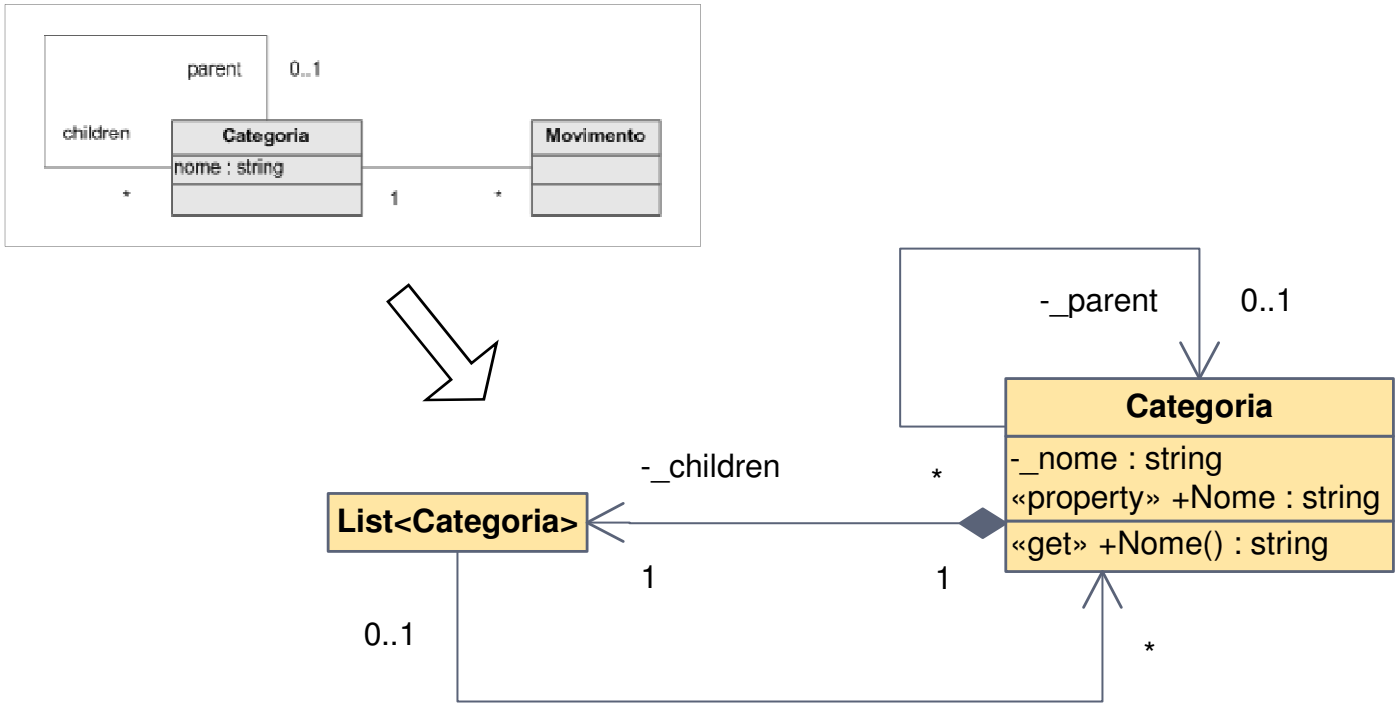
Progettazione

- 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

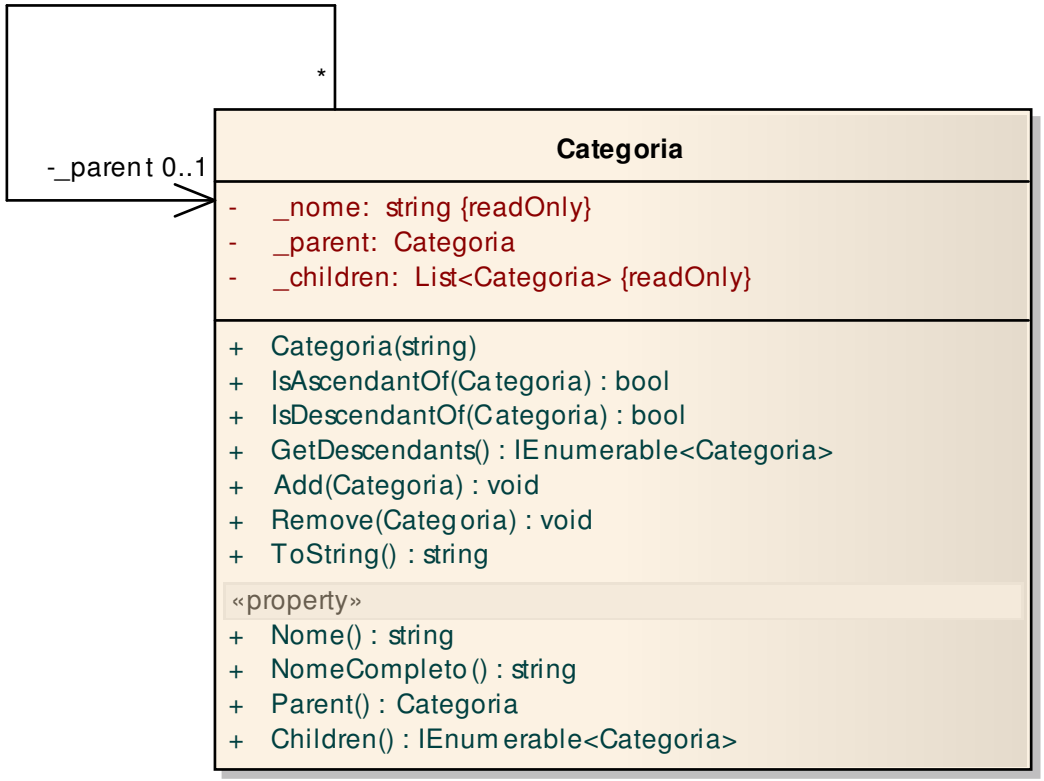
- Rendere il diagramma delle classi di analisi "implementabile"
  - Trasformare le classi associazione in classi normali
  - Determinare la navigabilità delle associazioni
  - Aggiungere le classi contenitore (elencare solo attributi e/o operazioni non standard)
  - Completare elenco attributi e operazioni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

Classi di progettazione
Categorie$_1$

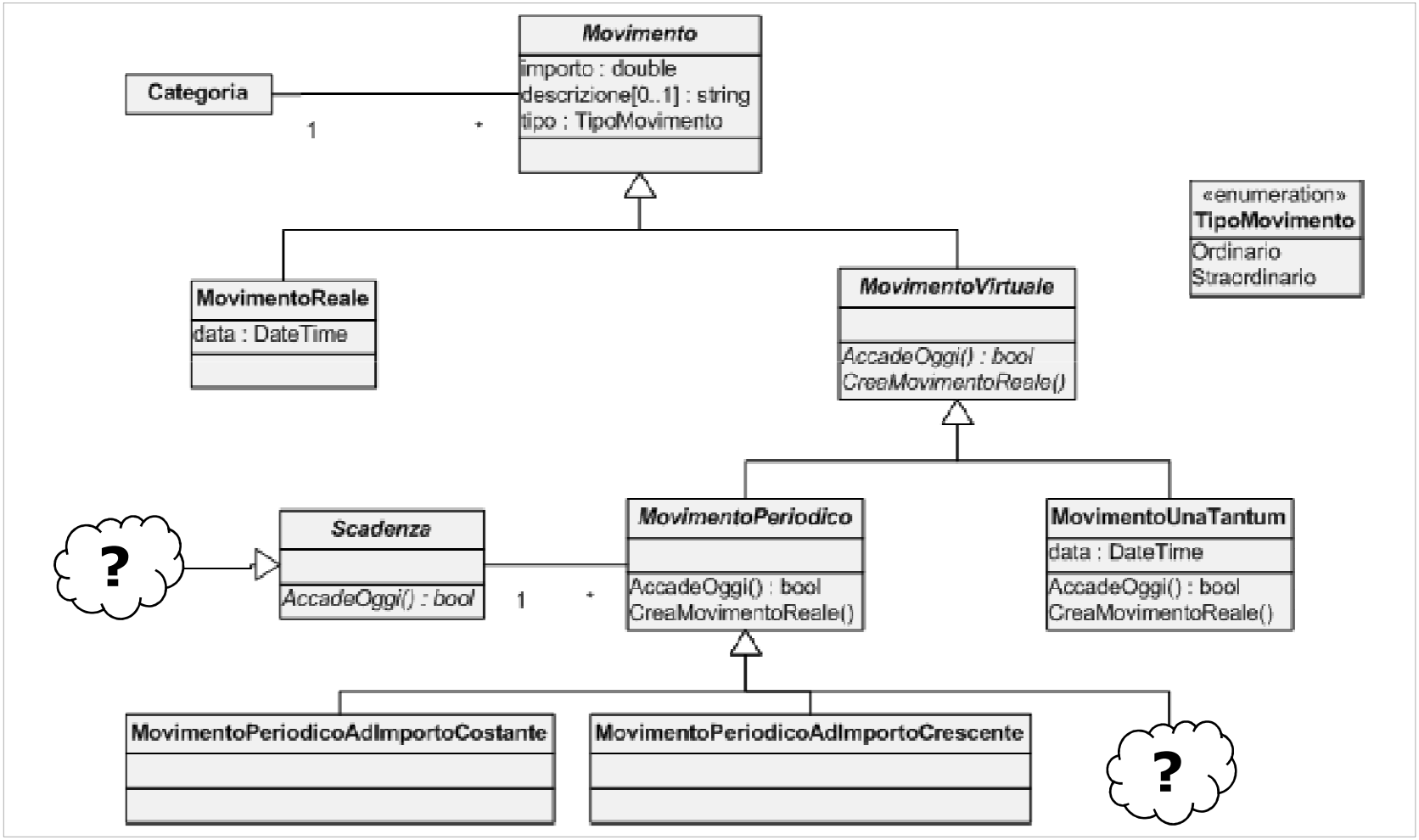


Struttura ad albero ▶ Composite semplificato

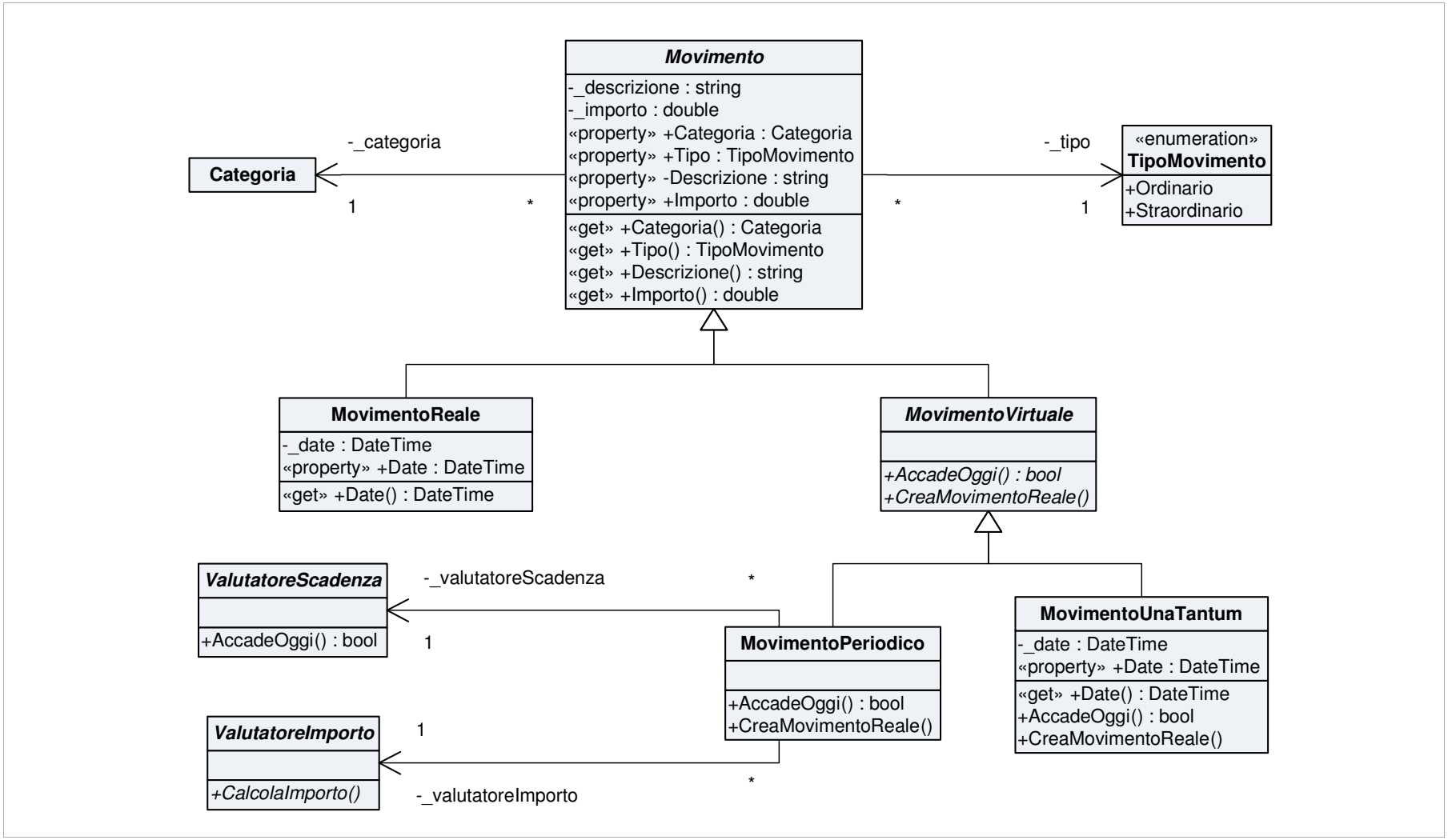
Classi di progettazione Categorie$_2$



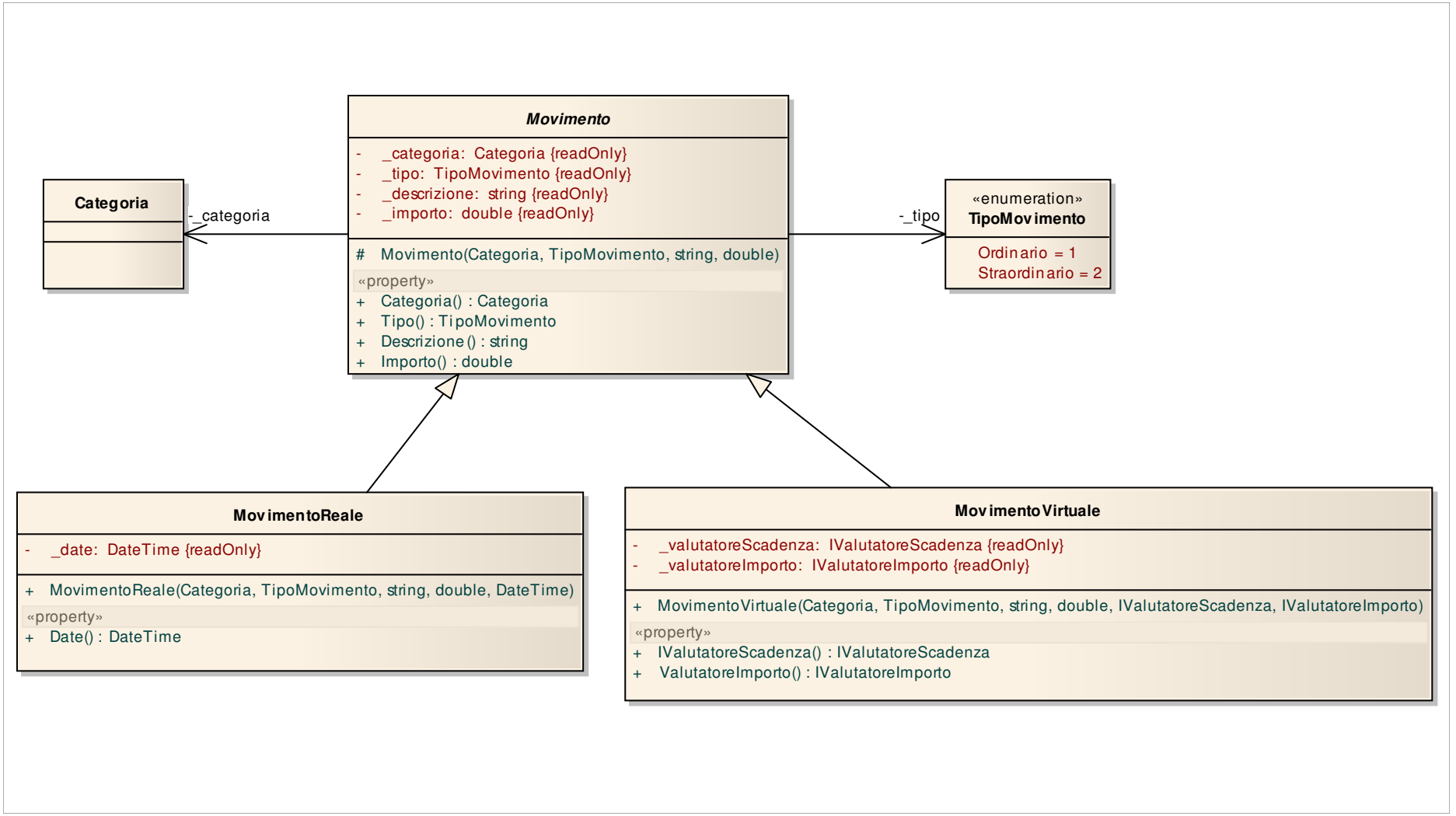
Classi di analisi
Movimenti



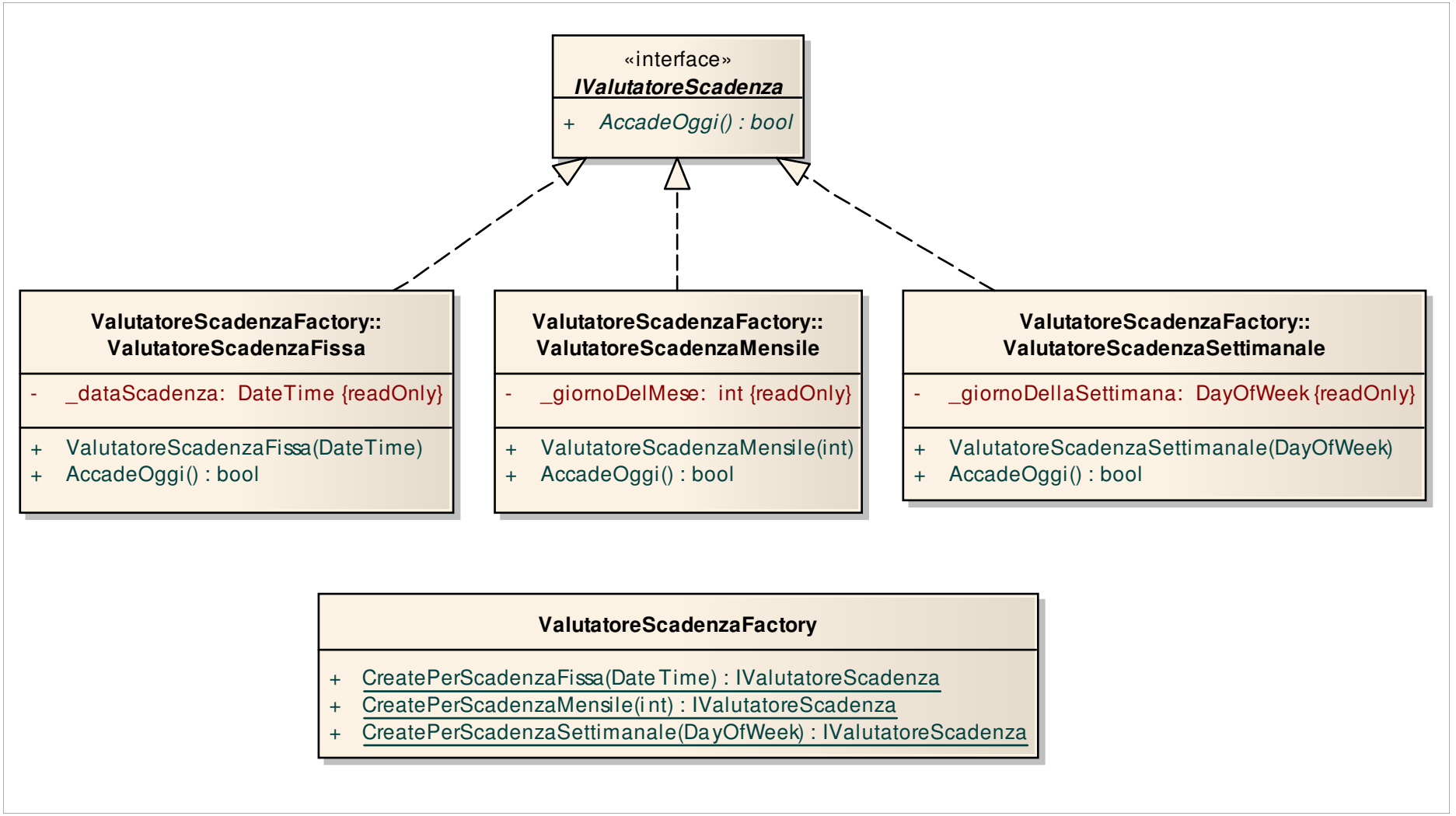
Classi di progettazione Movimenti$_1$



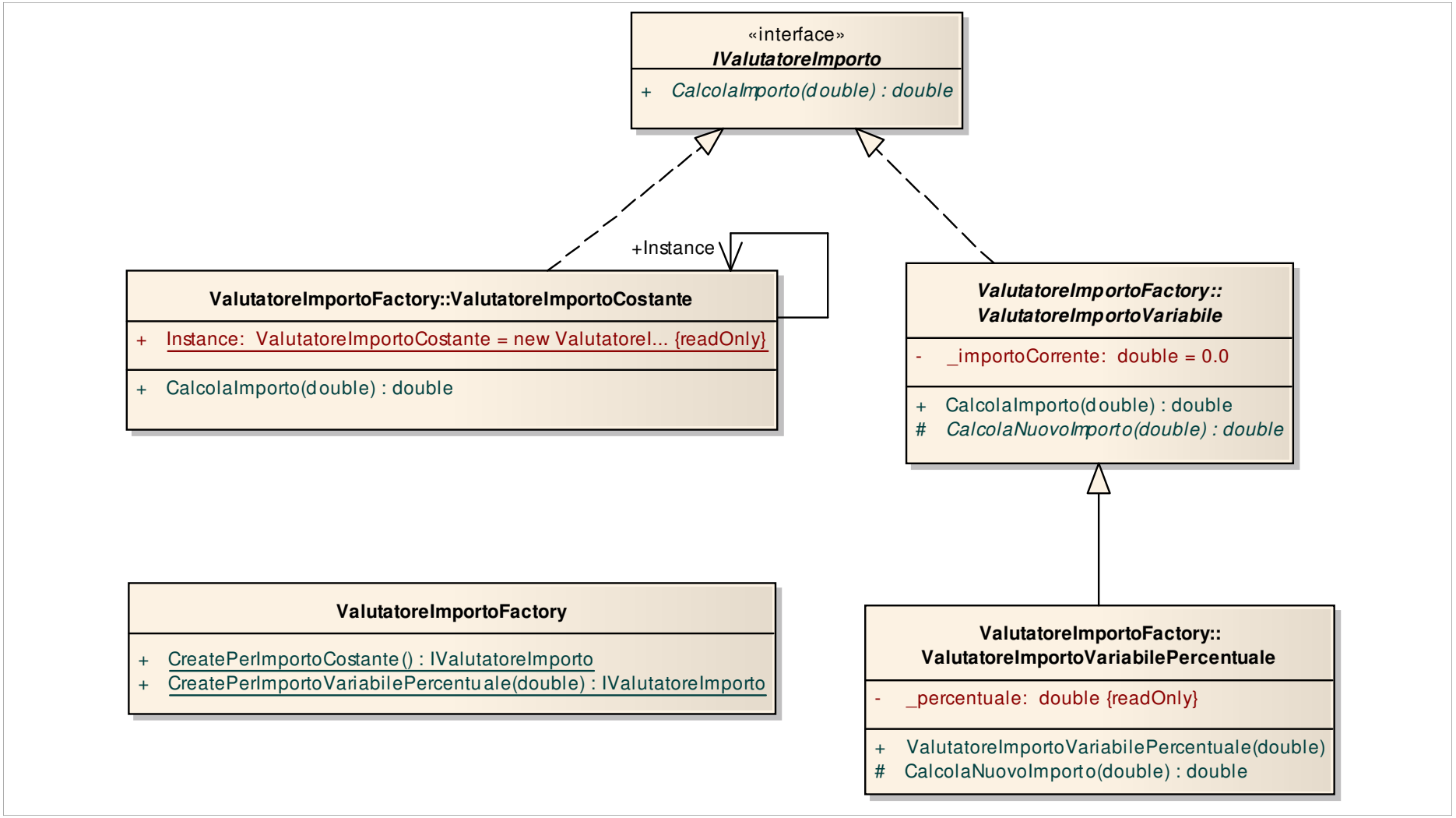
Classi di progettazione Movimenti$_2$



Classi di progettazione Movimenti$_2$



Classi di progettazione Movimenti$_2$

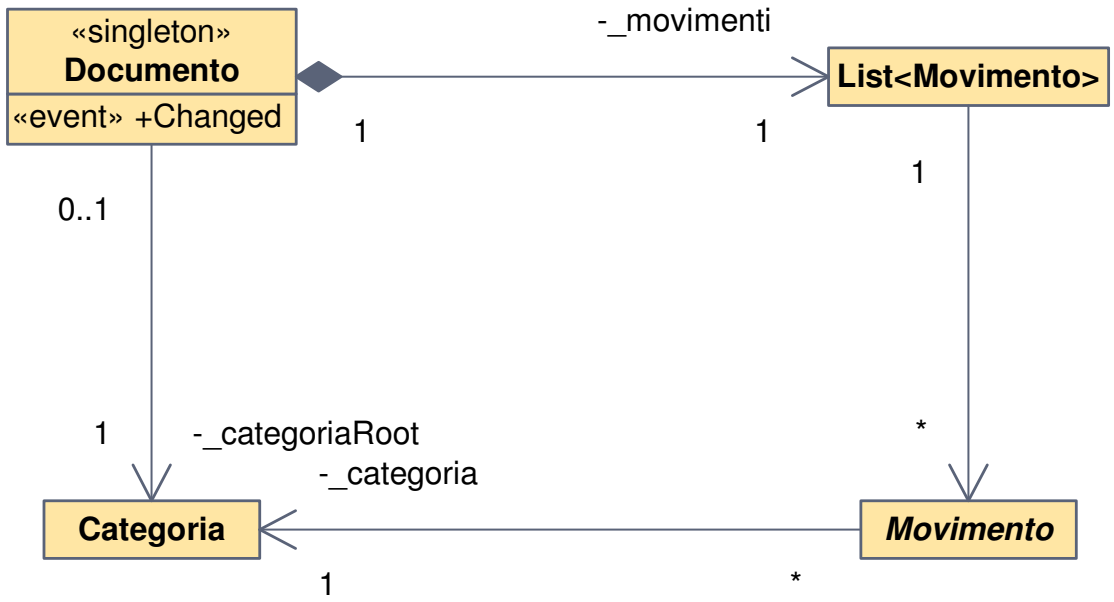


Classi di progettazione Movimenti$_2$

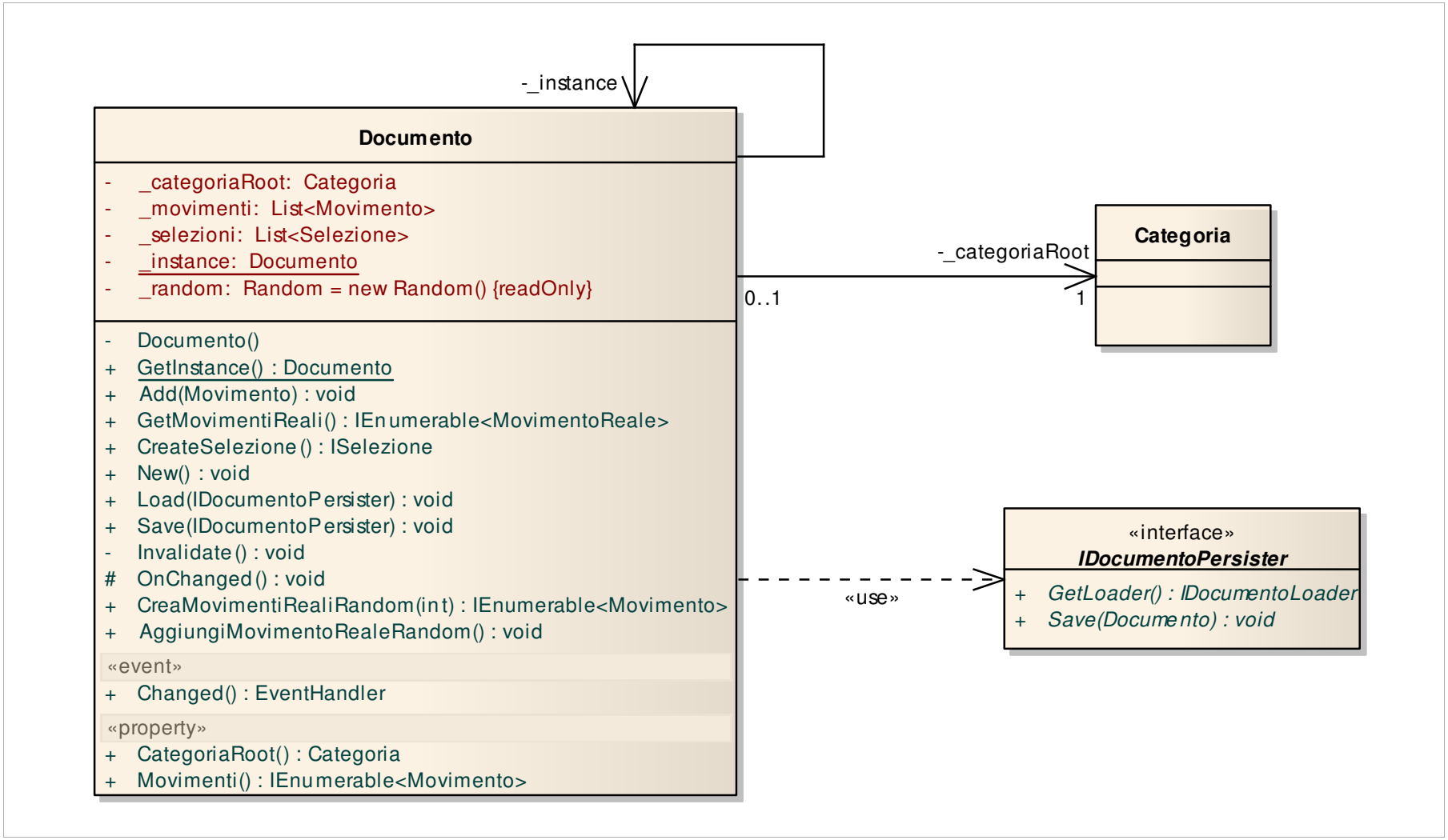
Classi di progettazione
Documento

- 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

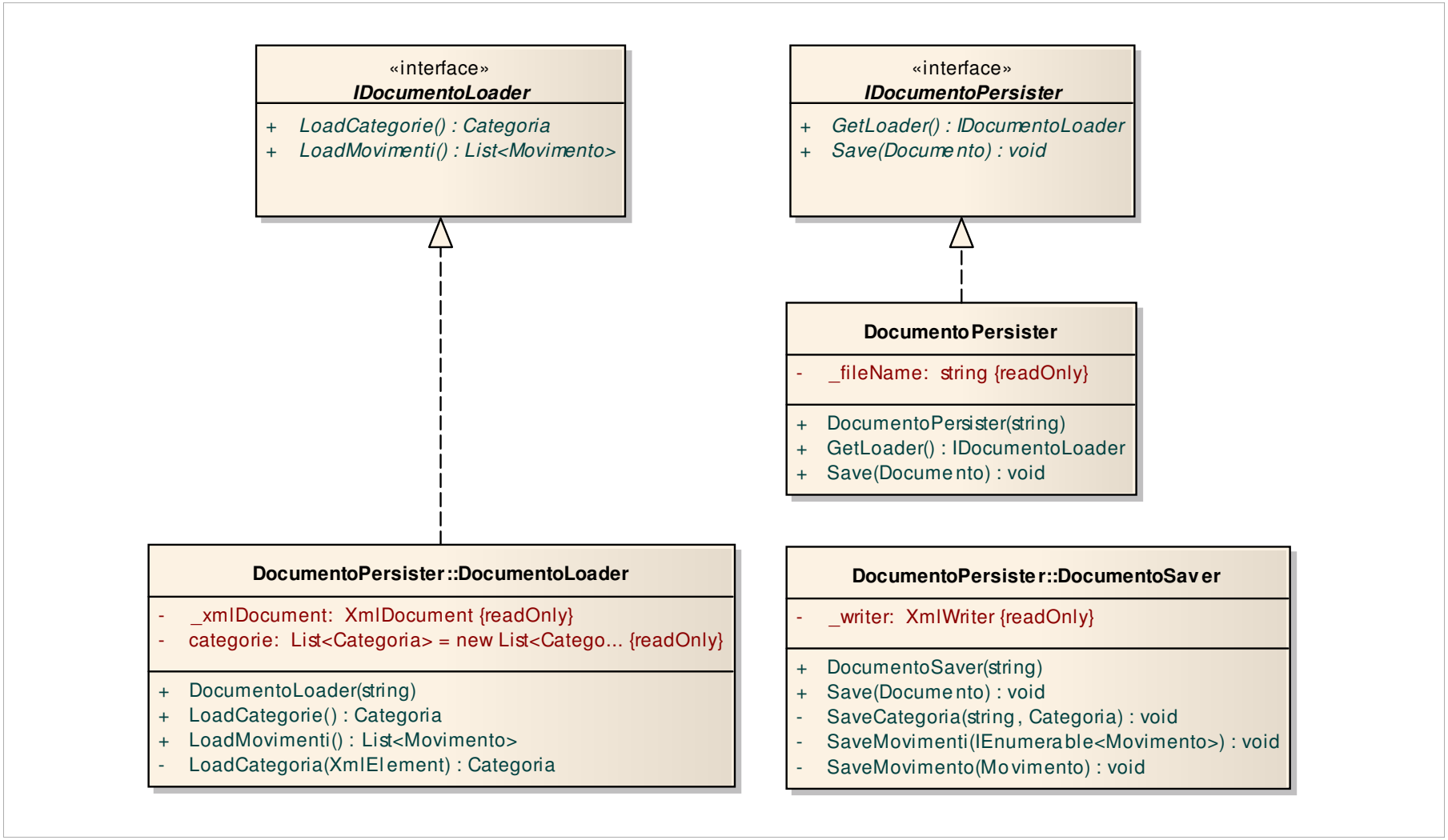
Classi di progettazione Documento$_1$



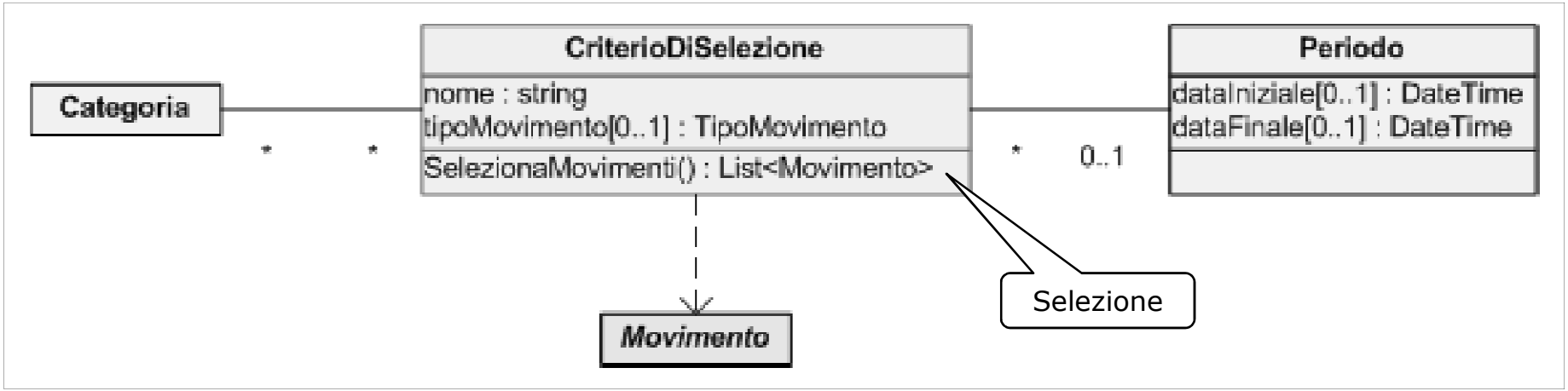
Classi di progettazione
Documento$_2$



Classi di progettazione Documento$_2$

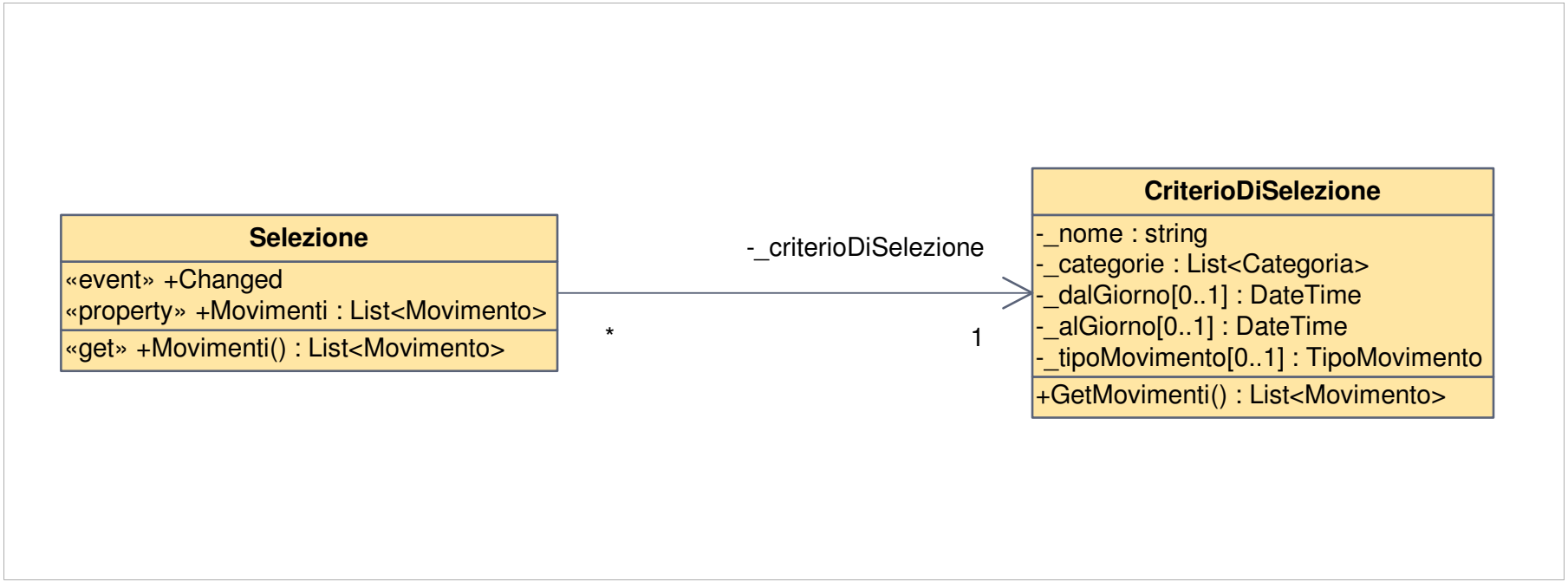


Classi di progettazione
Selezioni



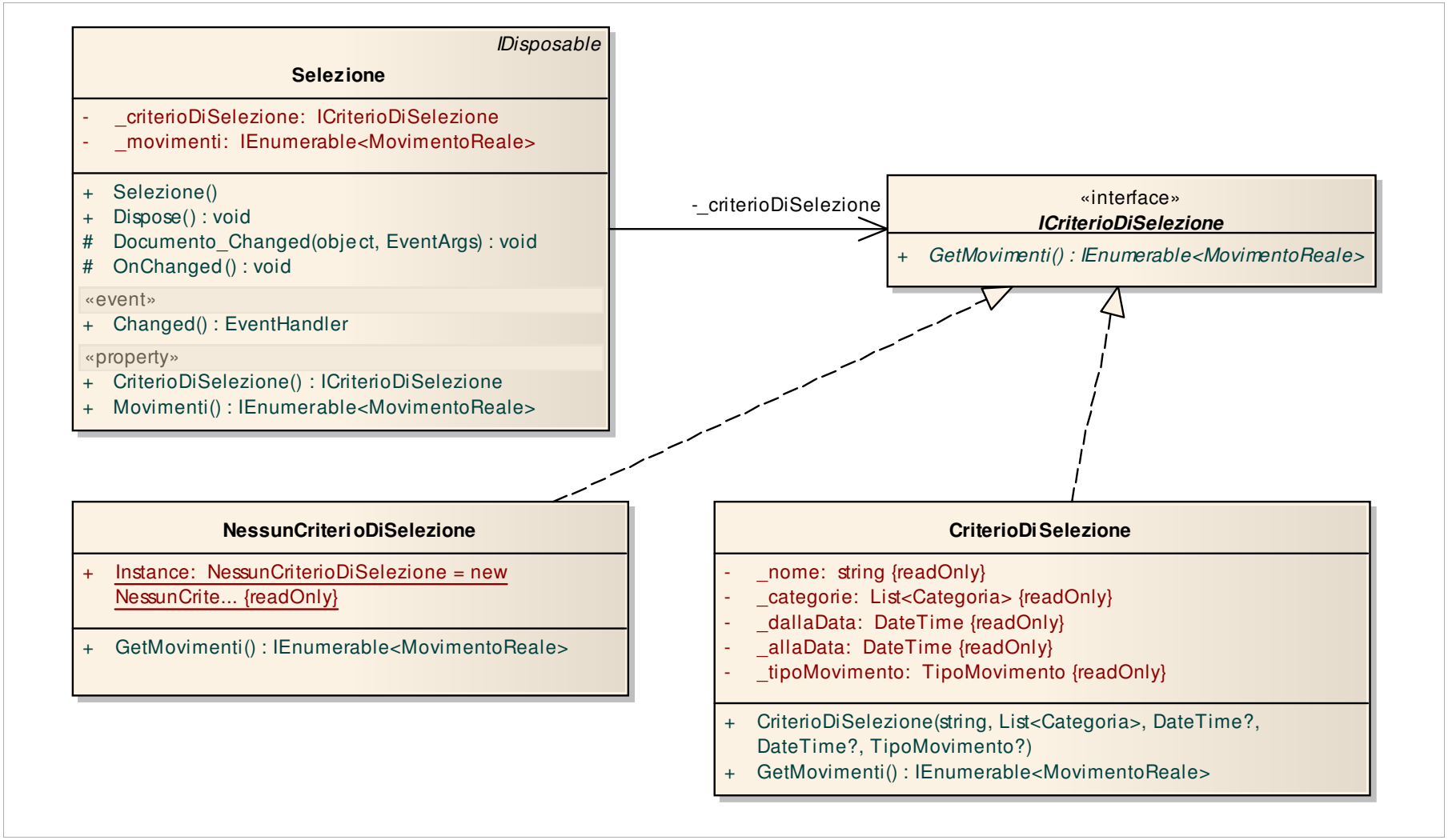
- Molte soluzioni possibili:
  - 1 sola classe
  - 1 gerarchia di decoratori
  - 2 classi (Selezione + CriterioDiSelezione)
  - 1 classe (Selezione) + 1 gerarchia di decoratori
- Per risolvere il problema delle selezioni “vive”:
  - registrazione all’evento Changed del Documento

Classi di progettazione Selezioni$_1$

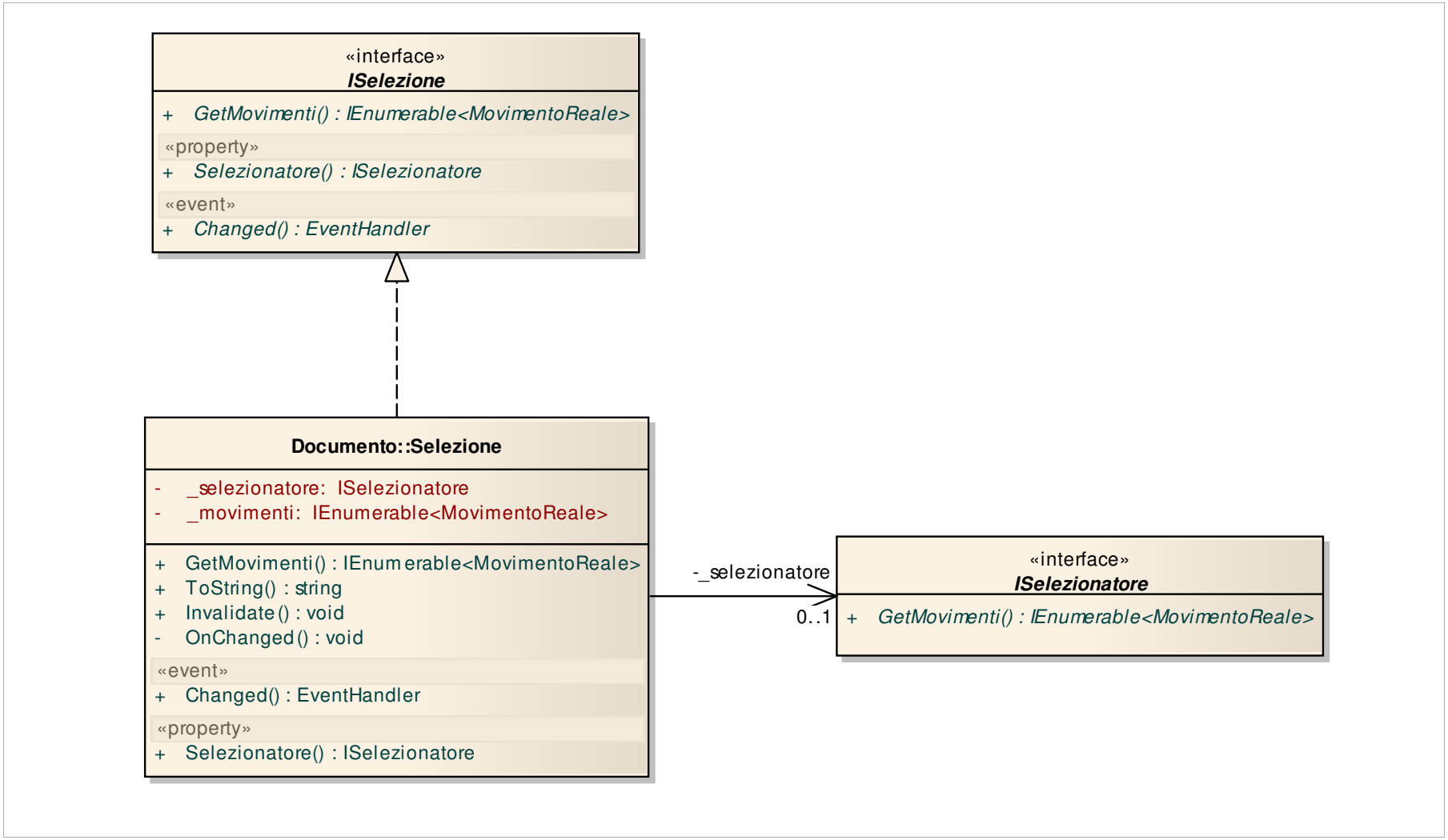


- Il criterio di selezione potrebbe essere realizzato con una query LINQ

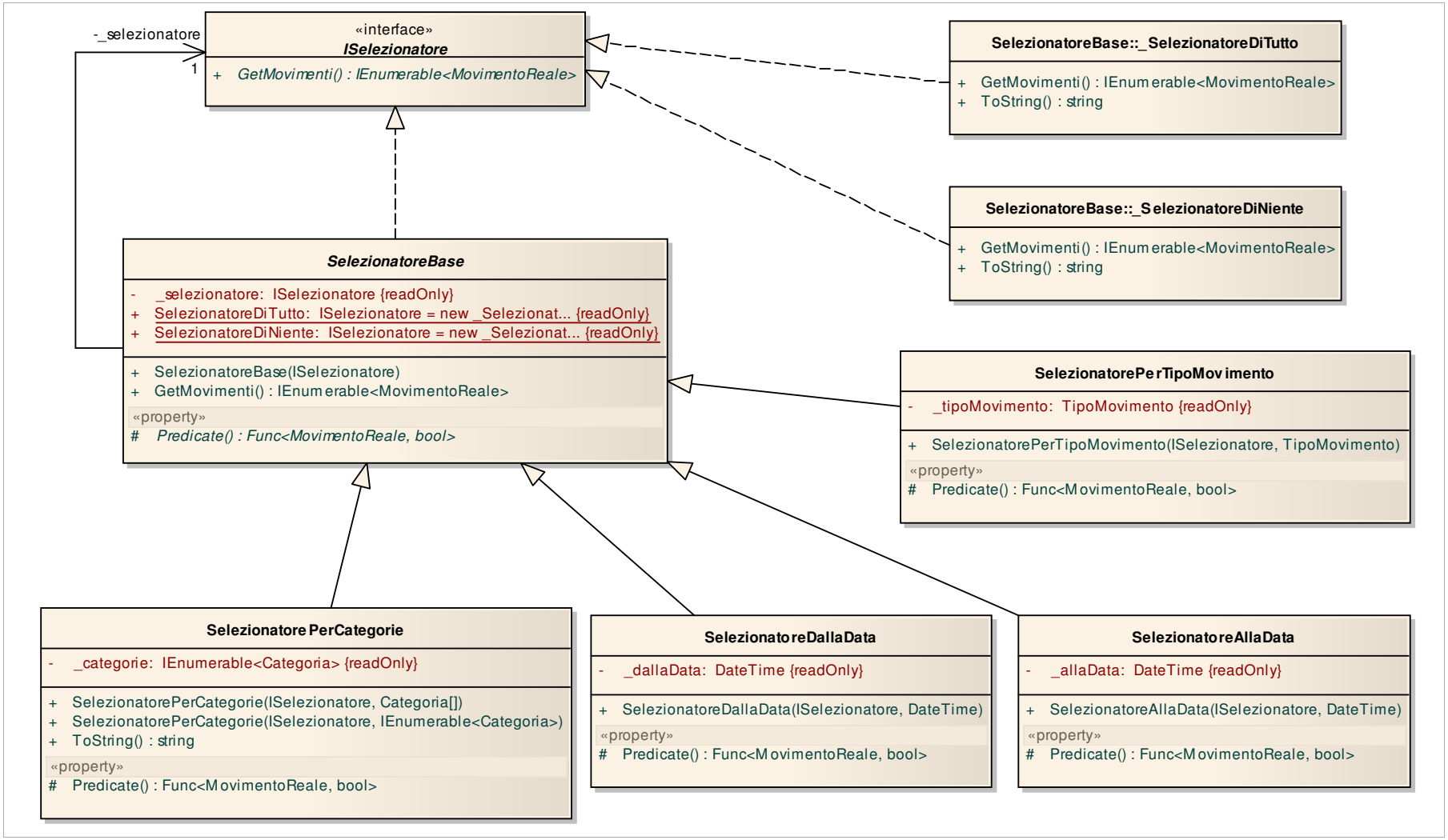
Classi di progettazione Selezioni$_2$



Classi di progettazione
Selezioni$_3$



Classi di progettazione Selezioni$_3$



Classi di progettazione
Calcoli$_1$