

DATA01

Build a LINQ-enabled Repository

Andrea Saltarello

Architect @ Managed Designs s.r.l.

andrea.saltarello@ugidotnet.org

<http://blogs.ugidotnet.org/pape>

<http://twitter.com/andysal74>



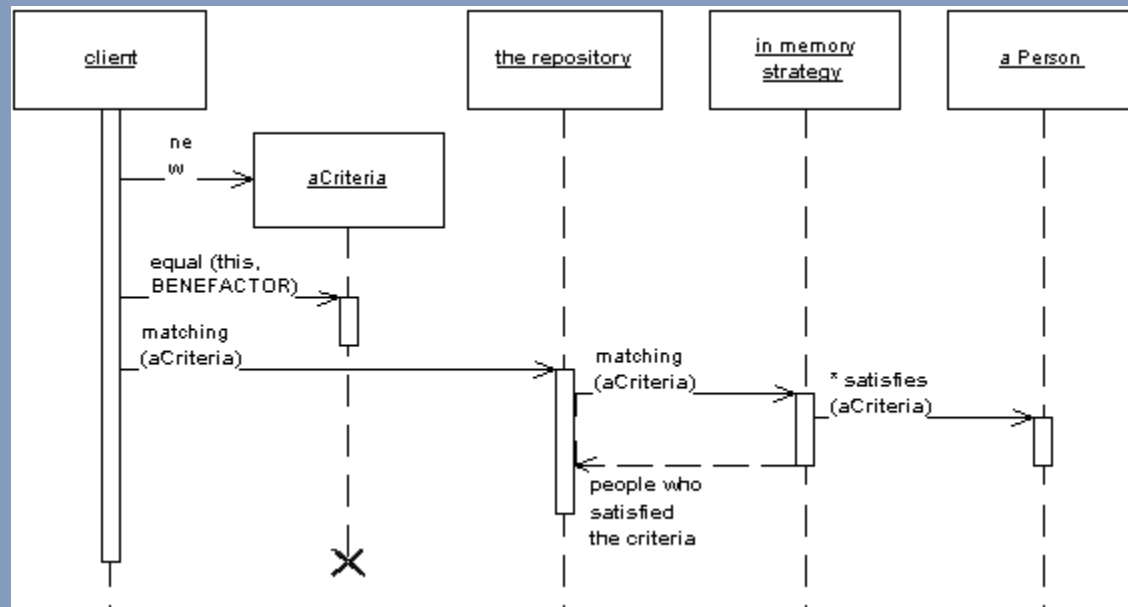
<http://creativecommons.org/licenses/by-nc-nd/2.5/>

Agenda

- Chi
- Perché
- Come
- Quisquilie IT
- Q&A

Repository pattern [P of EAA, 322]

Mediates between the domain and data mapping layers using a collection-like interface for accessing domain objects.



Repository pattern

A system with a complex domain model often benefits from a layer, such as the one provided by Data Mapper (165), that isolates domain objects from details of the database access code. In such systems it can be worthwhile to build another layer of abstraction over the mapping layer where query construction code is concentrated. This becomes more important when there are a large number of domain classes or heavy querying. In these cases particularly, adding this layer helps minimize duplicate query logic.

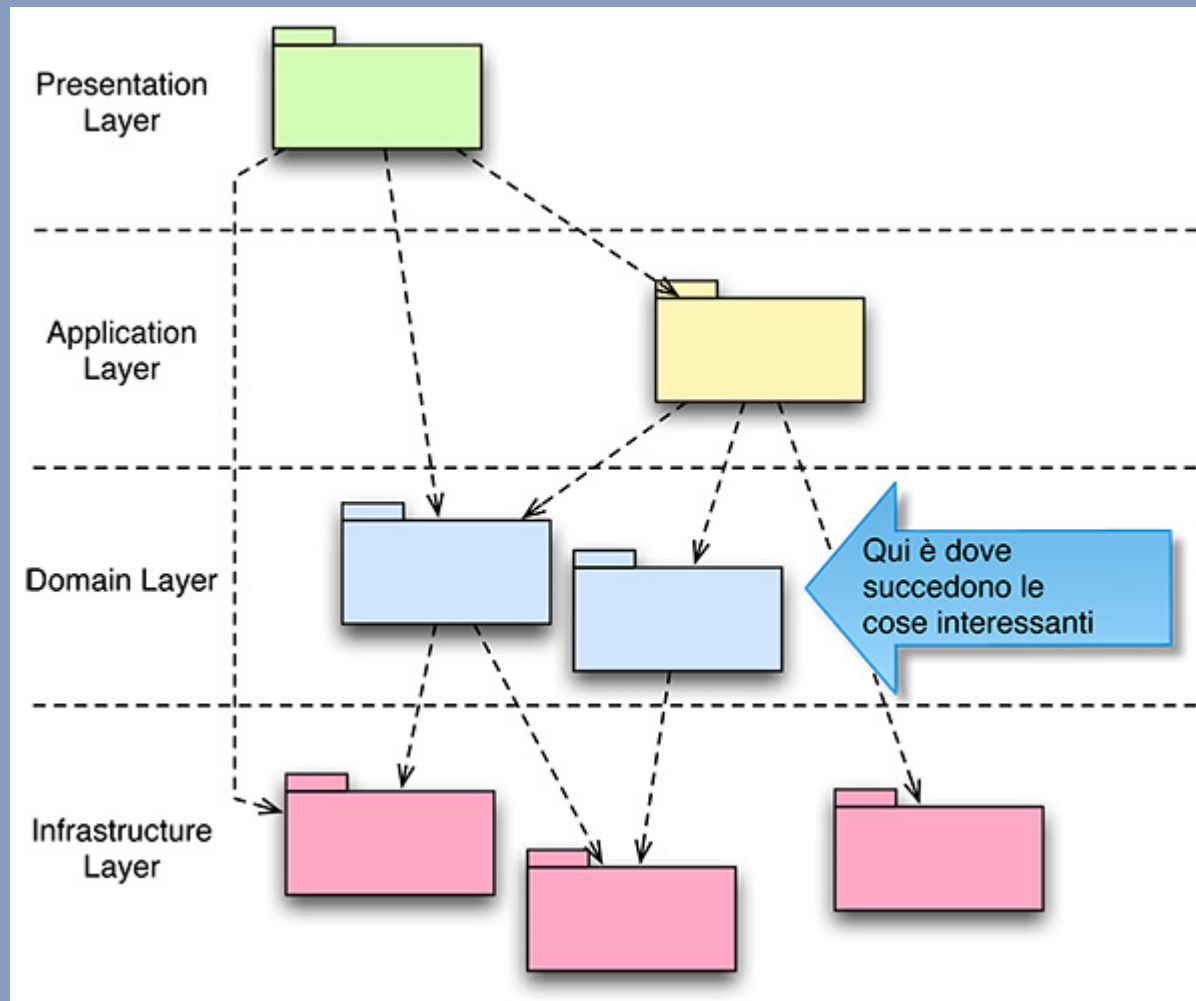
...

You can also find a good write-up of this pattern in [Domain Driven Design](#).

Repository: as easy as 1-2-3

1. Il Repository occorre per persistere un Domain Model
2. La «Bibbia» del Domain Model è [DDD]
3. E vediamo, 'sto DDD 😊

DDD Architecture



DDD Key Concepts (redux)

- Il Domain Layer contiene la *domain logic* ed è composto da
 - Model: **Entità** (identità e stato) e **Valori** (solo stato)
 - **Servizi**
- Entità e Valori a runtime formano dei grafi di oggetti. I grafi dotati di “dignità propria” sono chiamati **Aggregate** e il sistema (es: i **Repository**) si “impegna” a gestirli correttamente ed *atomicamente*
- Le istanze di entità/*aggregate* sono costruite da **Factory** (pattern **Builder** [GoF])

Da 0 ad Aggregate

- E' un insieme di elementi raggruppati in un'unità logica, quindi un grafo di oggetti
- Ha come radice l'entità principale dell'aggregato
- La radice è l'unico elemento che può essere referenziato fuori dai confini dell'aggregato
- Non è possibile agire direttamente sugli elementi senza passare dalla radice dell'aggregato
- L'aggregate ha la responsabilità di implementare la propria logica

Domain Model



Repository pattern (Reloaded)

Mediates between the domain and data mapping layers using a collection-like interface for accessing domain objects.

Ricapitolando:

- Interfaccia “collection like”
- Gestisce la persistenza degli Aggregate
- LINQ! (siamo dei buongustai 😊)

IRepository<T>



Repository++

Quisquilie IT:

- Code Contracts
- Repository <3 IoC

Demo code: NSK

Progetto open source (licenza CPL)
disponibile su CodePlex:

<http://nsk.codeplex.com>

Bibliografia

**[DDD] Domain Driven Design, Eric Evans,
Addison-Wesley**

**[P of EAA] Pattern of Enterprise Application
Architecture, Martin Fowler, Addison-
Wesley**

Contatti

- Mail: andrea.saltarello@ugidotnet.org
- Twitter: <http://twitter.com/andysal74>
- Blog: <http://blogs.ugidotnet.org/pape>
- Lavoro: <http://www.manageddesigns.it>

Slides e materiale

Nei prossimi giorni su

<http://www.communitydays.it/>