



Architecting Layered Applications

Speaker: Giancarlo Sudano



GUISA
Gruppo Utenti Italiani
Solution Architect

About me:



Giancarlo Sudano (*alias janky*)

- Software Architect in Objectway
- Co-fondatore di GUISA
www.guisa.org
- Blog su Ugidotnet:
<http://blogs.ugidotnet.org/janky>
- Email:
giancarlo.sudano@gmail.com
giancarlo.sudano@objectway.it

2a serie Webcast: "Aspire Architect"

- **Domain Driven Design: Overview**
(23/01/2007 11:00) [[Giancarlo Sudano](#)]
- **UML Reloaded**
(30/01/2007 : 14:30) [[Riccardo Golia](#)]
- **Design Principles**
(06/02/2007 14:30) [[Riccardo Golia](#)]
- **Architecting Layered Applications**
(13/02/2007 : 14:30) [[Giancarlo Sudano](#)]
- **Software Architecture: oltre il design**
(20/02/2007 14:30) [[Lorenzo Barbieri](#)]
- **Service Oriented != Object Oriented**
(22/02/2007 11:00) [[Andrea Saltarello](#)]
- **Software Architecture: soluzioni del mondo reale**
(22/02/2007 14.30) [[Andrea Saltarello](#)]

Webcast outline

- Premesse, Architettura e Stile Architeturale
- Layered Architecture
- 3-Layer
- 4-Layer
- N-Layer
- N-Layer Esempio
- Distribuzione su Tier

Architettura...Stile...



Software Architecture

- E' la naturale evoluzione dell'Ingegneria Software
- In principio, nella letteratura informatica veniva definita: programming in the large
- Gli architetti software sono solitamente Ingegneri Software con una certa esperienza



Software Architecture: Paradigm Shift

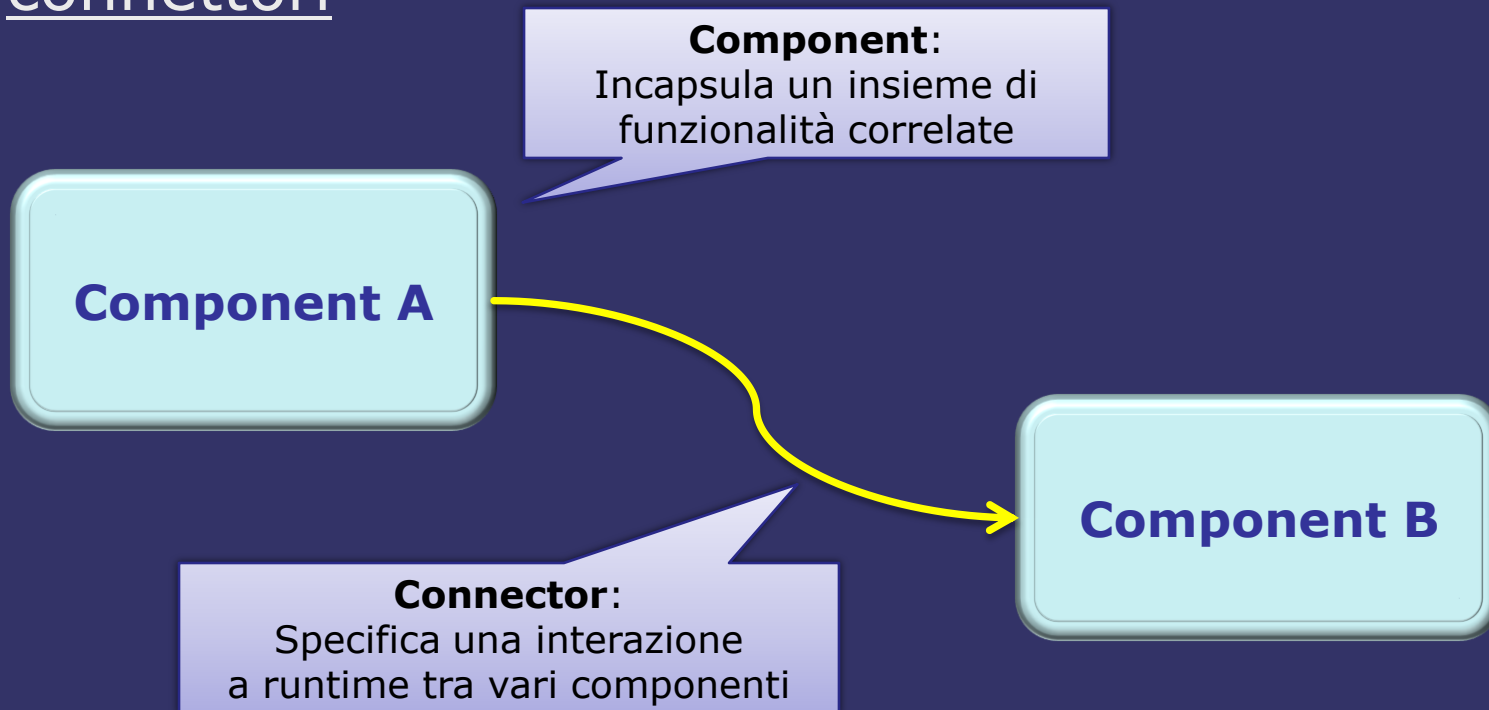
OOP ancora applicabile ma:

- Su una scala differente
- Su livelli differenti di astrazione



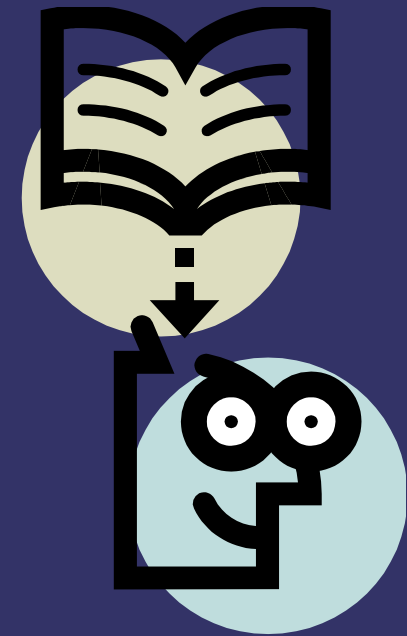
Architettura: Una vista dall'alto

- Le architetture software presentano delle viste di un sistema mediante l'uso di componenti e connettori



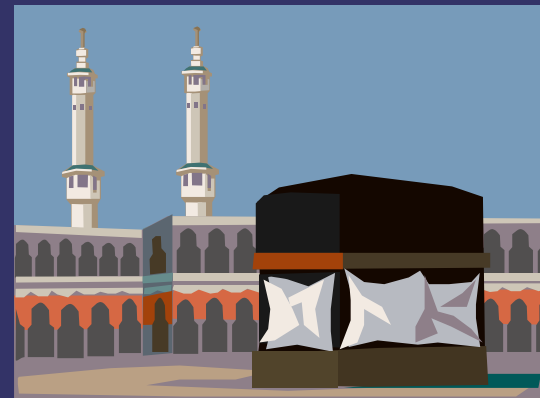
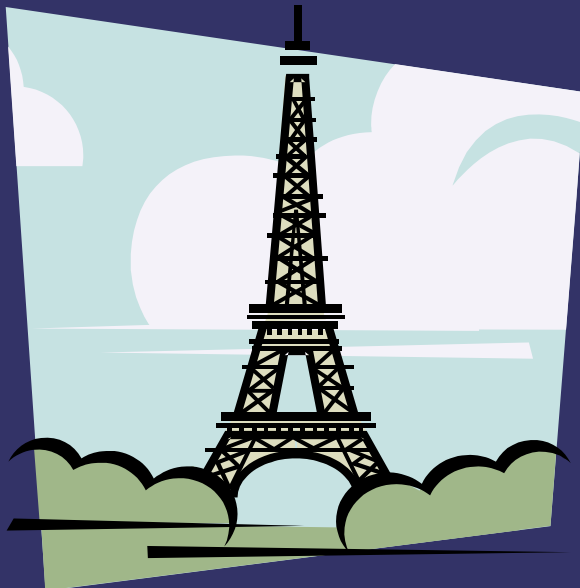
Fare Architettura significa...

- ...specificare/descrivere un sistema, che una volta costruito, esibirà le proprietà richieste e soddisferà di requisiti iniziali.
- ...creare la descrizione di un sistema allo scopo di valutazione
- ...fare un piano per una successiva implementazione



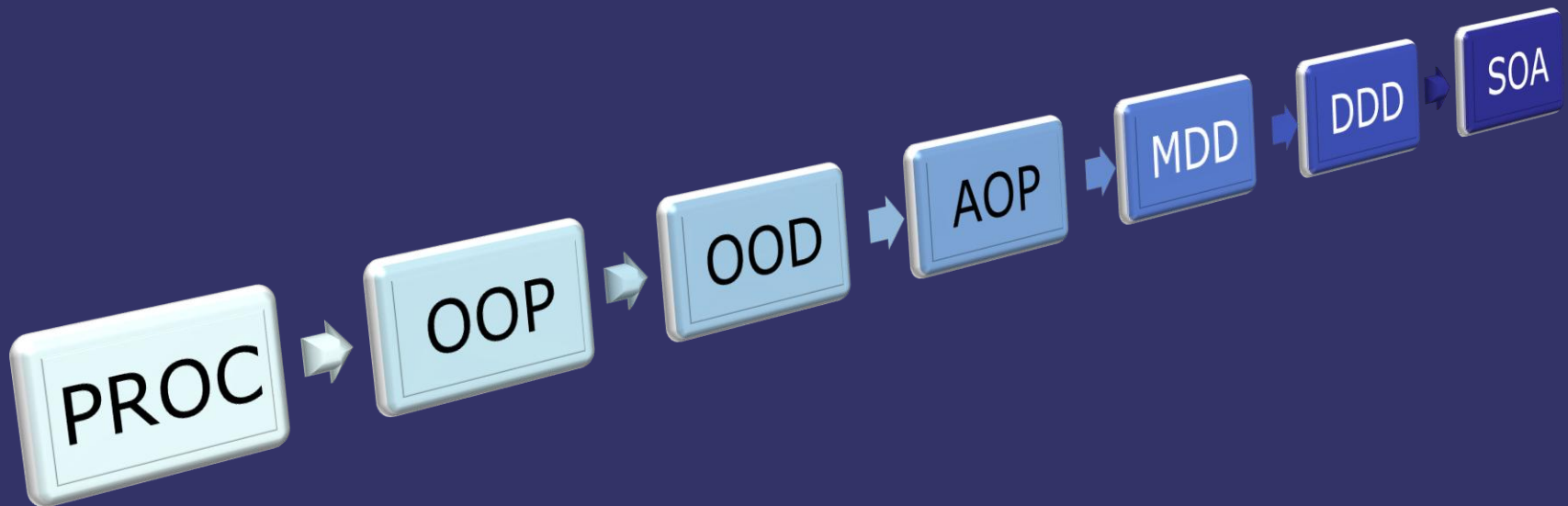
L'architettura software...

- ...significa anche "Stile".



L'architettura software...

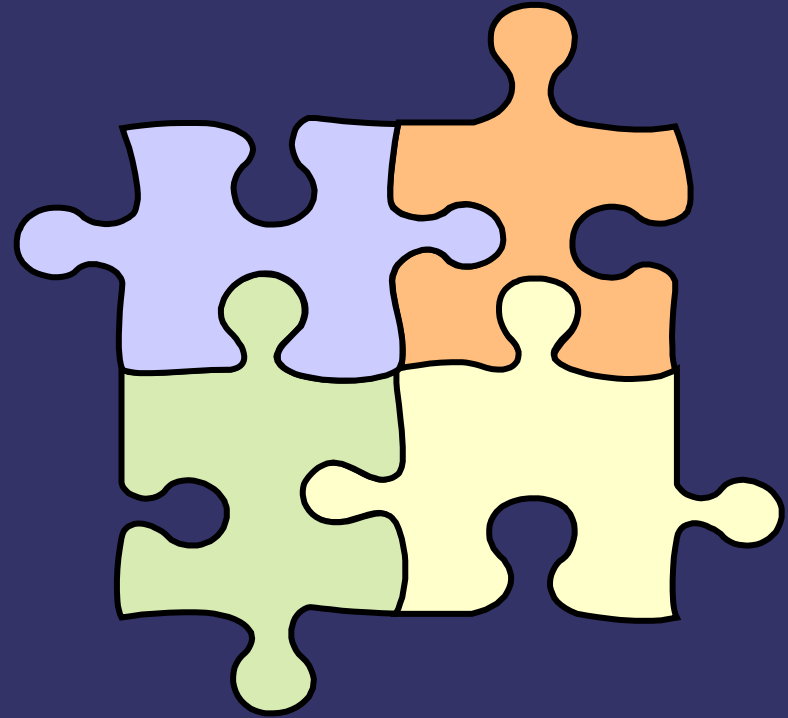
- ...sta cambiando!



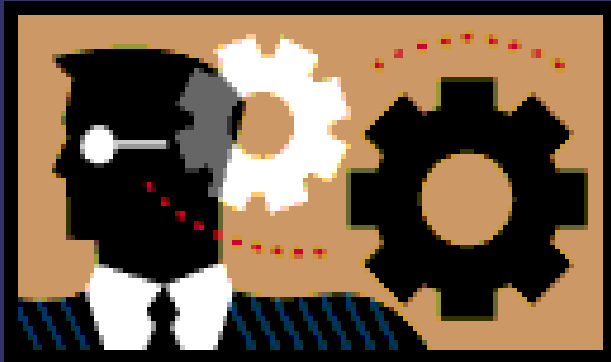
Layered Applications

Gestire la Complessità

- **Contesto:**
Stiamo lavorando su un sistema piuttosto complesso, e vogliamo gestire la complessità sfruttando la "decomposizione"...



Gestire la Complessità



- **Problema:**
Come possiamo strutturare una applicazione per supportare dei requisiti non funzionali come

Manutenibilità

Riusabilità

Estendibilità

Scalabilità

Sicurezza

...

Gestire la Complessità

- **Soluzione:**
Componiamo il nostro sistema in insiemi di **Layer**.

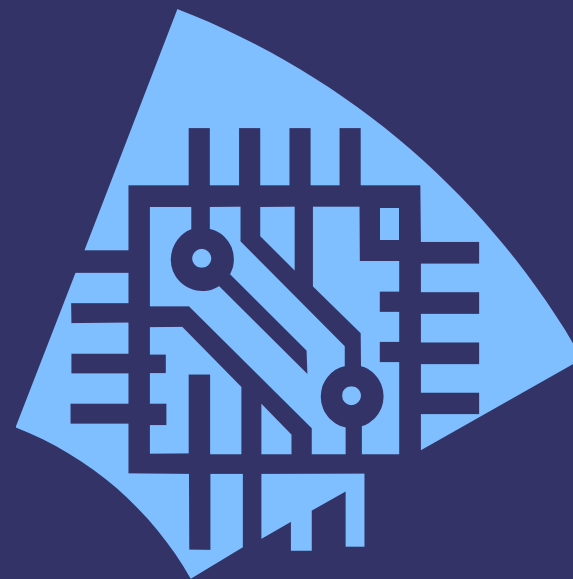
Ogni Layer:

- deve essere coeso
- deve essere debolmente accoppiato con i layer sottostanti



Componentizzazione

- Modellare componenti riutilizzabili da più applicazioni
- Componenti devono essere coesi
- Componenti devono avere basso accoppiamento tra di loro
- Team indipendenti dovrebbero poter lavorare su parti della soluzione con dipendenze minimali sugli altri team



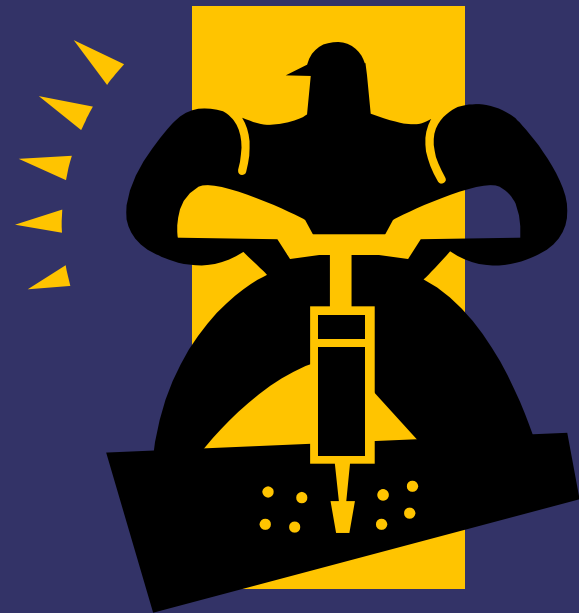
Complessità sempre crescente?

- Aggiungere tipi di componenti ai layer del nostro sistema, non è il solo modo...
- Si può strutturare una architettura disegnando ulteriori livelli

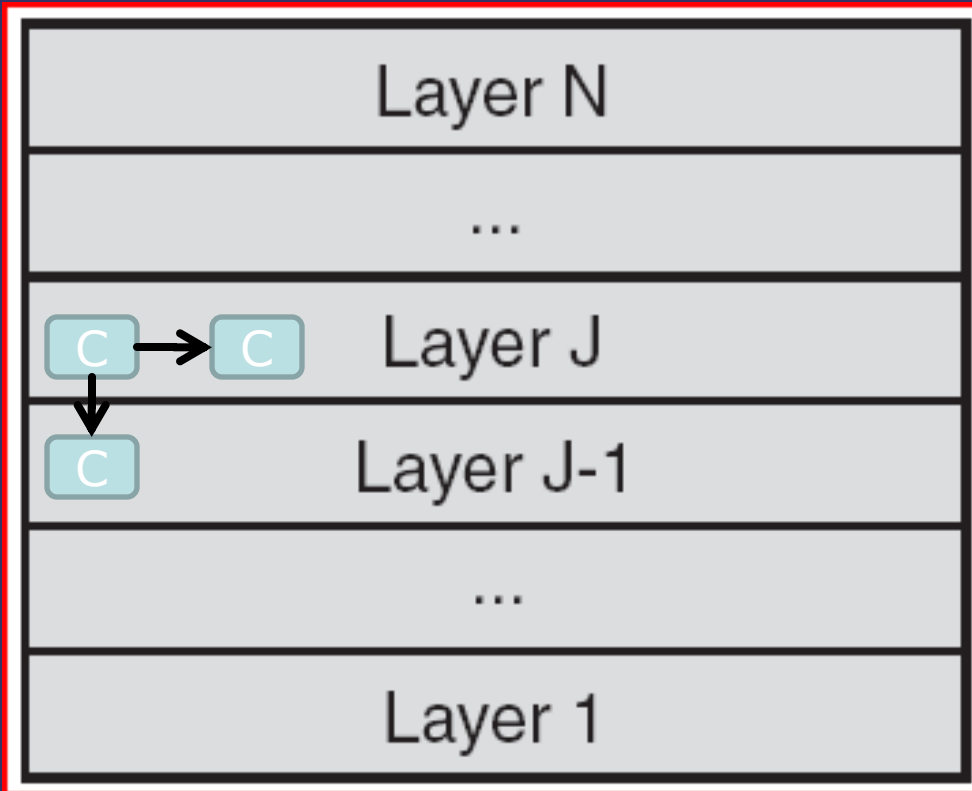


Testabilità

- Per garantire Performance, e Affidabilità, la nostra solution deve essere Testabile!



Layering



- Componenti di un layer interagiscono con componenti dello stesso layer o di layer sottostanti

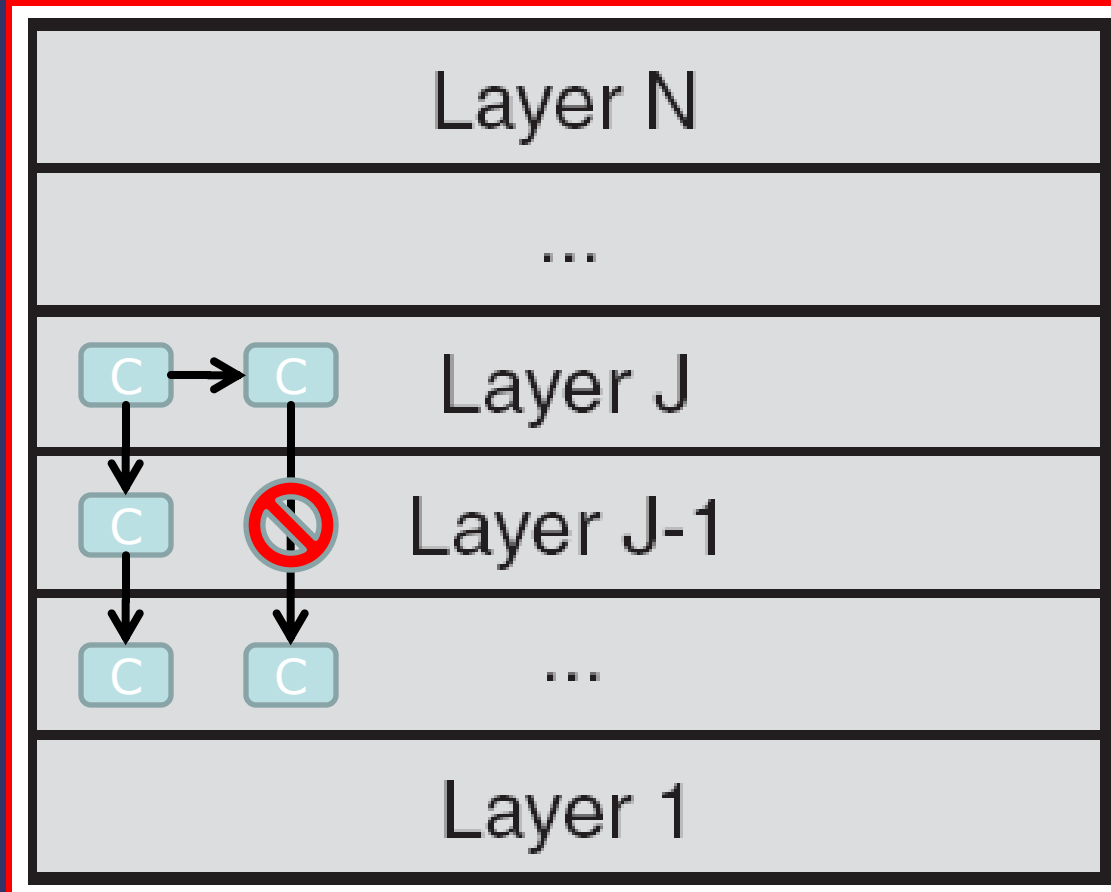
Occhio alla differenza...

Tra Layer e Tier

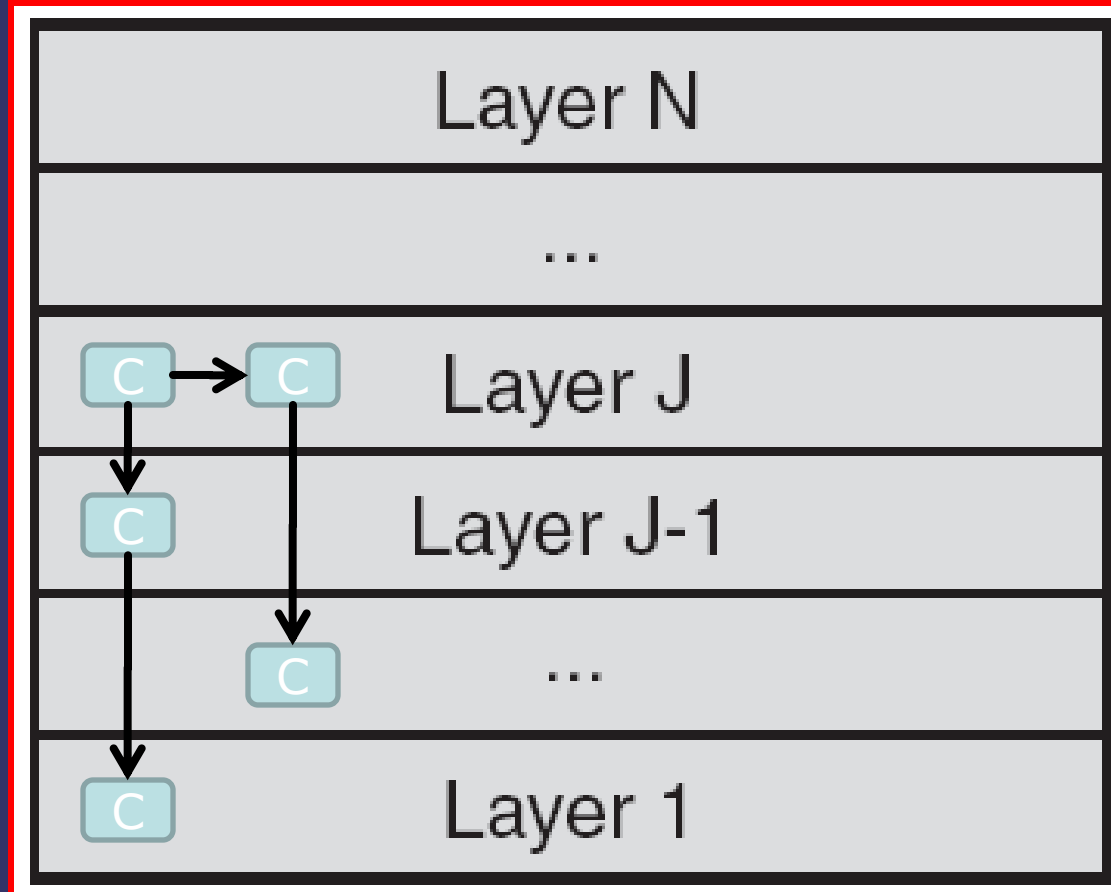
Layer: Raggruppamento di componenti secondo un criterio funzionale

Tier: Strato Fisico (un Server o un Cluster di Server) su cui può essere installato uno o più layer applicativi

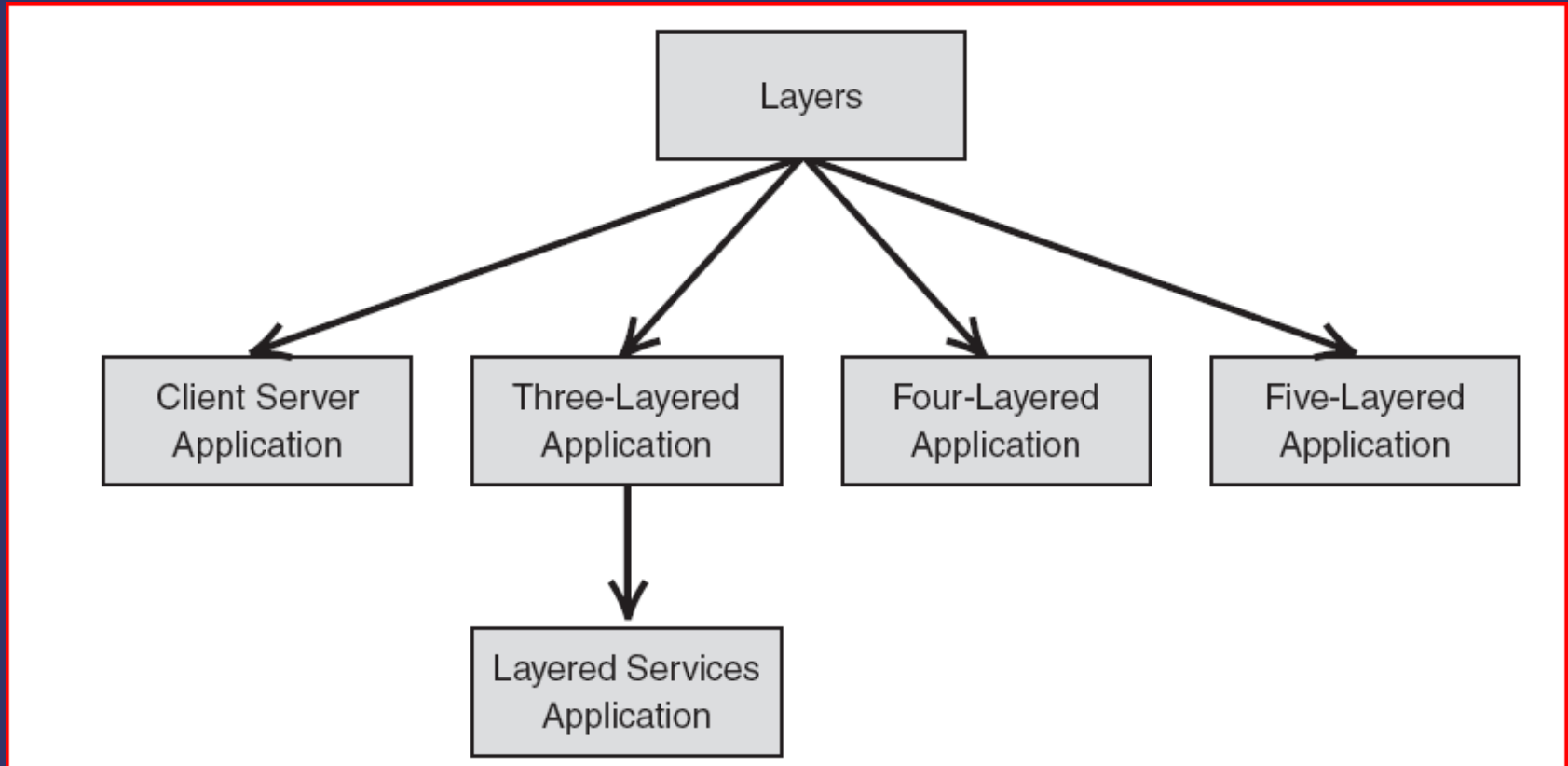
Strict Layered Approach



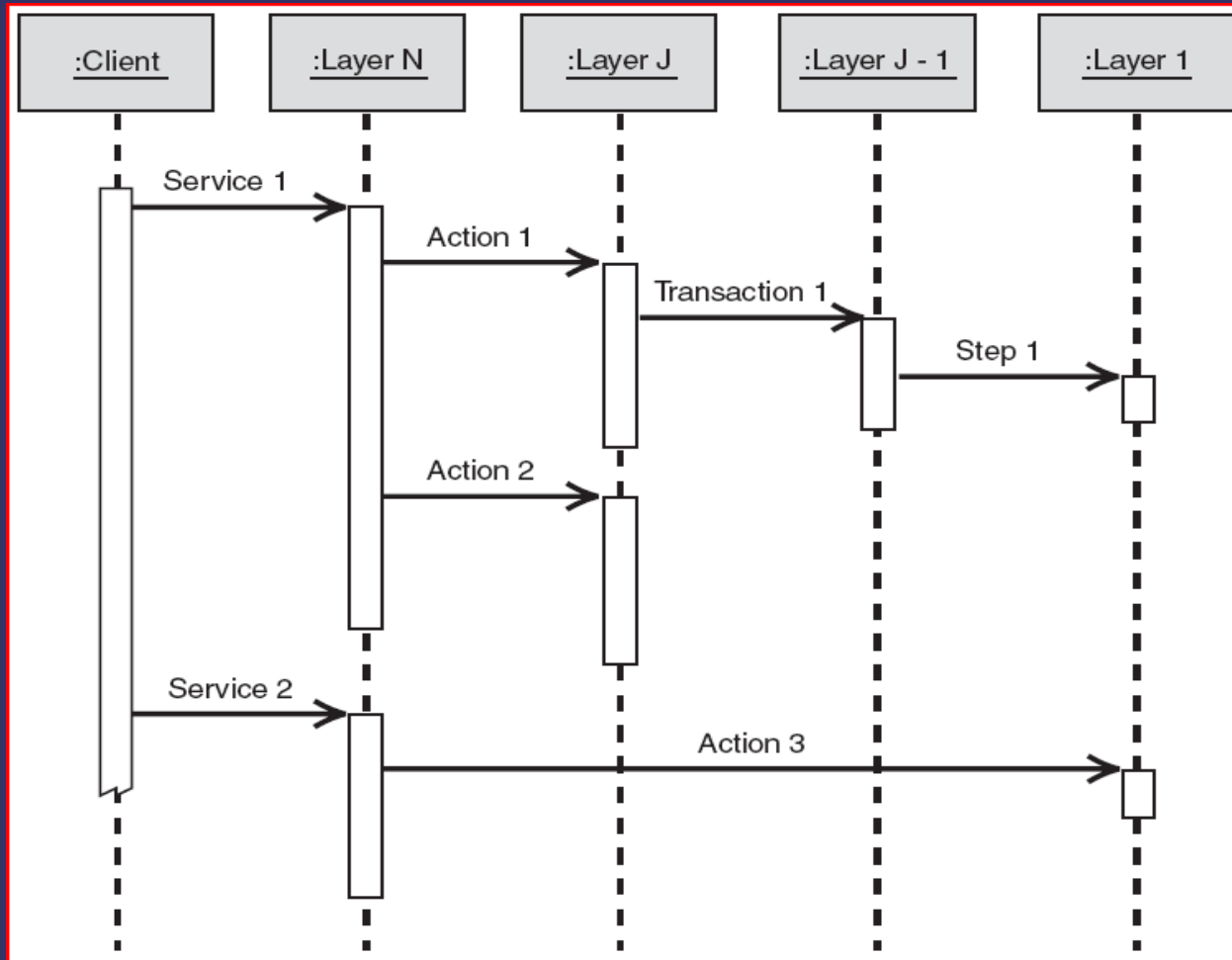
Relaxed Layered Approach



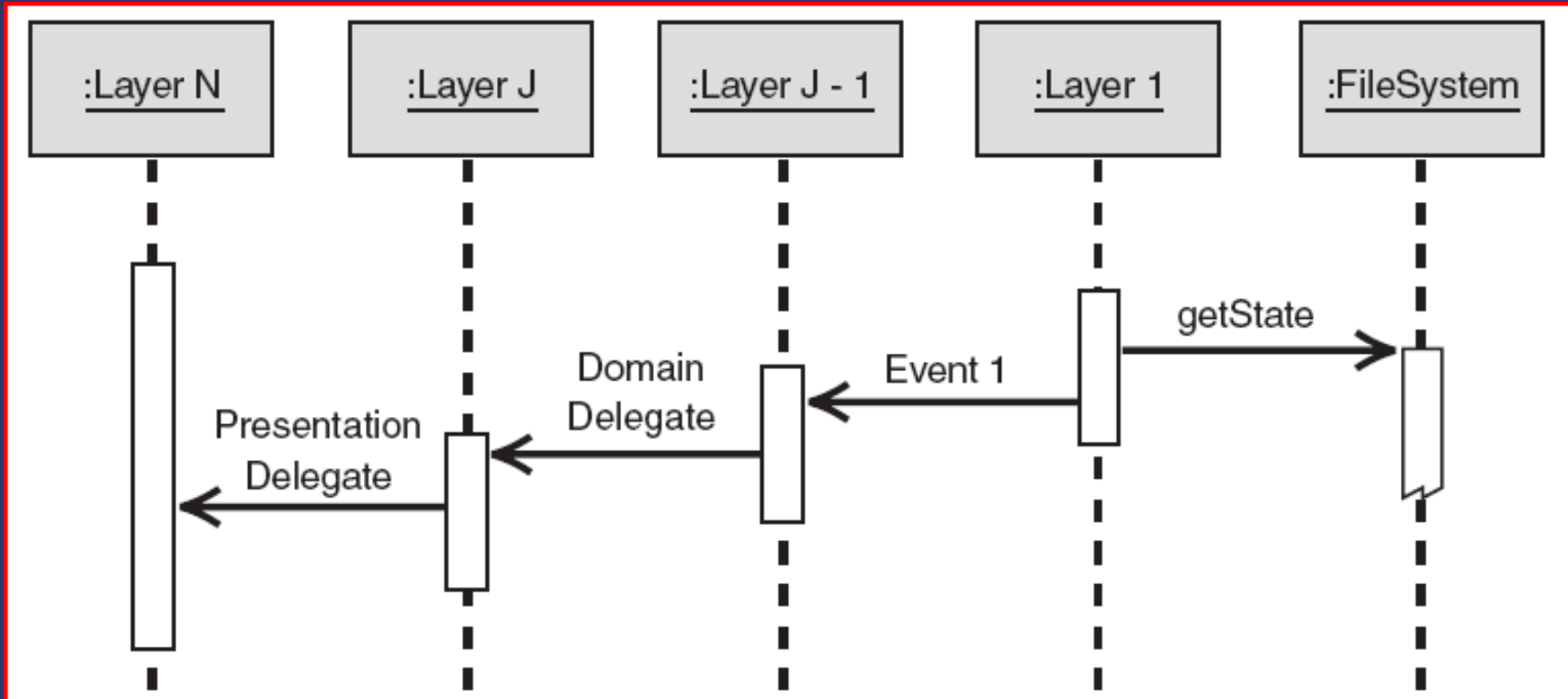
Layers Refining



Layer: Interazione Top down



Layer: Interazione Bottom Up



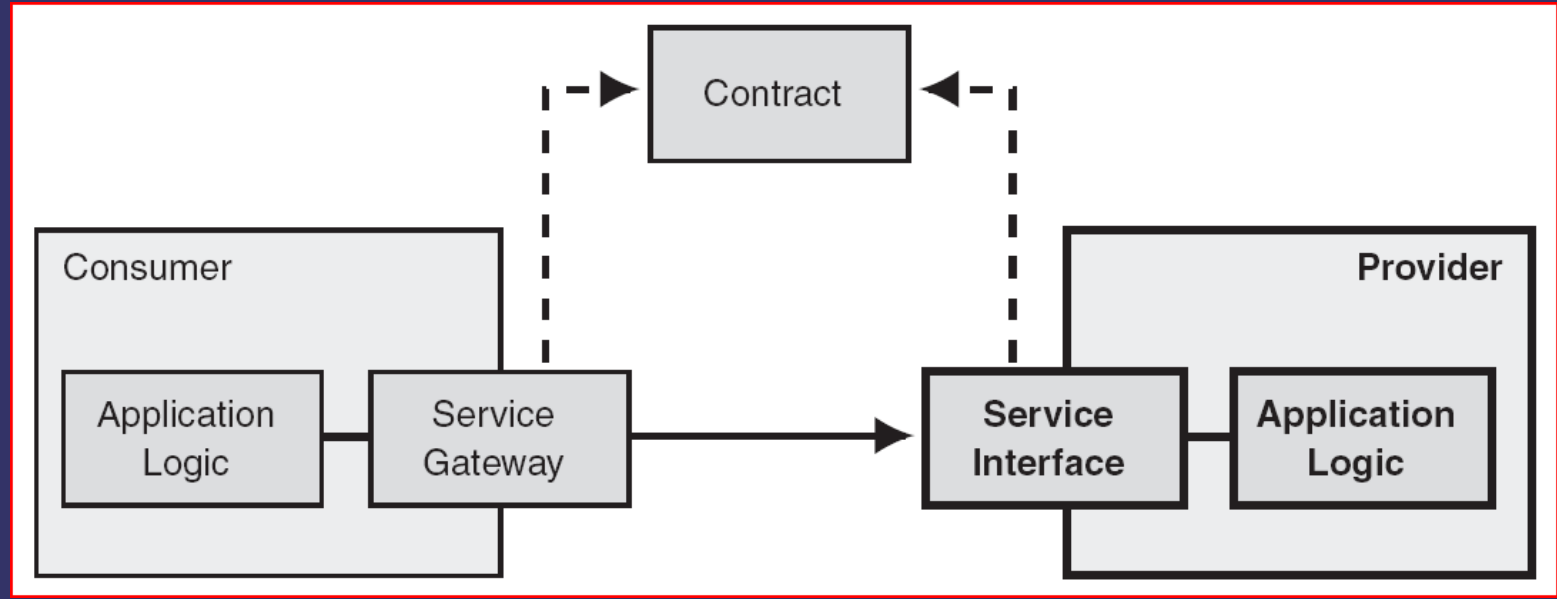
Come disegnare Layered Applications

- Usare un criteri ben definiti per raggruppare i componenti in layer
- Definire una interfaccia di comunicazione per ogni livello
- Per comunicare con i layer di più alto livello utilizzare
 - Callback
 - Asynchronous messaging
 - Events
- Eccezioni
 - Gestire le eccezioni al più basso livello possibile
 - Evitare l'esposizione di dettagli di basso livello a layer di alto livello
 - Trasformare eccezioni di basso livello in eccezioni di alto livello
- Usare Design Pattern (*)

Alcuni Pattern

- Service Gateway
- Layer SuperType
- Data Transfer Object
- Layer Façade
- Adapted, Bridge, Strategy

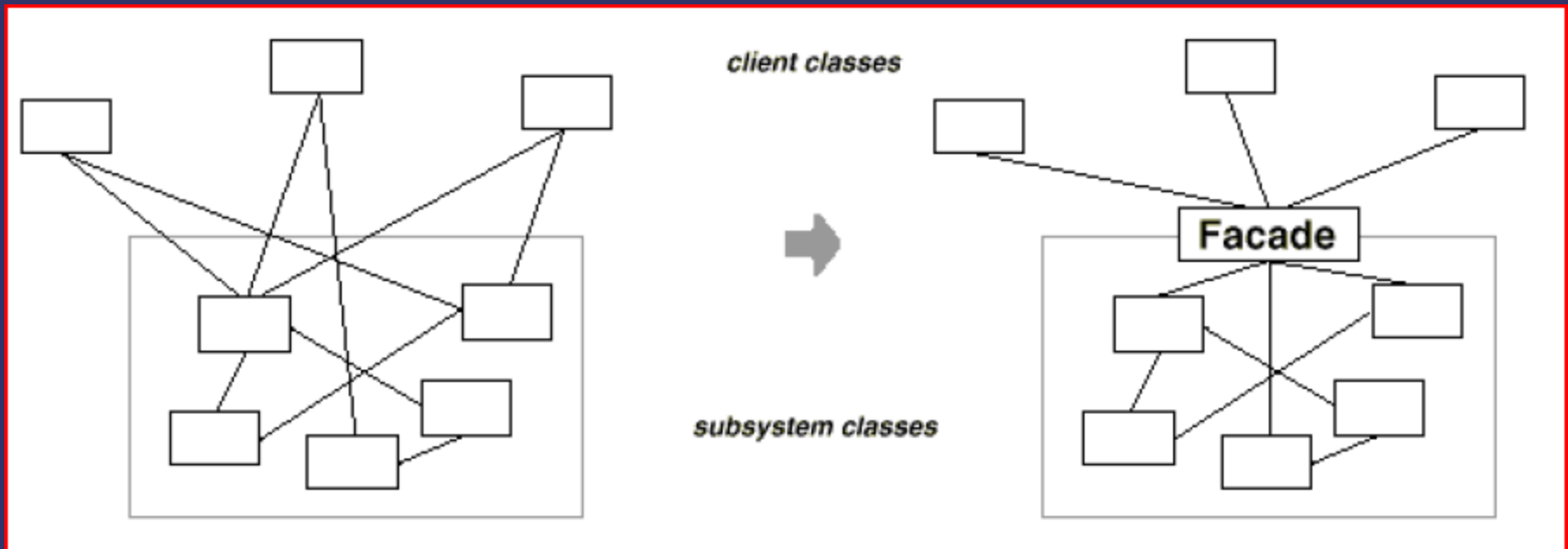
Service Gateway



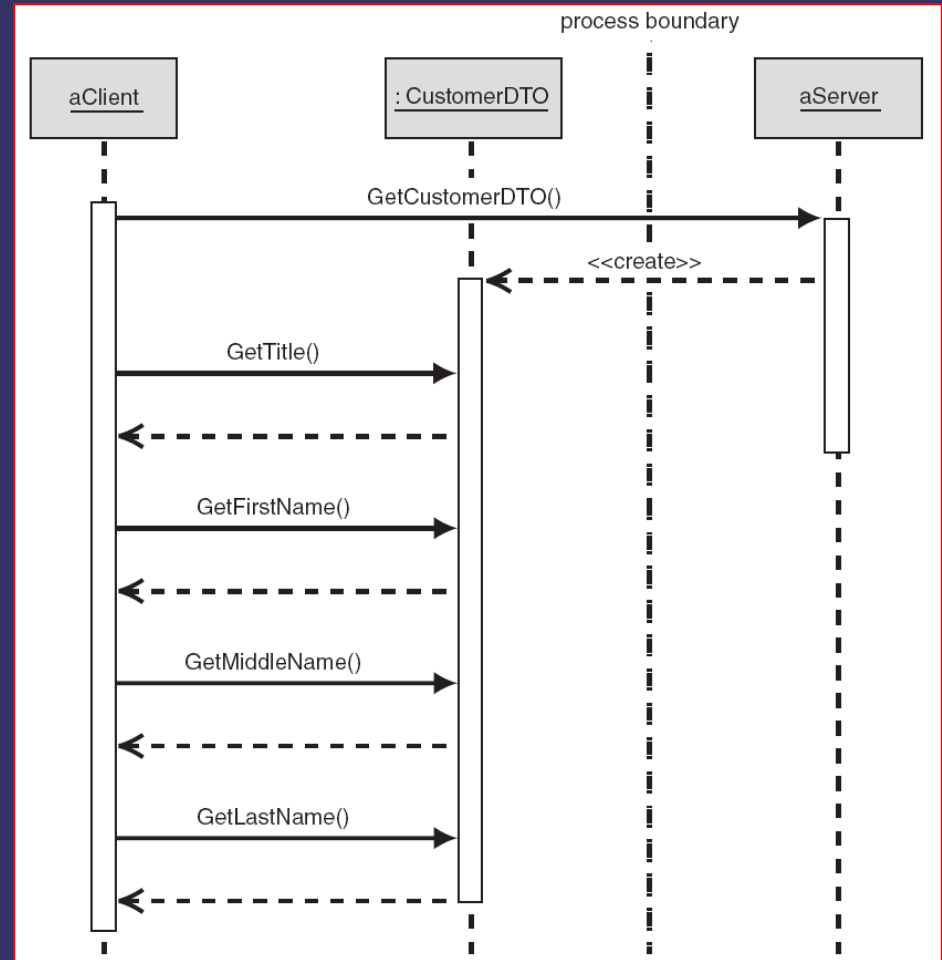
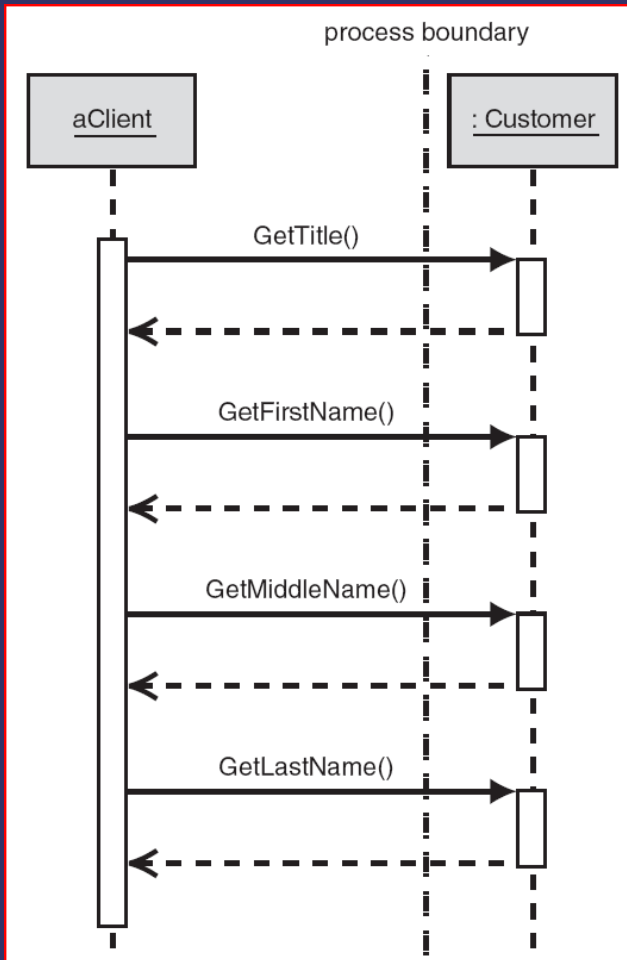
Layer Supertype

- Active Record (Castle Project)
- CLSA
- WebForm, WinForm
- ...

Layer Façade



Data Transfer Object



Vantaggi del Layering

- Basso accoppiamento favorisce al Manutenibilità
- Isolamento dei layer favorisce la Testabilità
- Riutilizzo dei layer
- Distribuzione del lavoro (layer boundaries)

- Distribuzione dei layer su Tier
 - Aumentare le Performance
 - Scalabilità: Scale out
 - Fault tolerance



Svantaggi

- Valutare bene le performance per la comunicazione tra layer

Ma soprattutto

- ...l'astrazione costa!
 - Skill
 - Complessità del codice
 - Propagazione verso l'alto delle modifiche ai layer inferiori
 - Codice aggiuntivo per interfacce

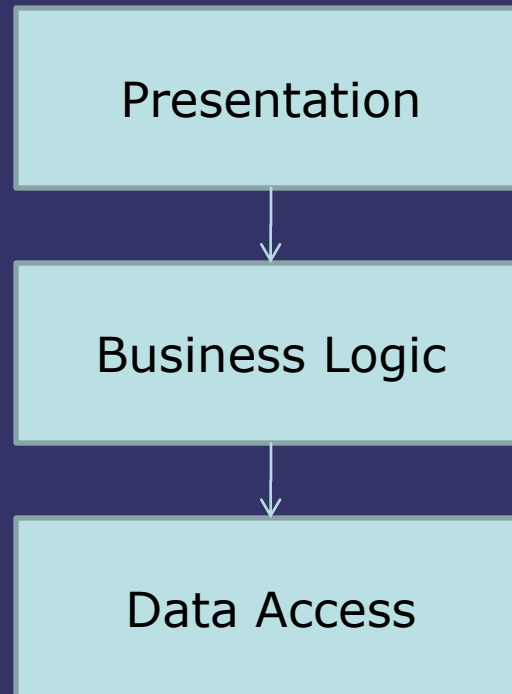


3-Layer Architecture

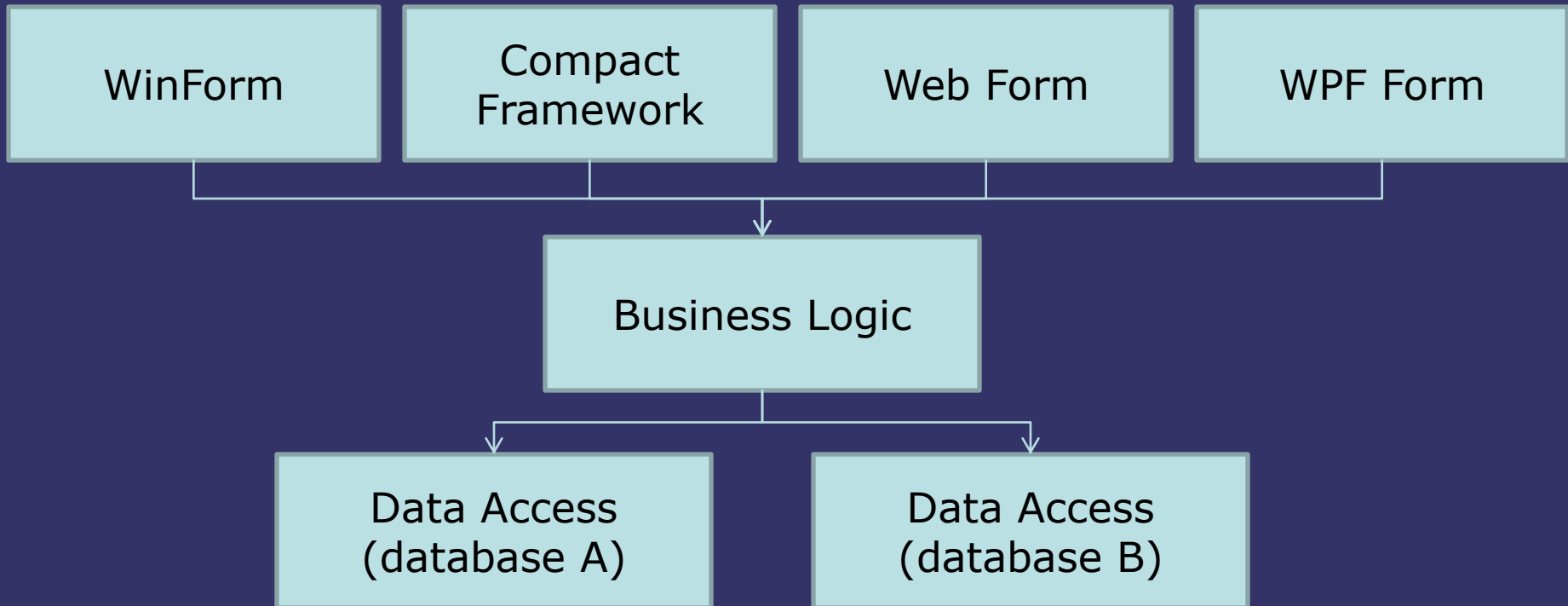
Contesto

- Isolare il codice dalla interfaccia utente
- Isolare il codice di accesso al database

Soluzione



Soluzione

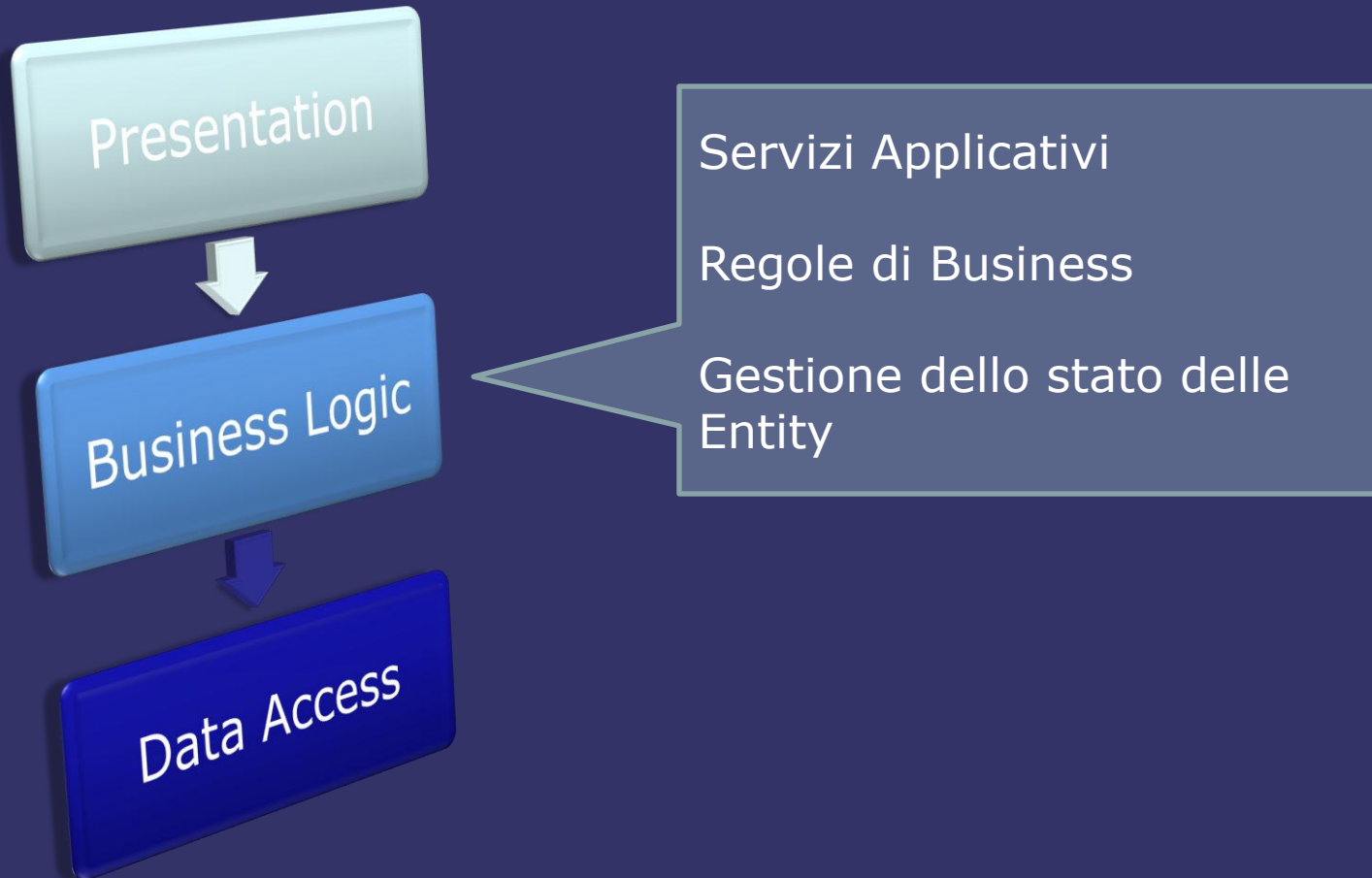


Responsabilità

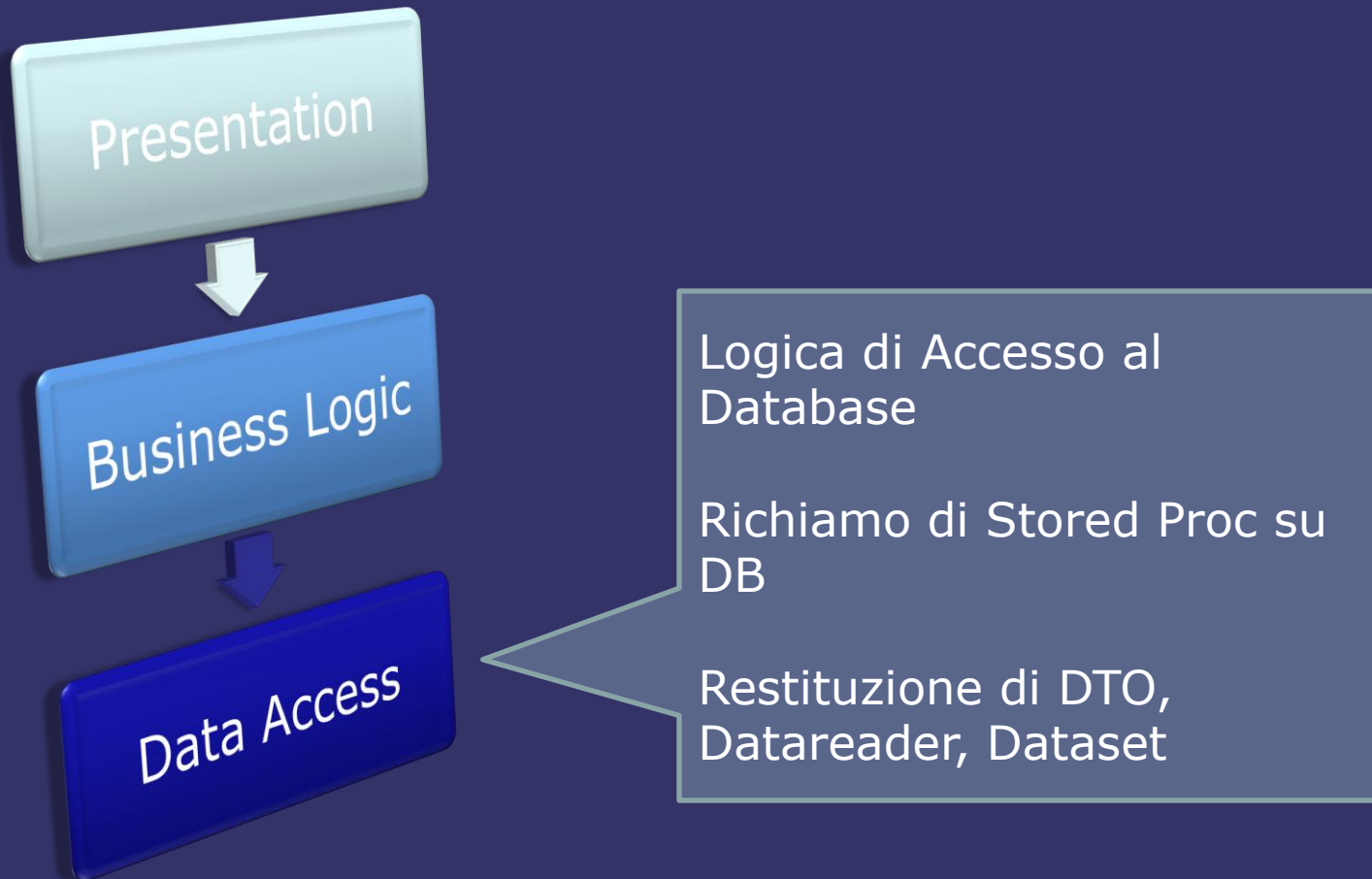


Costruzione delle View
Bindind View/Entity
Tiene solitamente lo stato dell'applicazione
Tiene la logica di flusso tra View

Responsabilità



Responsabilità



Svantaggi

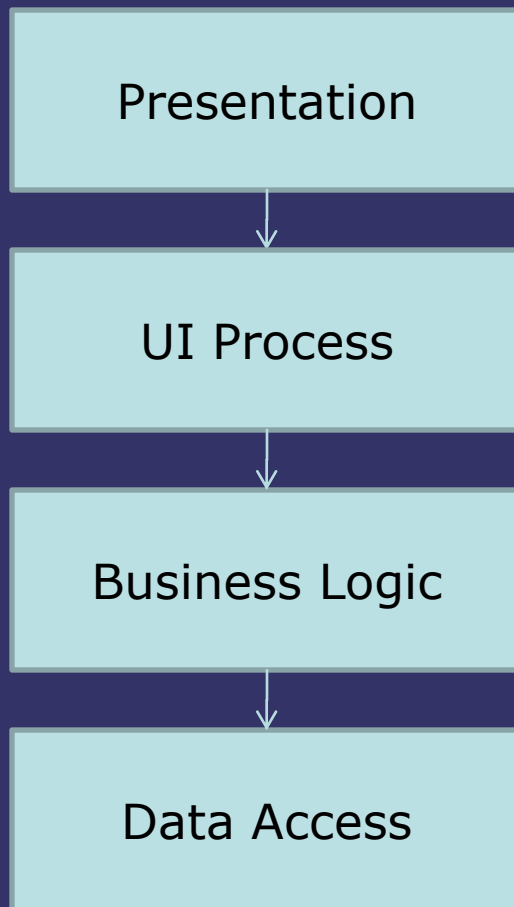
- Logica di Flusso, Binding Accoppiata alla UI
- ...

4-Layer Architecture

Contesto

- Riutilizzo più pesante del codice tra una Presentation Layer ed un altro
- Migliorare le capacità di Test delle View e della logica di Flusso

Soluzione



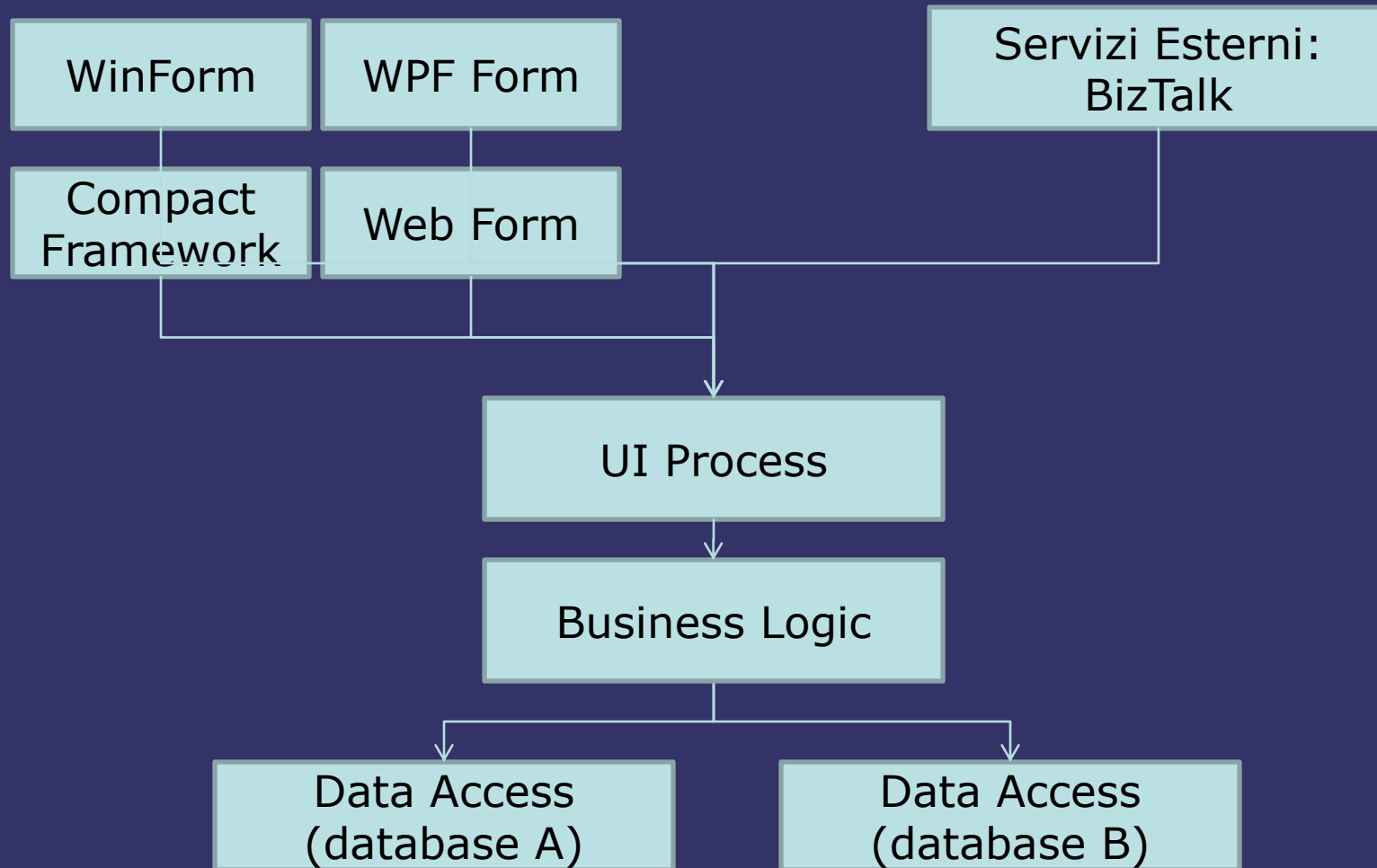
Coordina le attività dell'applicazione.

Non contiene logica di business.

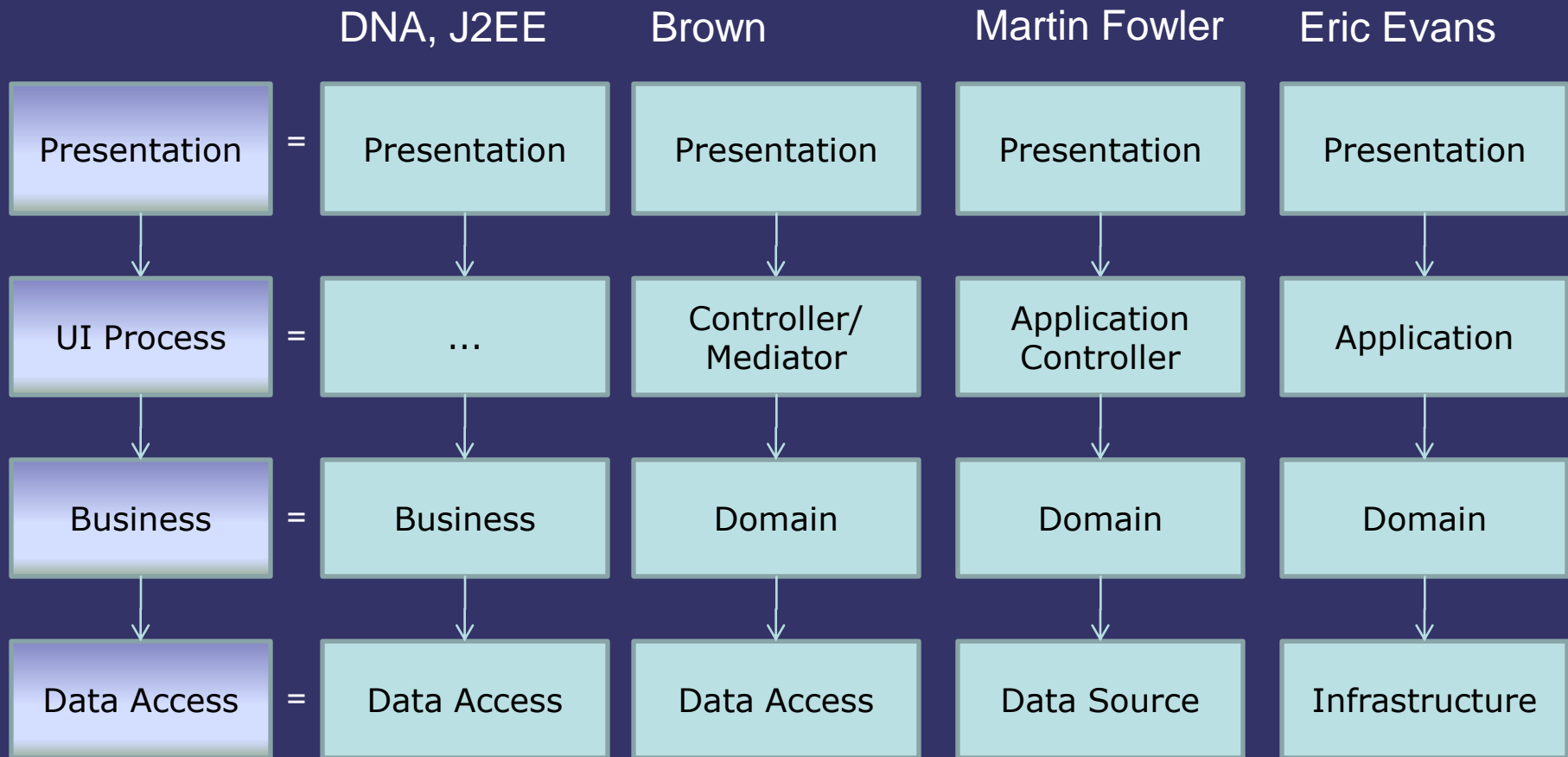
Non gestisce lo stato delle Entity, ma gestisce lo stato dell'applicazione e dei progresso dei task

Può eseguire Processi di Business e Servizi in seguito a comandi dalla UI o a Eventi Esterni

Soluzione



Confusioni sui nomi di layer?

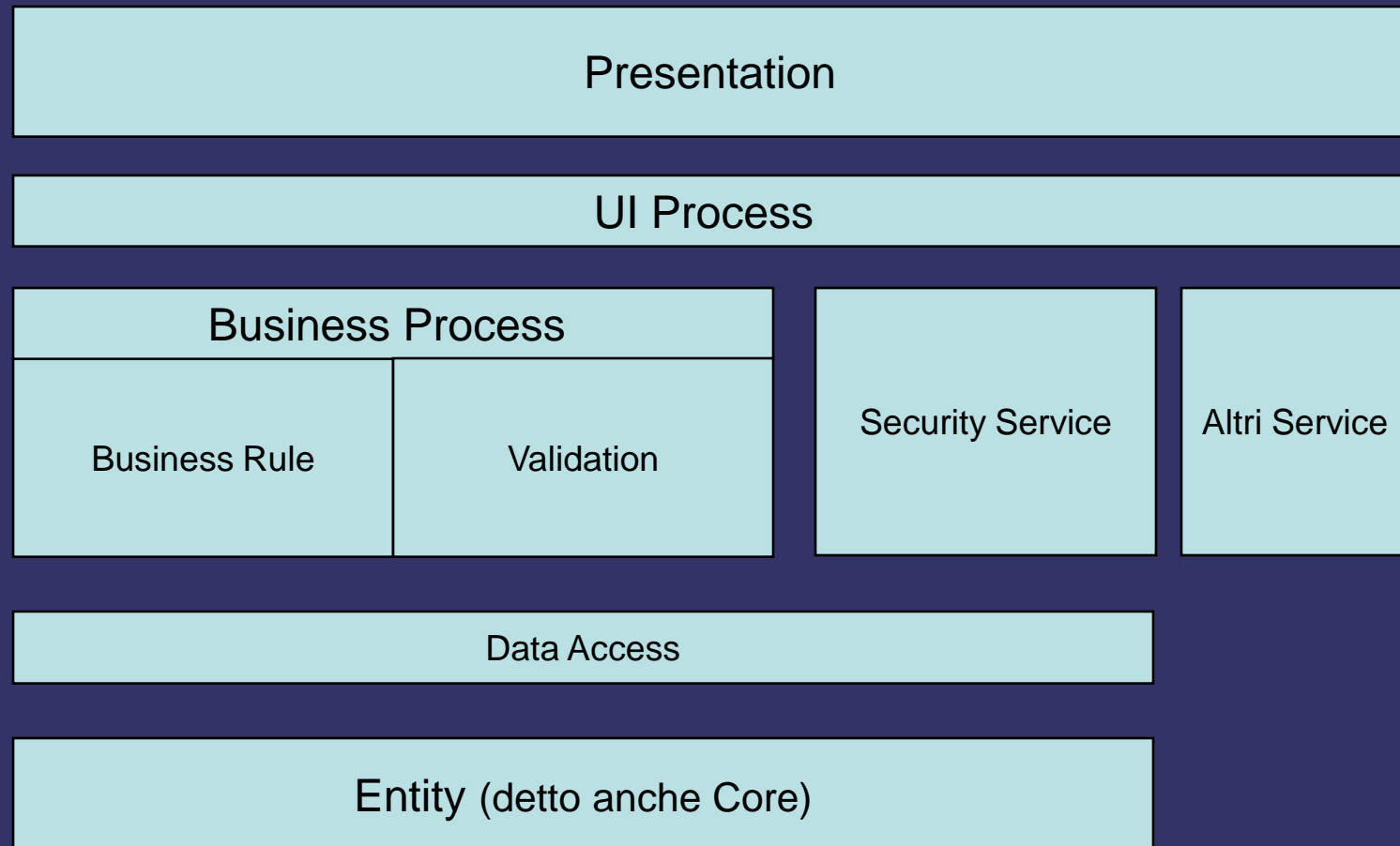


N-Layer Architecture

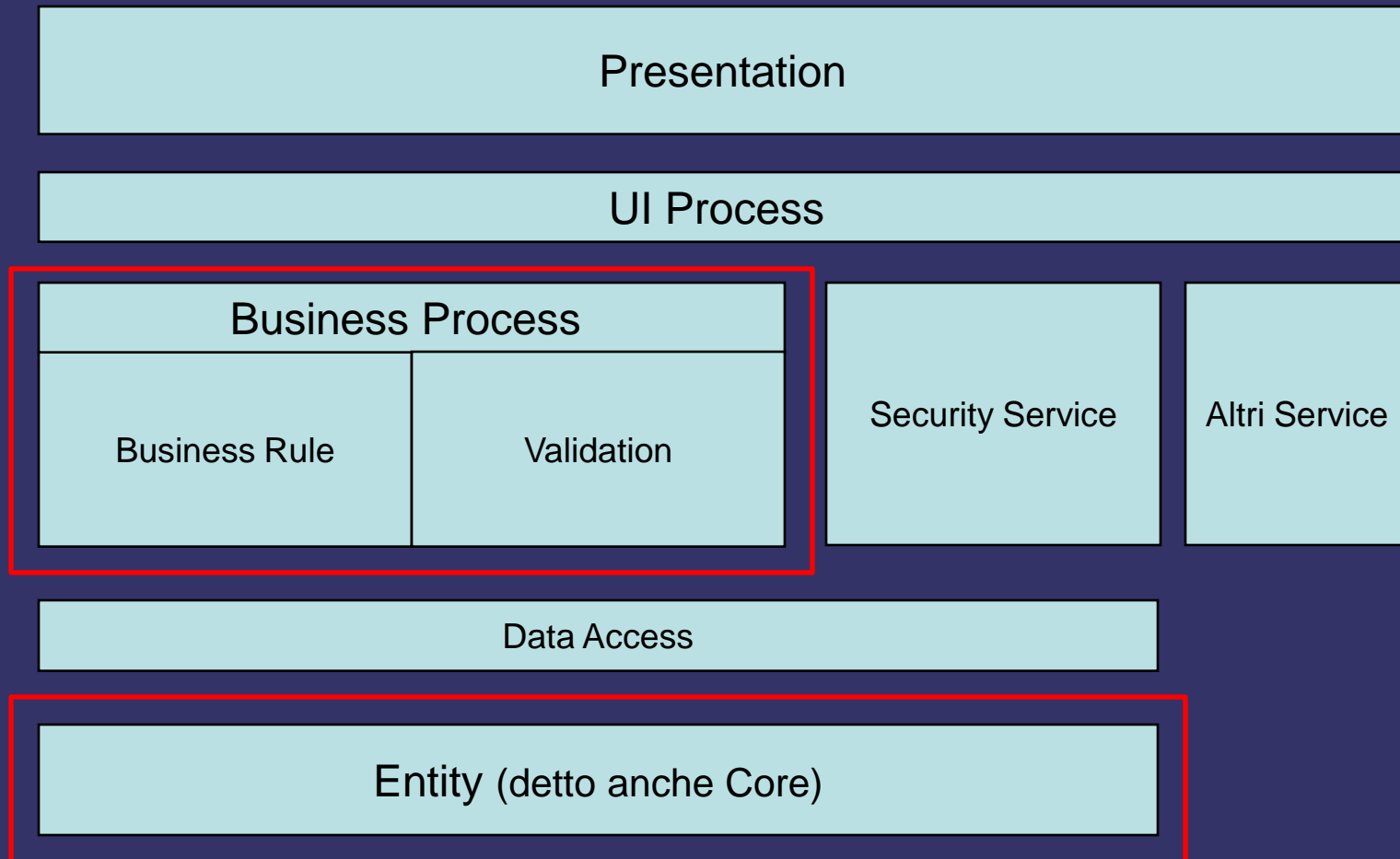
Contesto

- Migliorare il passaggio di dati tra layer
- Distinzione tra processi e servizi
- Distinzione tra servizi applicativi e infrastrutturali

Soluzione



Soluzione



Cenno alle Business Rule

- **Restrictions**: A customer must not place more than three rush orders charged to its credit account.
- **Heuristics**: A customer with preferred status should have its orders filled immediately.
- **Computations**: A customer's annual order volume must be computed as total sales closed during the company's fiscal year.
- **Inference**: A customer must be considered preferred if the customer places more than five orders over \$1,000.
- **Timing**: A customer must be archived if the customer doesn't place any orders for 36 consecutive months.
- **Triggers**: "Send advance notice" must be executed for an order when the order is shipped.

Isolamento del Dominio

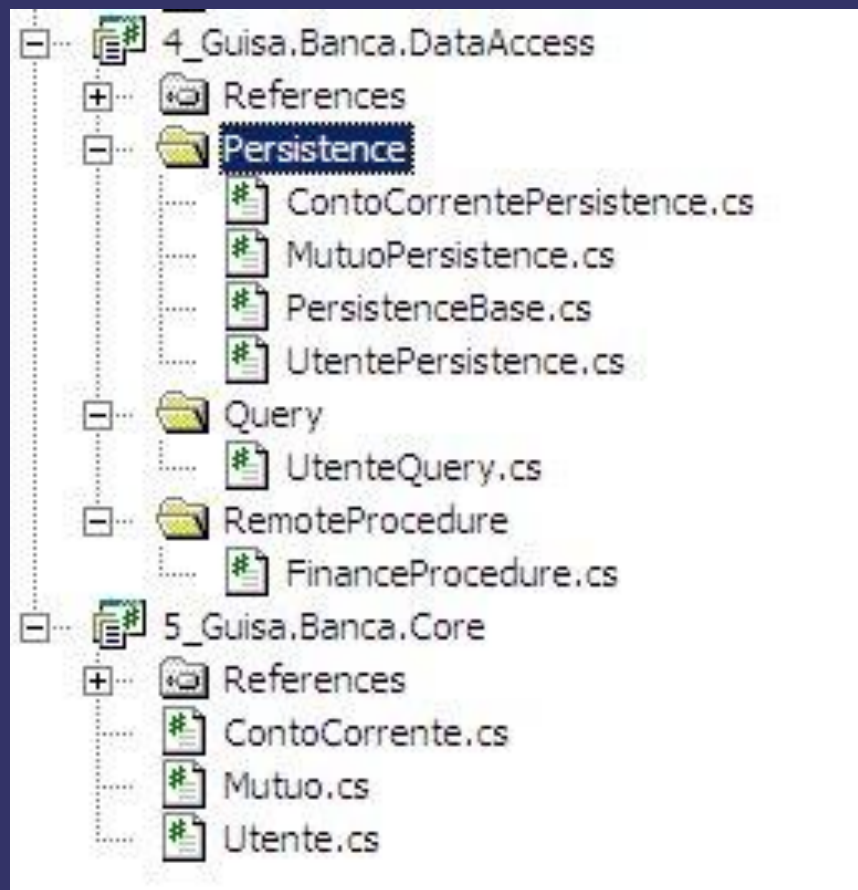
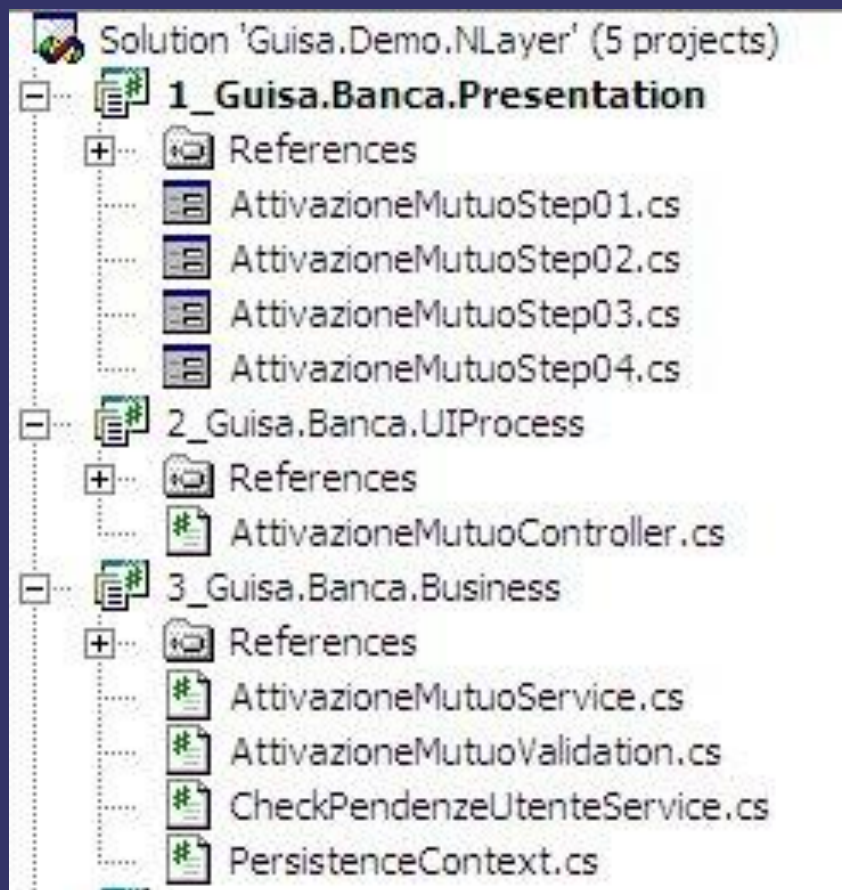
- Il DM deve essere quanto più indipendente da tutte le altre parti del sistema.
- E' buona norma rendere il DM facilmente isolabile, testabile, modificabile.
- La logica di business deve essere ortogonale ai servizi infrastrutturali. Evitare di inserire codice per aspetti infrastrutturali (persistenza, gestione delle transazioni, autorizzazioni e autenticazioni) nelle classi che costituiscono il Domain Model.
- Minimizzare la dipendenza anche dalle API di .NET. Renderà più facile il test senza il bisogno di ricreare a runtime particolari contesti come server, container etc.

N-Layer - Esempio

Processo di Attivazione Mutuo

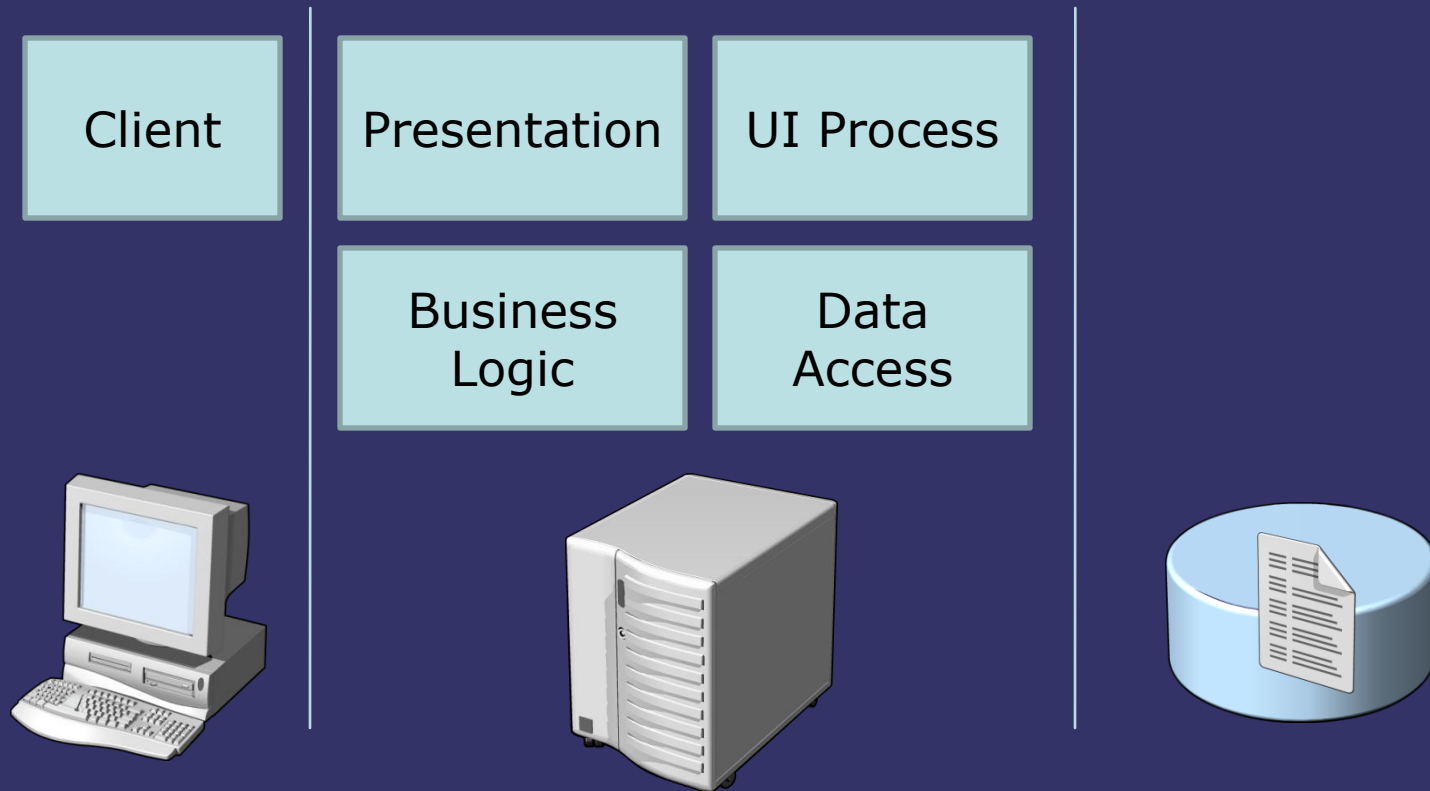


Processo di Attivazione Mutuo

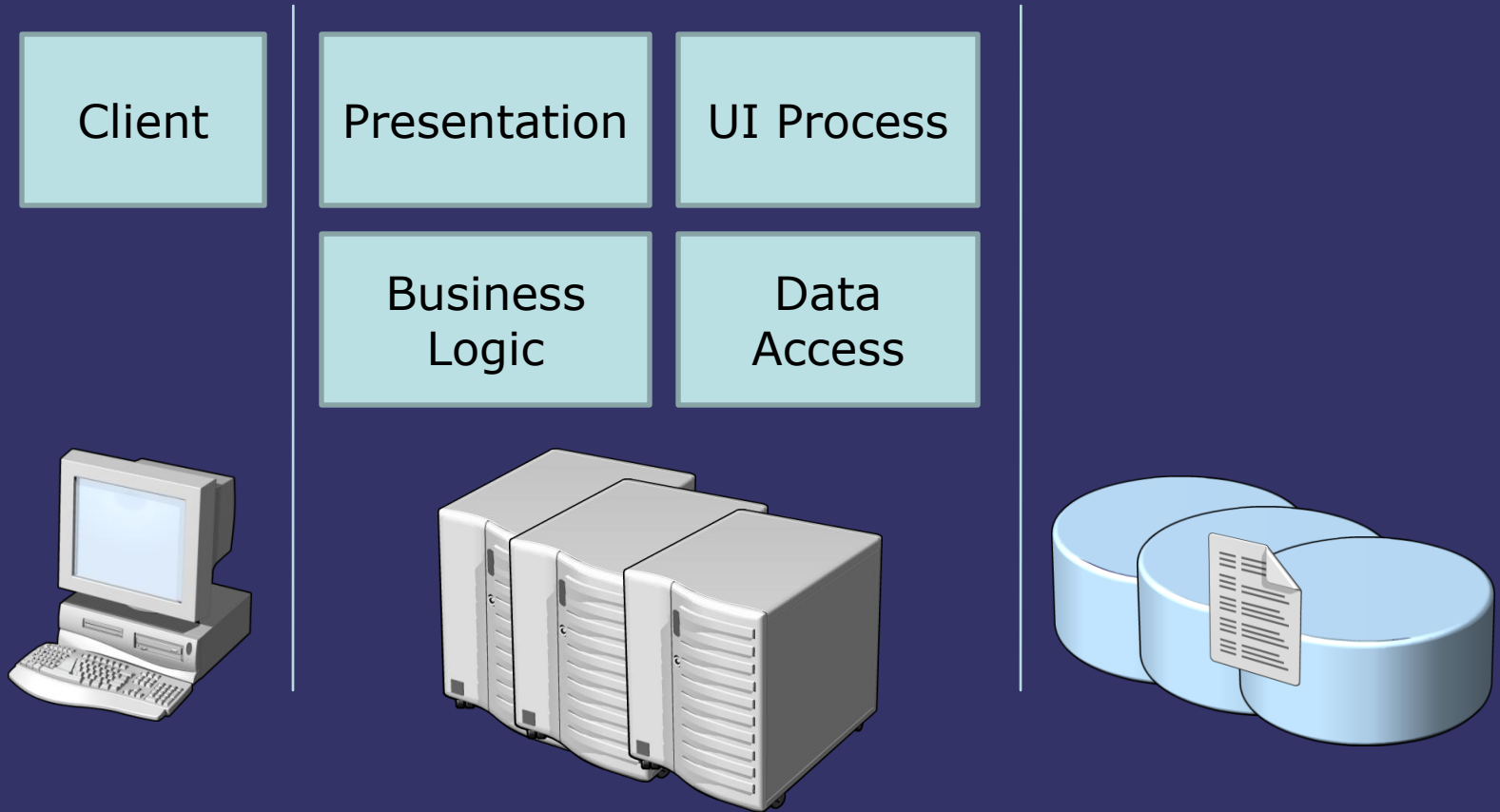


Distribuzione su Tier

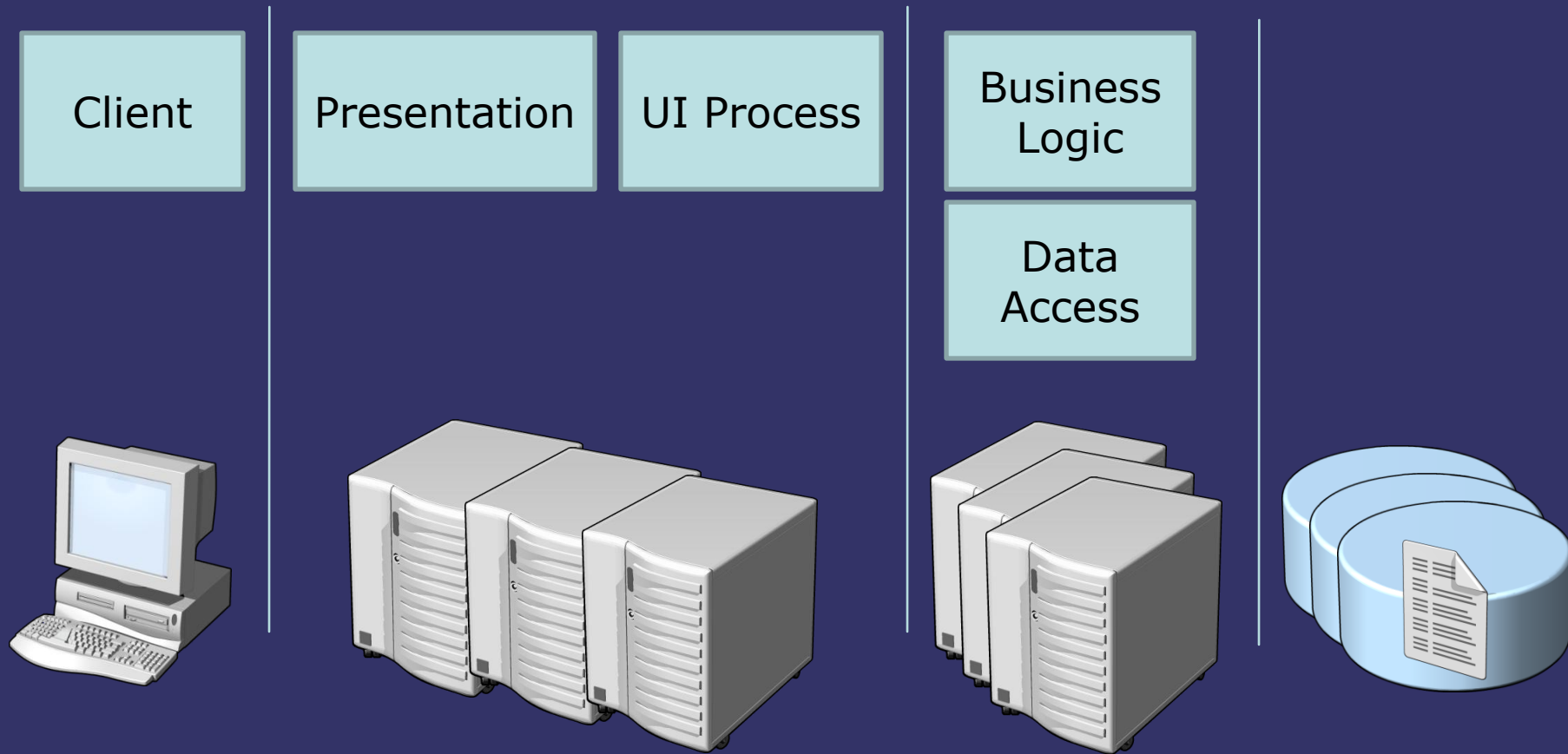
2-Tier



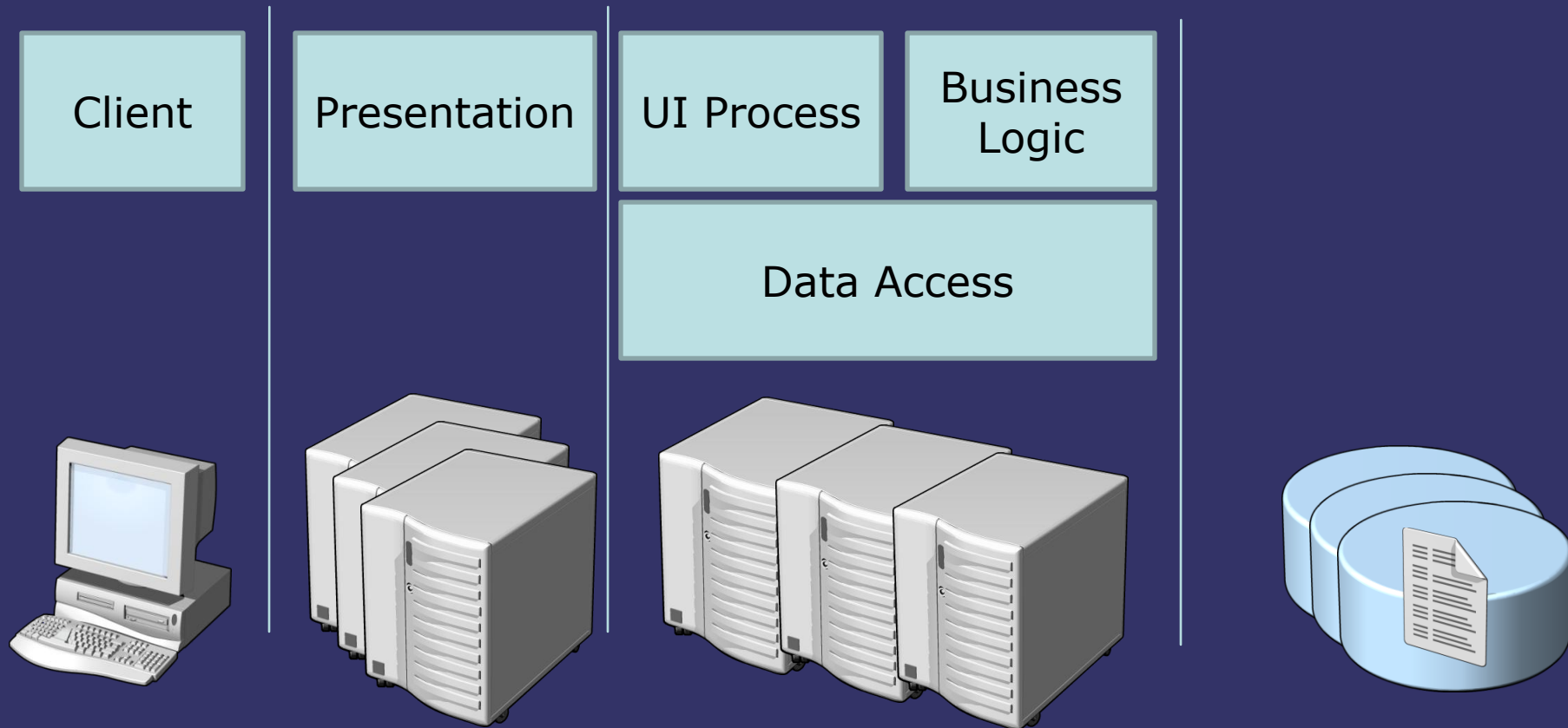
2-Tier (Load Balancing)



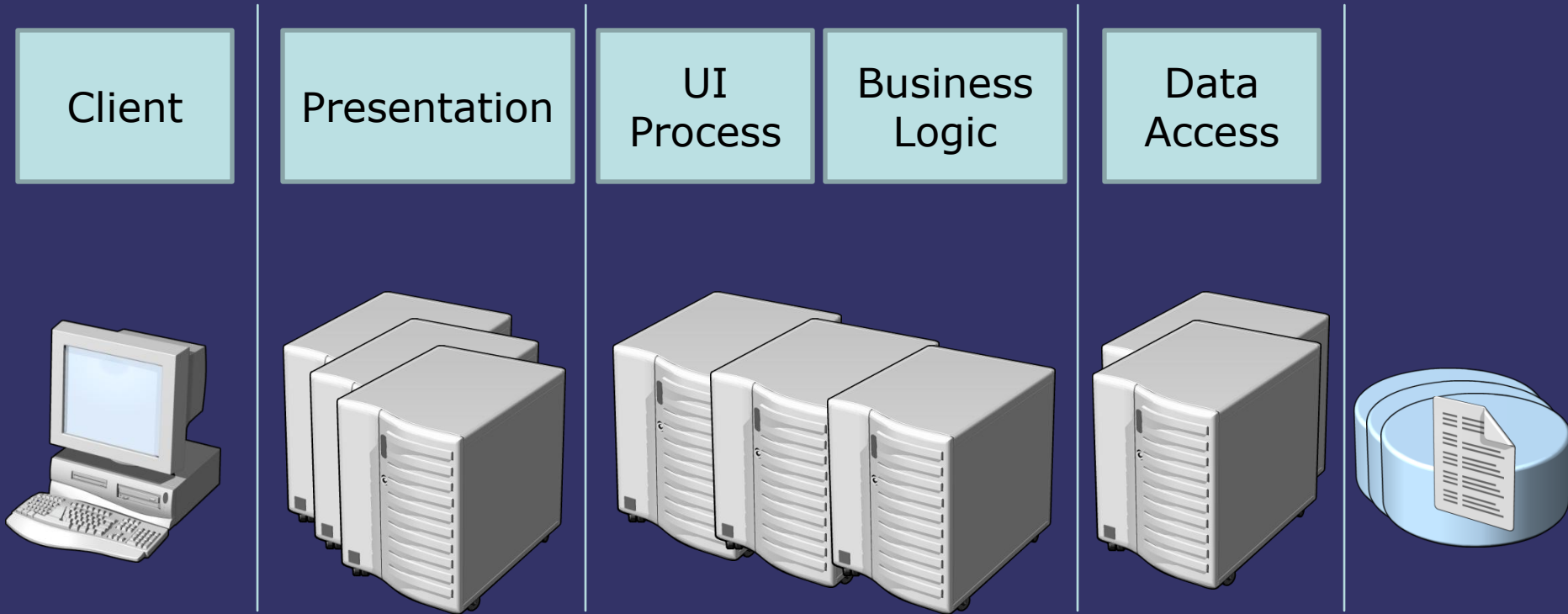
3-Tier (Load Balancing)



3-Tier (Load Balancing) (2)



4-Tier (Load Balancing)



Visual Studio 2005 è disponibile
scegli il prodotto per te www.microsoft.it/msdn/vs2005/

– **Visual Studio 2005 Team Edition**

- Visual Studio Team Edition con MSDN Premium (for Architects, Developers, Testers, **Database Professionals**)
- Visual Studio Team Suite con MSDN Premium



– **Strumenti professionali**

- Visual Studio 2005 Professional
- Visual Studio 2005 Professional con MSDN Professional
- Visual Studio 2005 Professional con MSDN Premium
- Visual Studio 2005 Tools for Microsoft Office System

– **Strumenti di base**

- Visual Studio 2005 Standard
- Visual Studio 2005 Express Edition

– **Altri strumenti**

- Visual SourceSafe 2005
- VisualFox Pro 9.0



Dove acquistare:

www.microsoft.it/msdn/rivenditori/

Per informazioni:

itamsgn@microsoft.com

Domande?

(facili facili...)