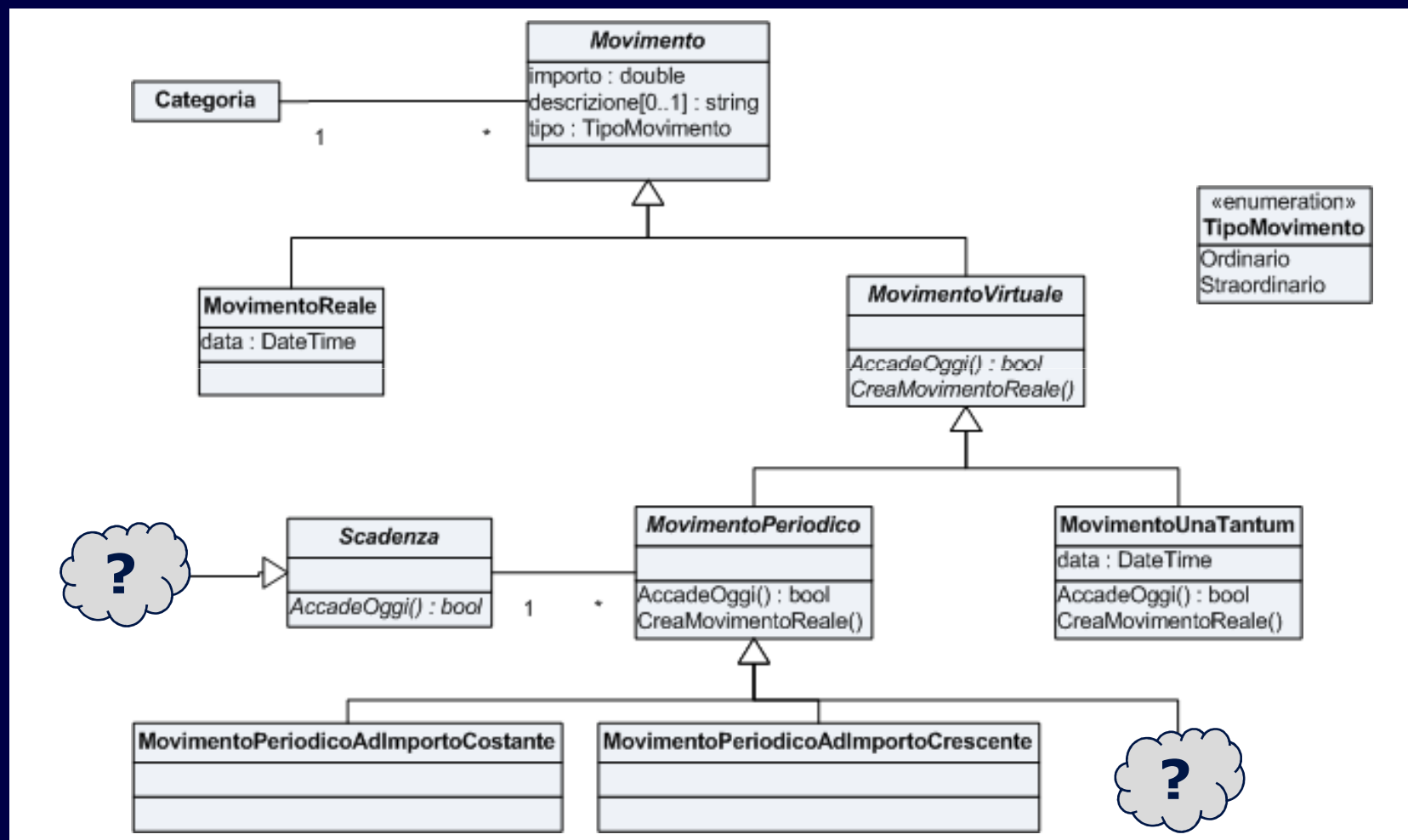


Struttura ad albero ► Composite semplificato

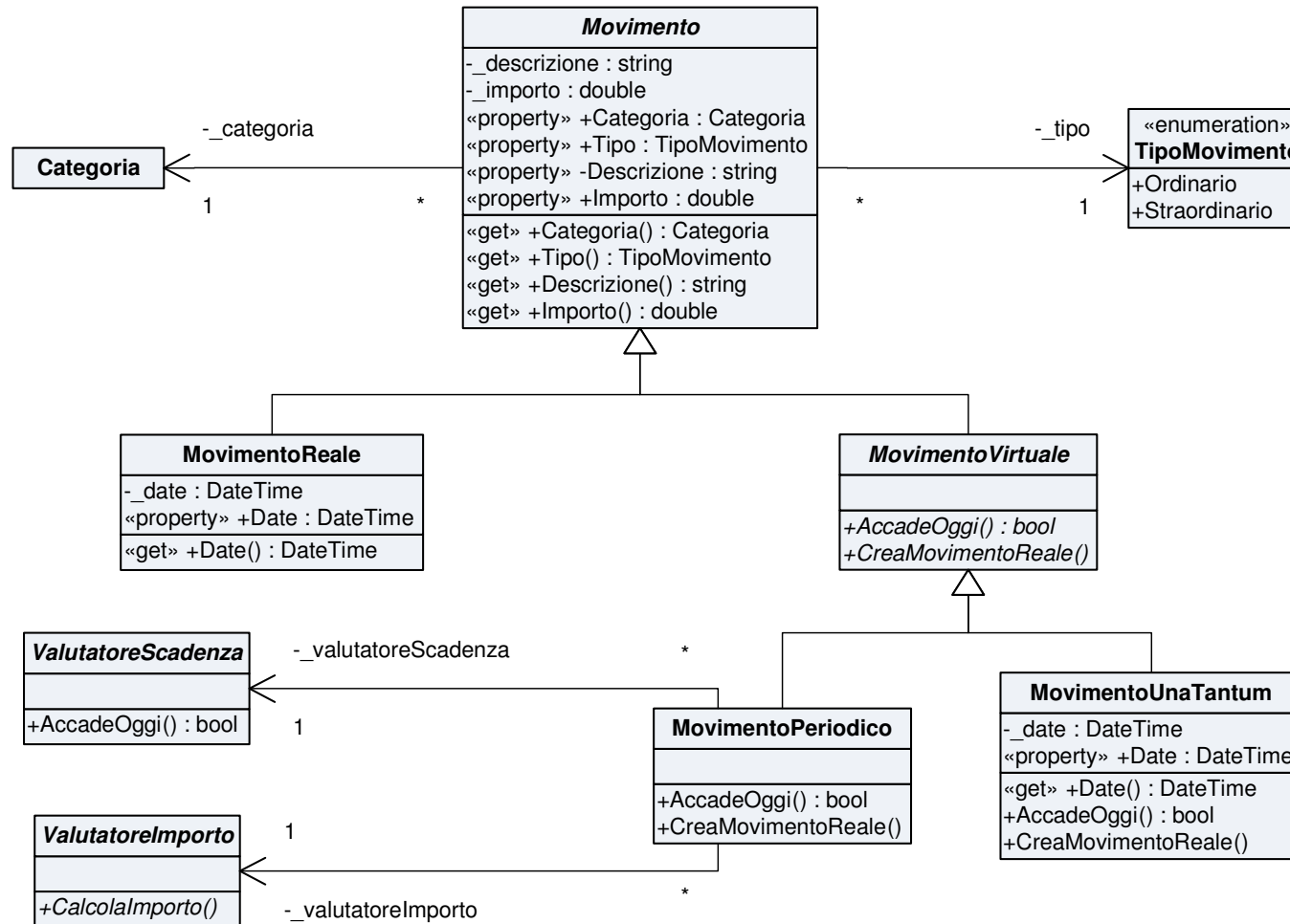
# Classi di progettazione Categorie<sub>2</sub>



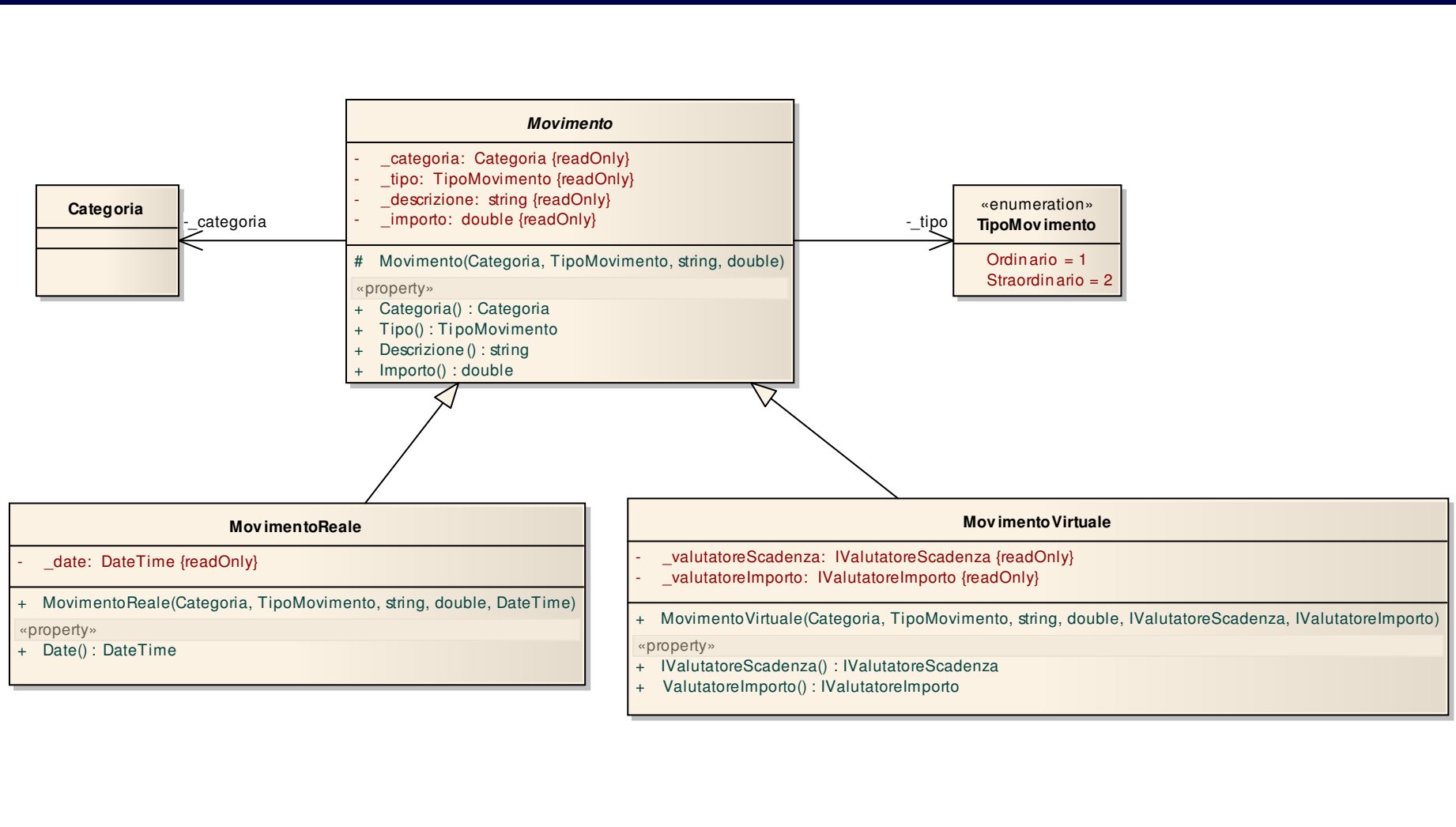
# Classi di analisi Movimenti



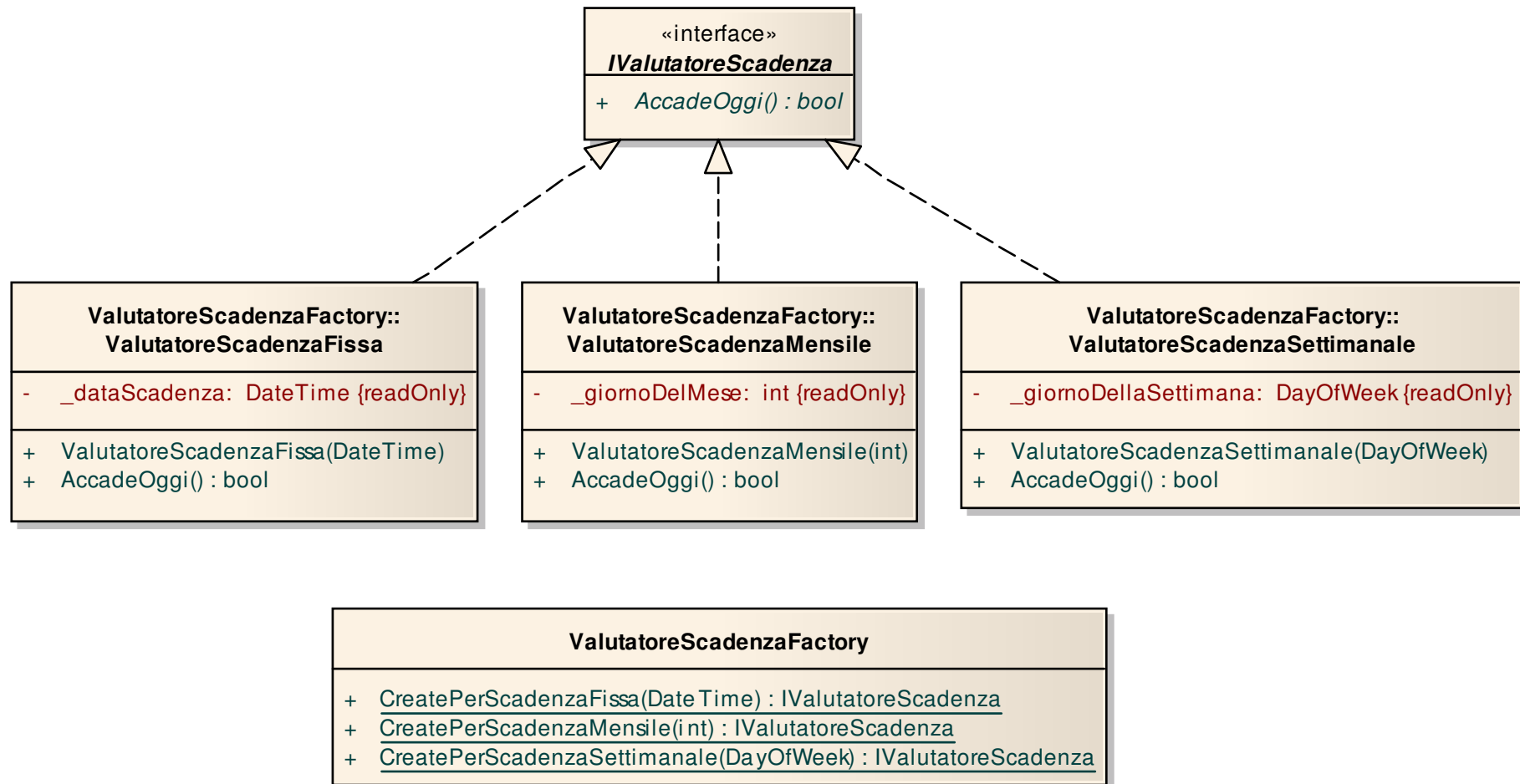
# Classi di progettazione Movimenti<sub>1</sub>



# Classi di progettazione Movimenti<sub>2</sub>

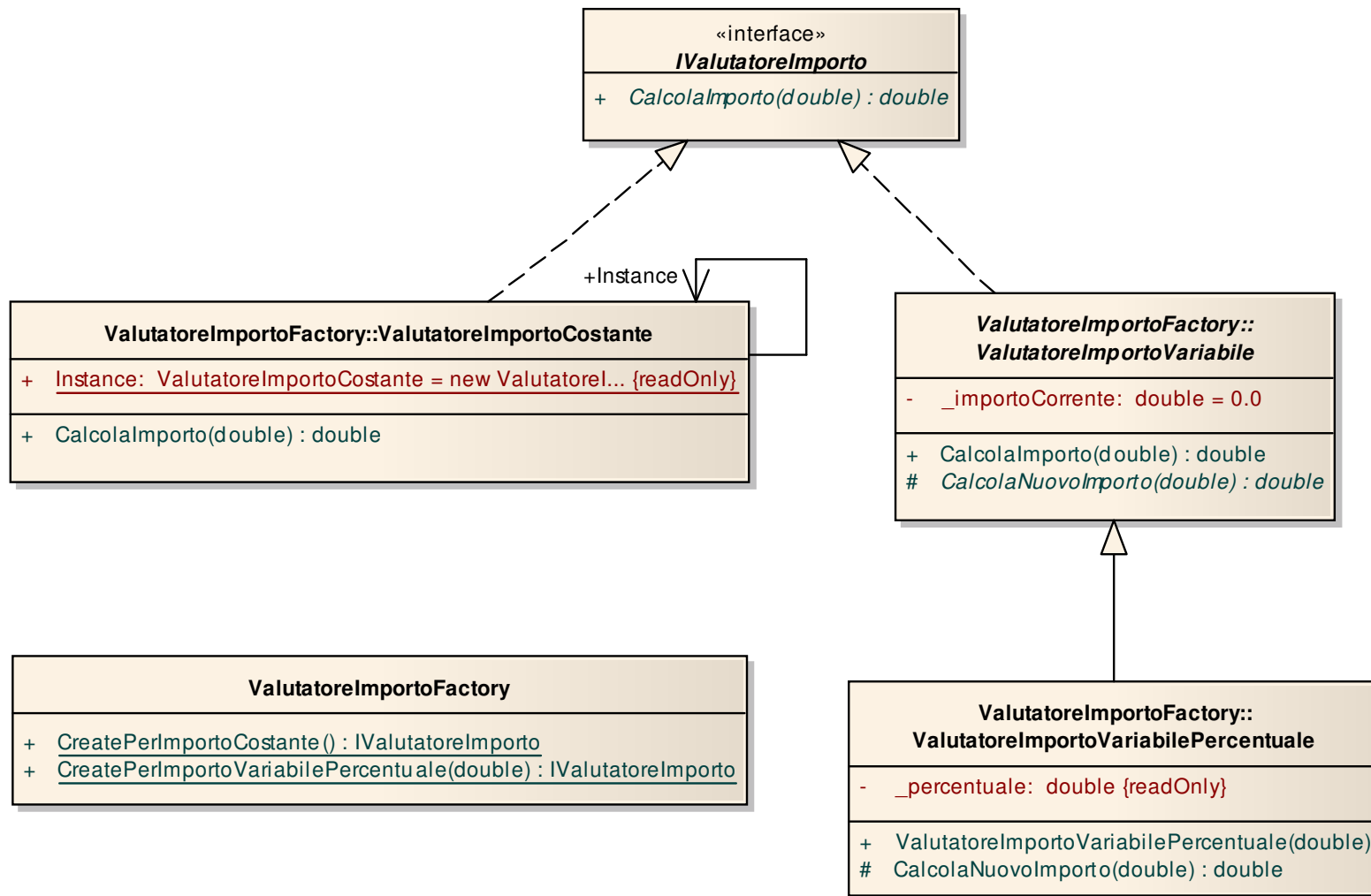


# Classi di progettazione Movimenti<sub>2</sub>





# Classi di progettazione Movimenti<sub>2</sub>



# Classi di progettazione

## Movimenti<sub>2</sub>

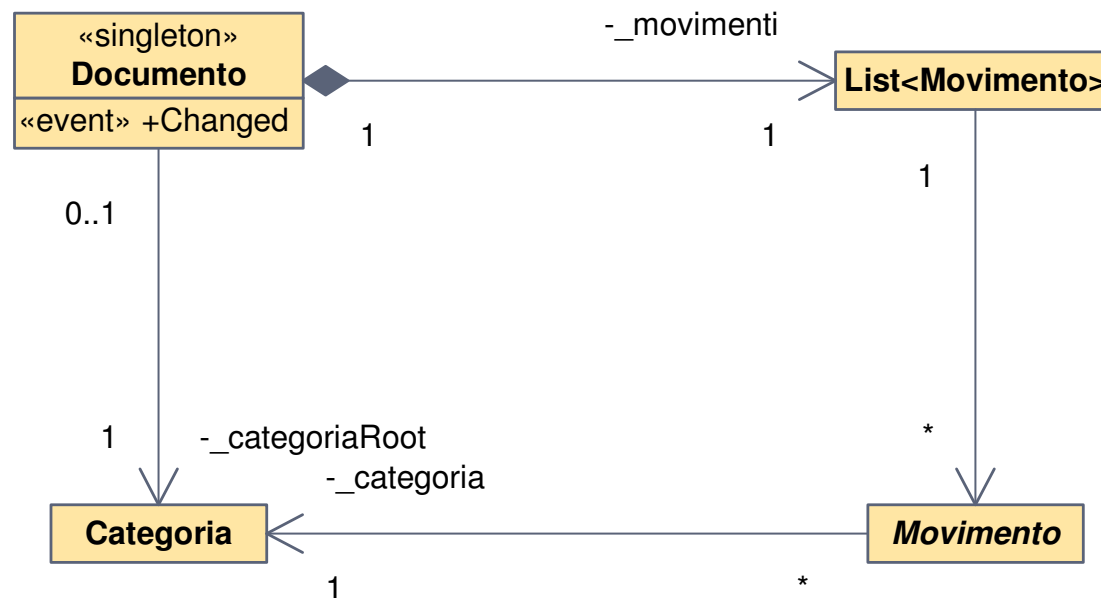
### MovimentoVirtualeFactory

- + CreateMovimentoUnaTantum(Categoria, TipoMovimento, string, double, DateTime) : MovimentoVirtuale
- + CreateMovimentoSettimanaleConCrescitaPercentuale(Categoria, TipoMovimento, string, double, DayOfWeek, double) : MovimentoVirtuale

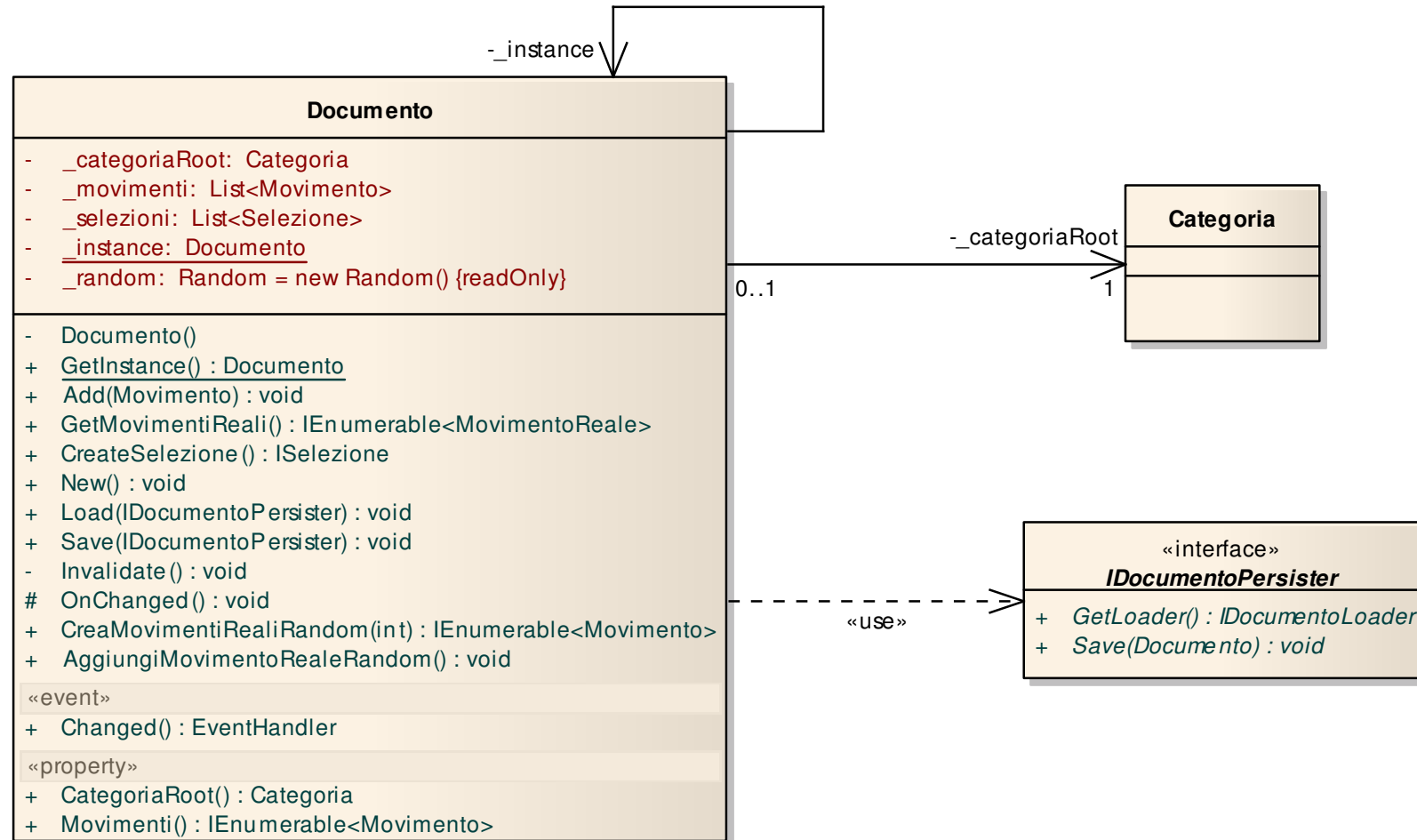
# Classi di progettazione Documento

- ▶ Per una questione pratica sarebbe bene che Categorie e Movimenti fossero facilmente raggiungibili → tipicamente si realizza un *singleton* (**Documento**) contenente i dati “importanti”
- ▶ Nel caso specifico, il Documento contiene:
  - la Categoria radice dell’albero delle categorie
  - la collezione di tutti i Movimenti
  - altro?
- ▶ Il Documento deve notificare ogni cambiamento → pattern *observer* / eventi

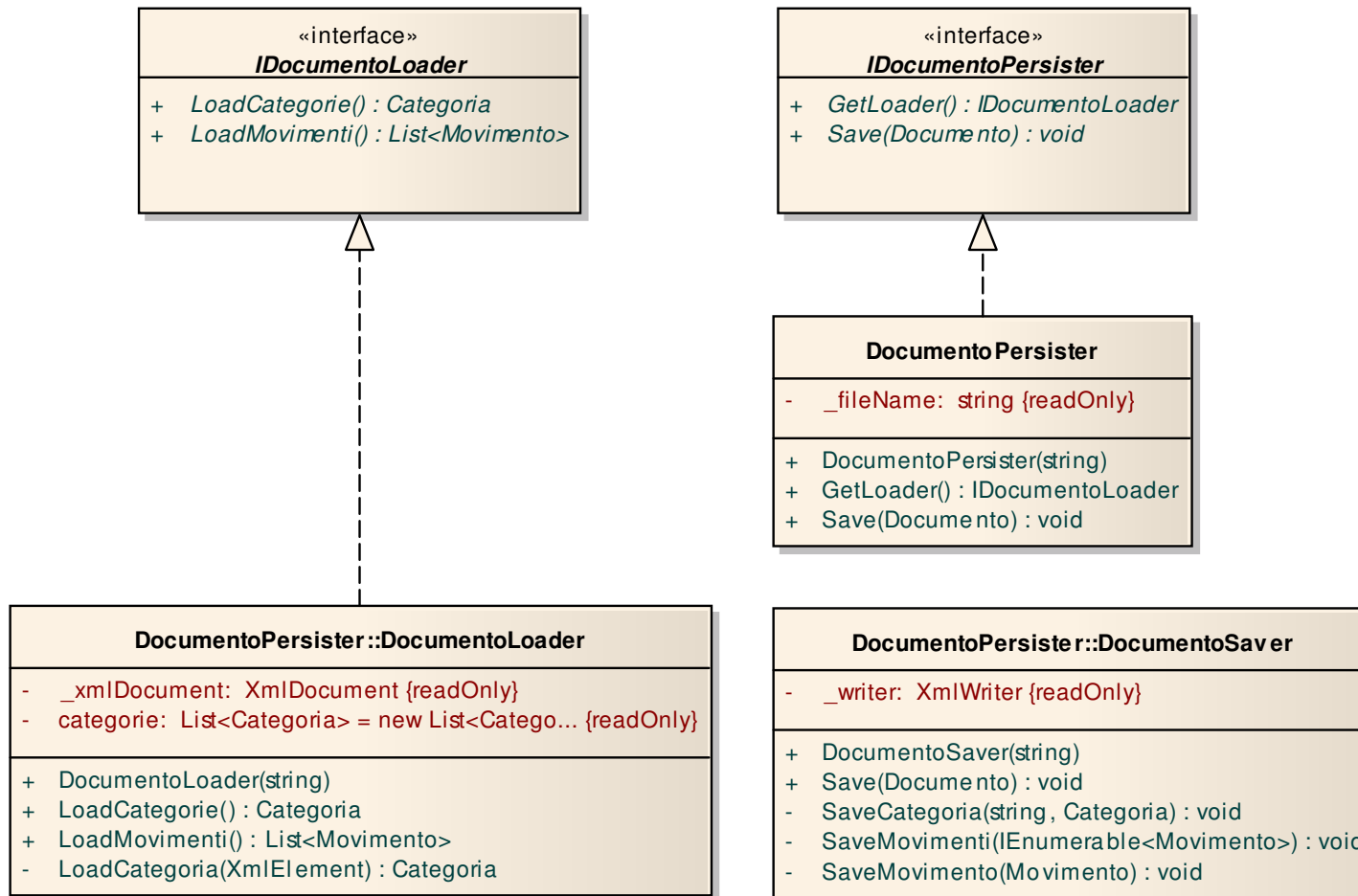
# Classi di progettazione Documento<sub>1</sub>



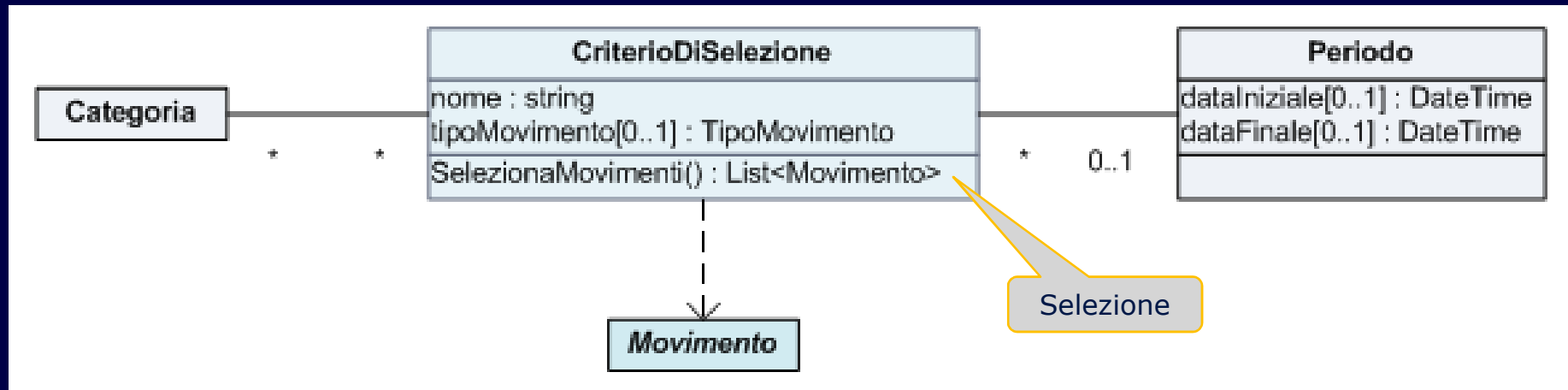
# Classi di progettazione Documento<sub>2</sub>



# Classi di progettazione Documento<sub>2</sub>

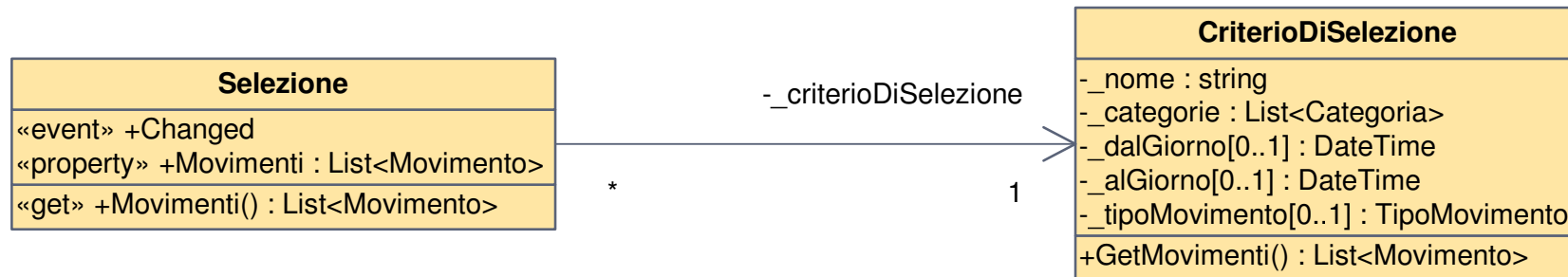


# Classi di progettazione Selezioni



- ▶ Molte soluzioni possibili:
  - 1 sola classe
  - 1 gerarchia di decoratori
  - 2 classi (Selezione + CriterioDiSelezione)
  - 1 classe (Selezione) + 1 gerarchia di decoratori
- ▶ Per risolvere il problema delle selezioni “vive”:
  - registrazione all’evento Changed del Documento

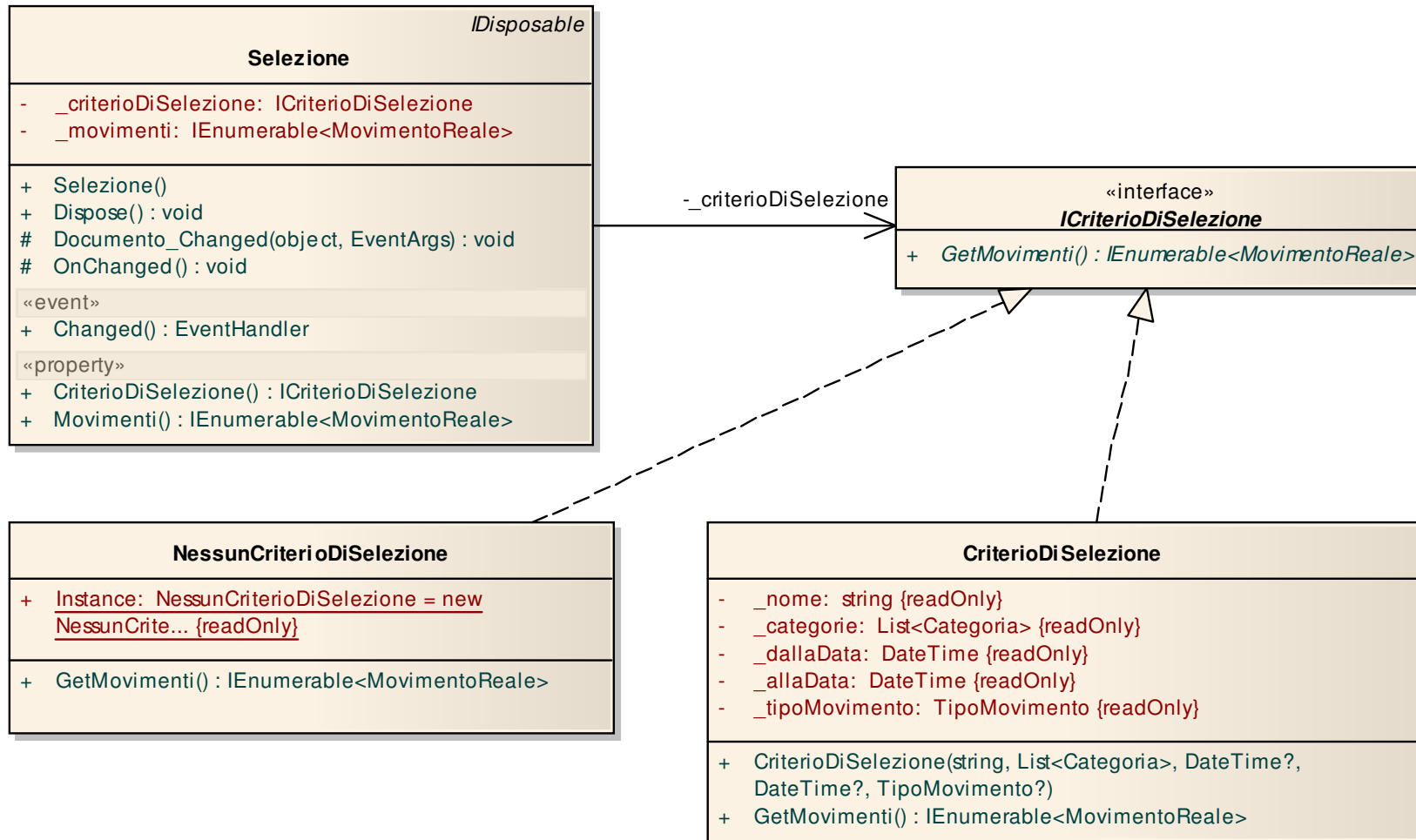
# Classi di progettazione Selezioni<sub>1</sub>



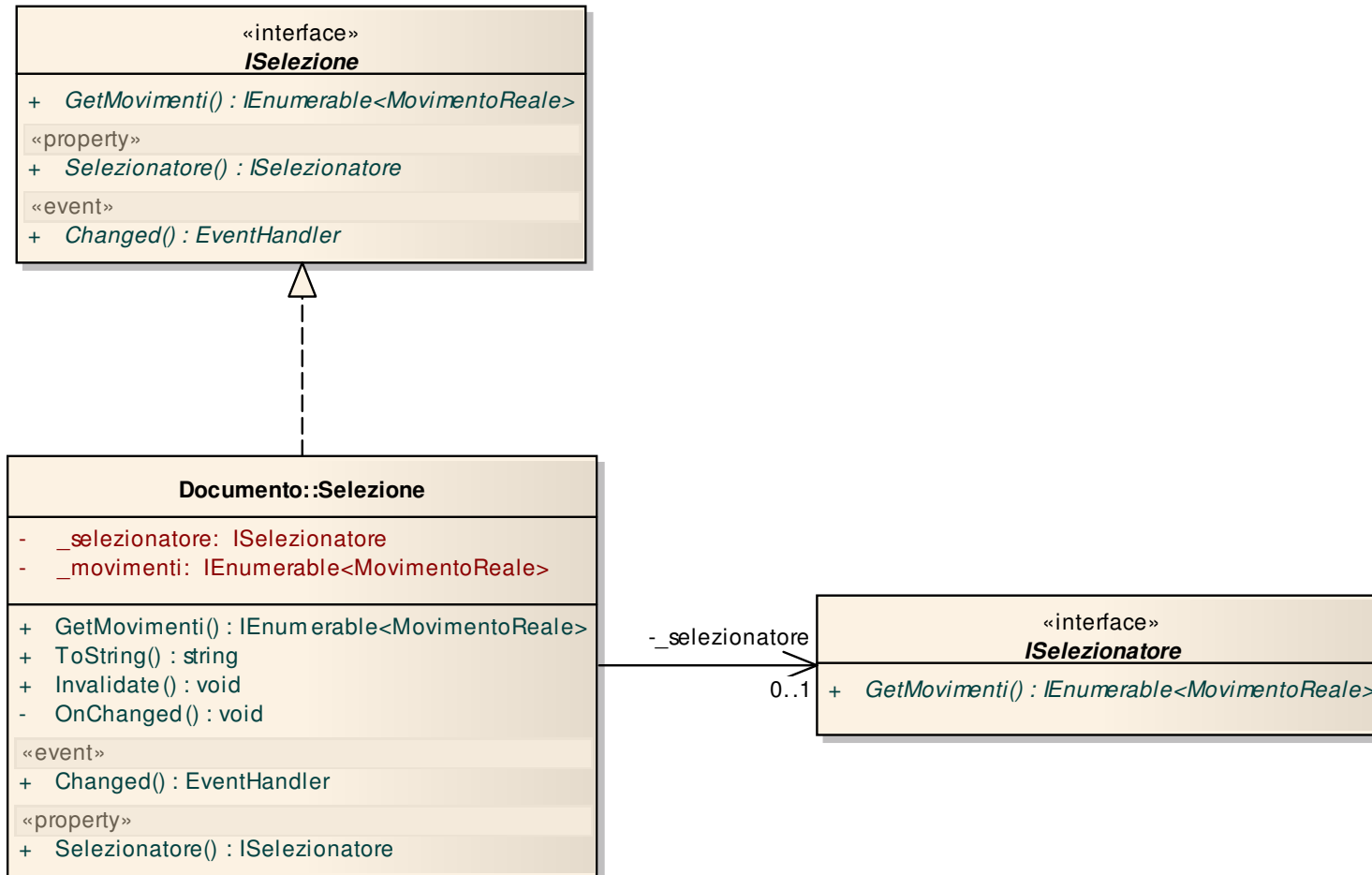
- Il criterio di selezione potrebbe essere realizzato con una query LINQ



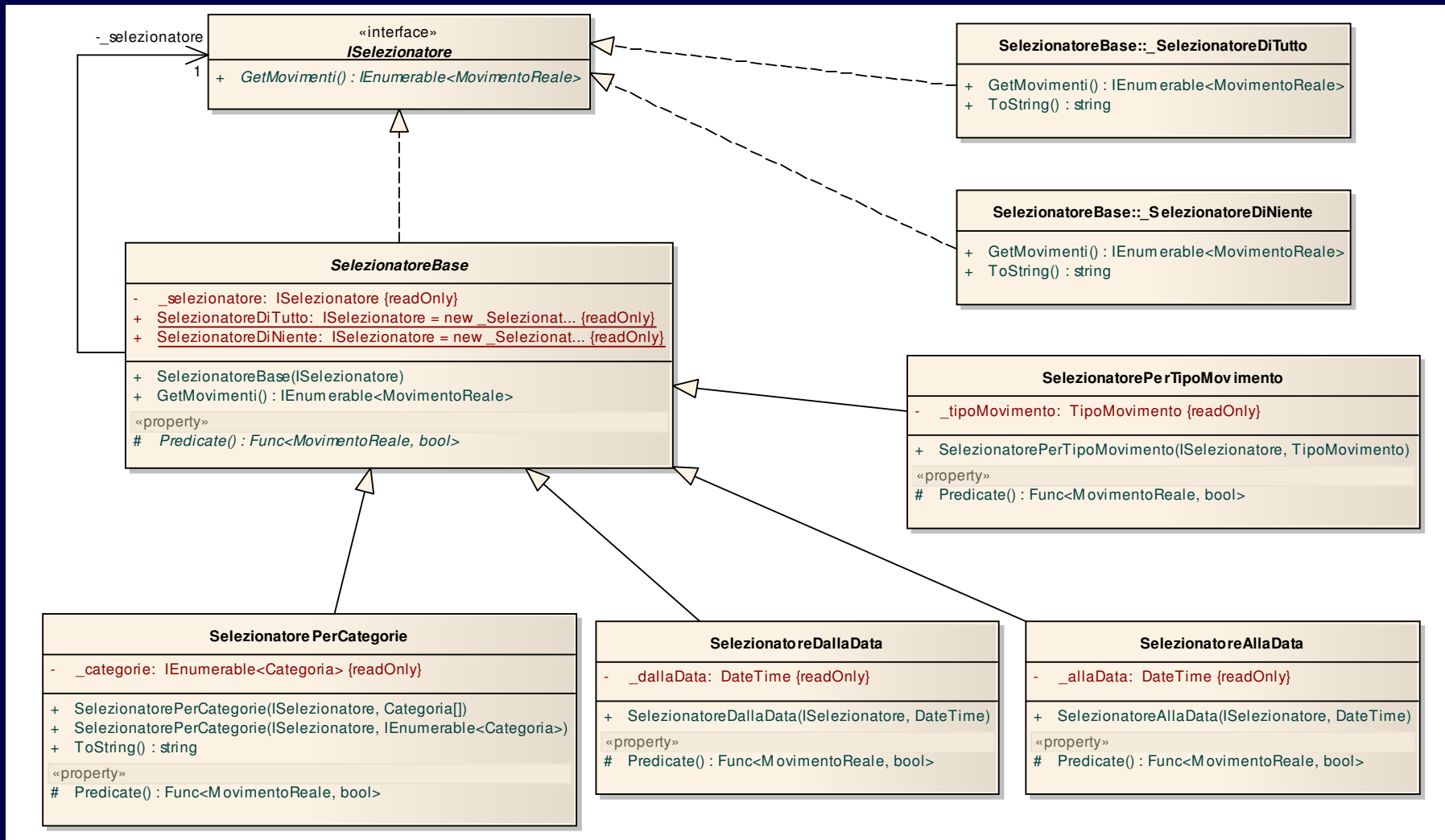
# Classi di progettazione Selezioni<sub>2</sub>



# Classi di progettazione Selezioni<sub>3</sub>



# Classi di progettazione Selezioni<sub>3</sub>



# Classi di progettazione Calcoli<sub>1</sub>

