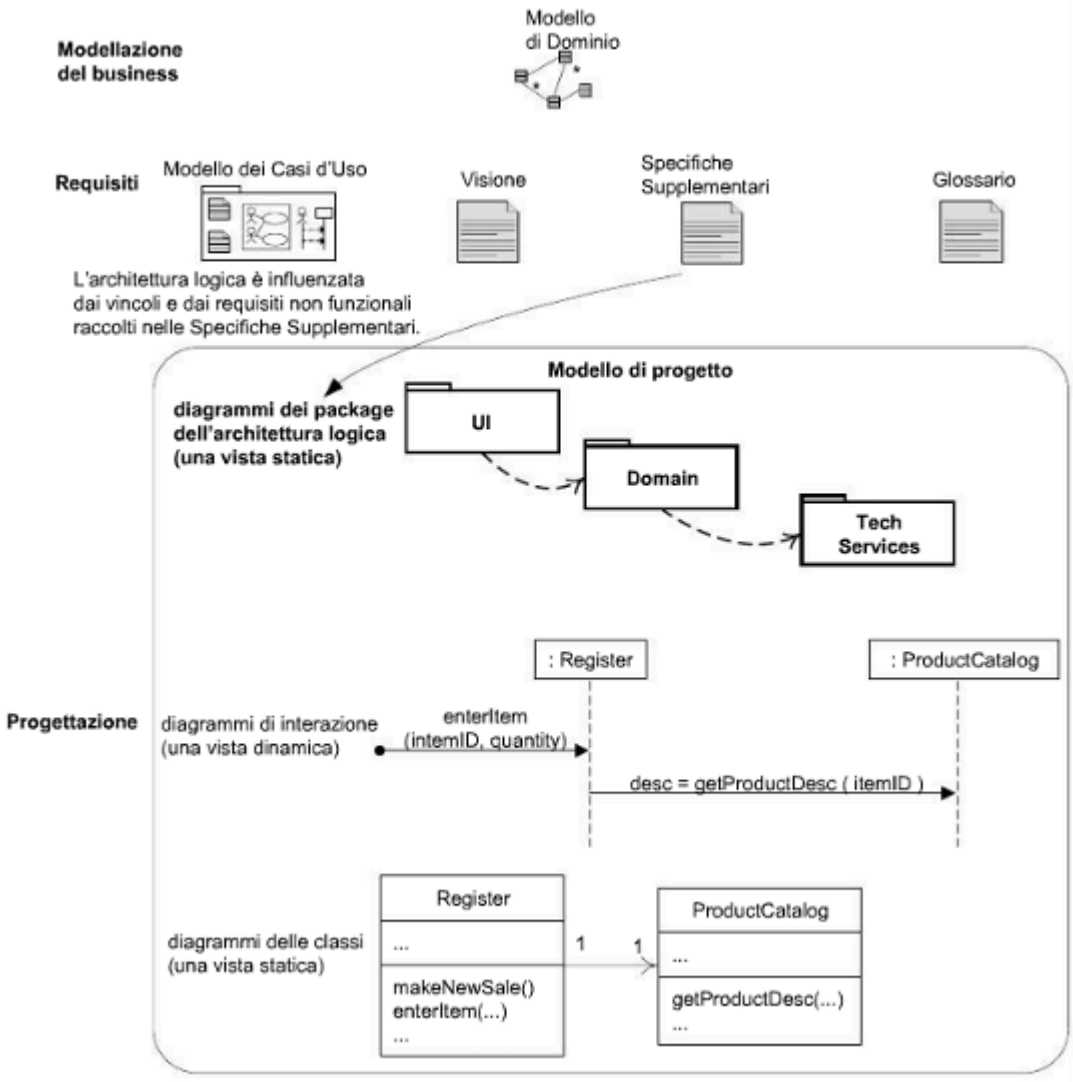
07 - Architettura logica e Organizzazione in Layer

L'architettura di un tipico sistema orientato agli oggetti è basata su diversi strati architetturali, come un'interfaccia utente, una logica applicativa (o "del dominio") e servizi tecnici.
L'architettura logica può essere illustrata con i diagrammi dei package di UML.



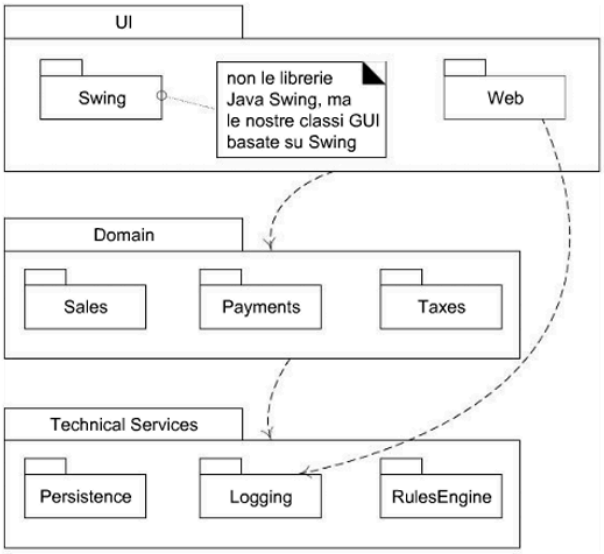
L'architettura logica di un sistema software è la **macro-organizzazione** su larga scala delle **classi software in package** (o namespace), **sottoinsiemi** e **strati**.
È chiamata "logica" perché non prende decisioni sulla distribuzione su processi o computer fisici (queste sono decisioni di architettura di deployment); è una **platform independent architecture**.

Un **layer** (**strato**) è un **gruppo di classi software, package o sottosistemi con responsabilità condivisa** su un aspetto importante del sistema.
Gli strati di un'applicazione software comprendono normalmente:

- **User interface** (interfaccia utente o presentazione): Oggetti software per gestire l'interazione con l'utente e gli eventi.

- **Application logic** o **domain objects** (logica applicativa o oggetti del dominio): Oggetti software che rappresentano concetti di dominio.
- **Technical services** (servizi tecnici): Oggetti e sottosistemi d'uso generale che forniscono servizi tecnici di supporto.

Un esempio dell'architettura per POS NextGen in diagrammi di package UML:




```
/*** Strato UI ***/

com.mycompany.nextgen.ui.swing
com.mycompany.nextgen.ui.web

/*** Strato DOMAIN ***/

// package specifici del progetto NextGen
com.mycompany.nextgen.domain.sales
com.mycompany.nextgen.domain.payments

/*** Strato TECHNICAL SERVICES ***/

// il nostro strato di persistenza (accesso alla base di dati)
com.mycompany.service.persistence

// terze parti
org.apache.log4j
org.apache.soap.rpc

/*** Strato FOUNDATION ***/

// package foundation creati dal nostro team
com.mycompany.util
```

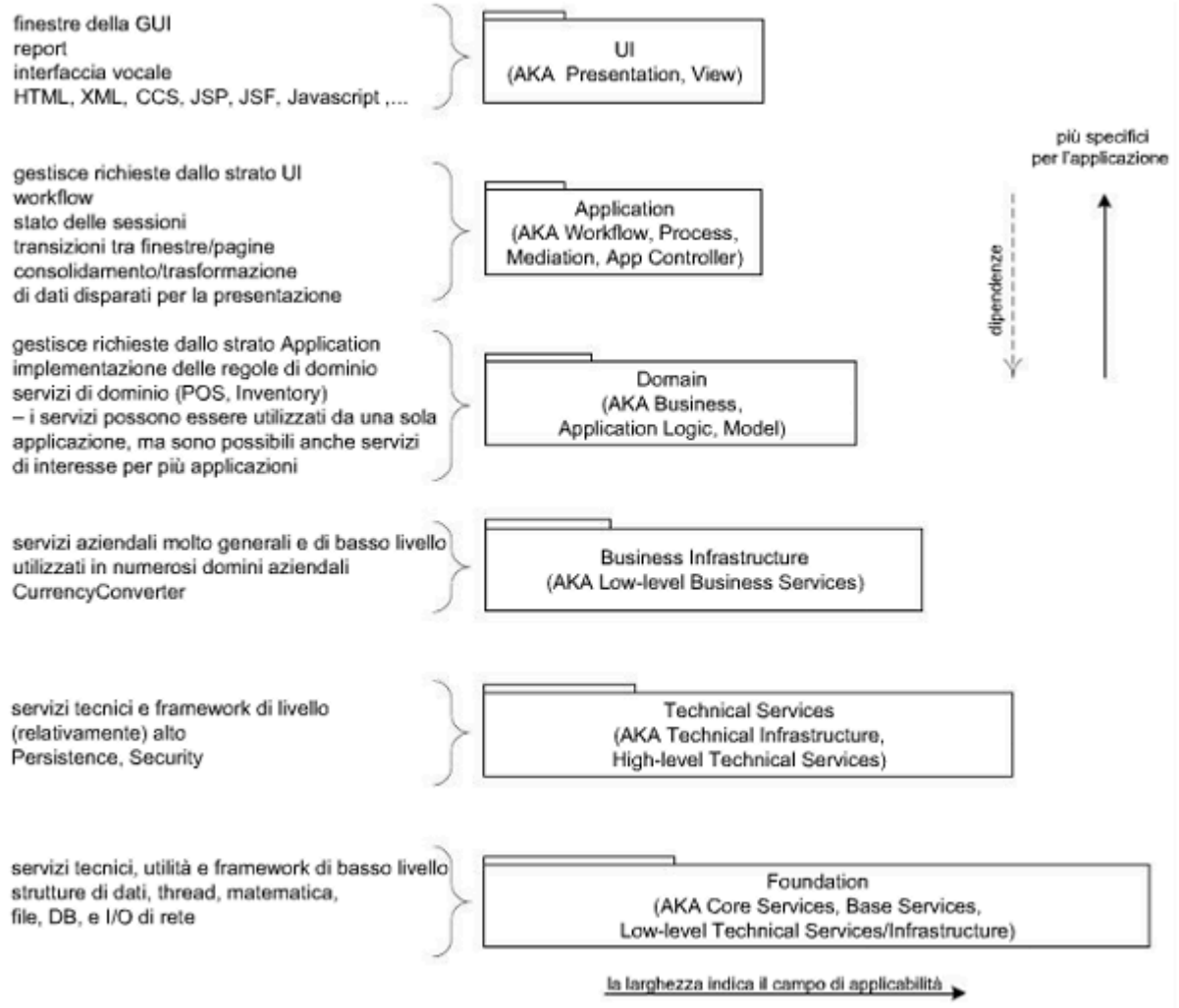
©C. Larman. Applicare UML e i Pattern. Pearson, 2016.

Architettura a Strati

L'obiettivo dell'architettura a strati è la suddivisione di un sistema complesso in elementi software che possono essere sviluppati e modificati il più indipendentemente possibile.

Questo promuove la "***Separation of concerns***" (separazione degli interessi), riducendo l'accoppiamento e le dipendenze, aumentando il riuso, facilitando la manutenzione e aumentando la chiarezza.
Favorisce anche l'**Alta Coesione**, dove le responsabilità degli oggetti in uno strato sono fortemente correlate e non mescolate con altri strati (es. oggetti UI non implementano logica applicativa), il che aumenta riuso, manutenibilità e chiarezza.

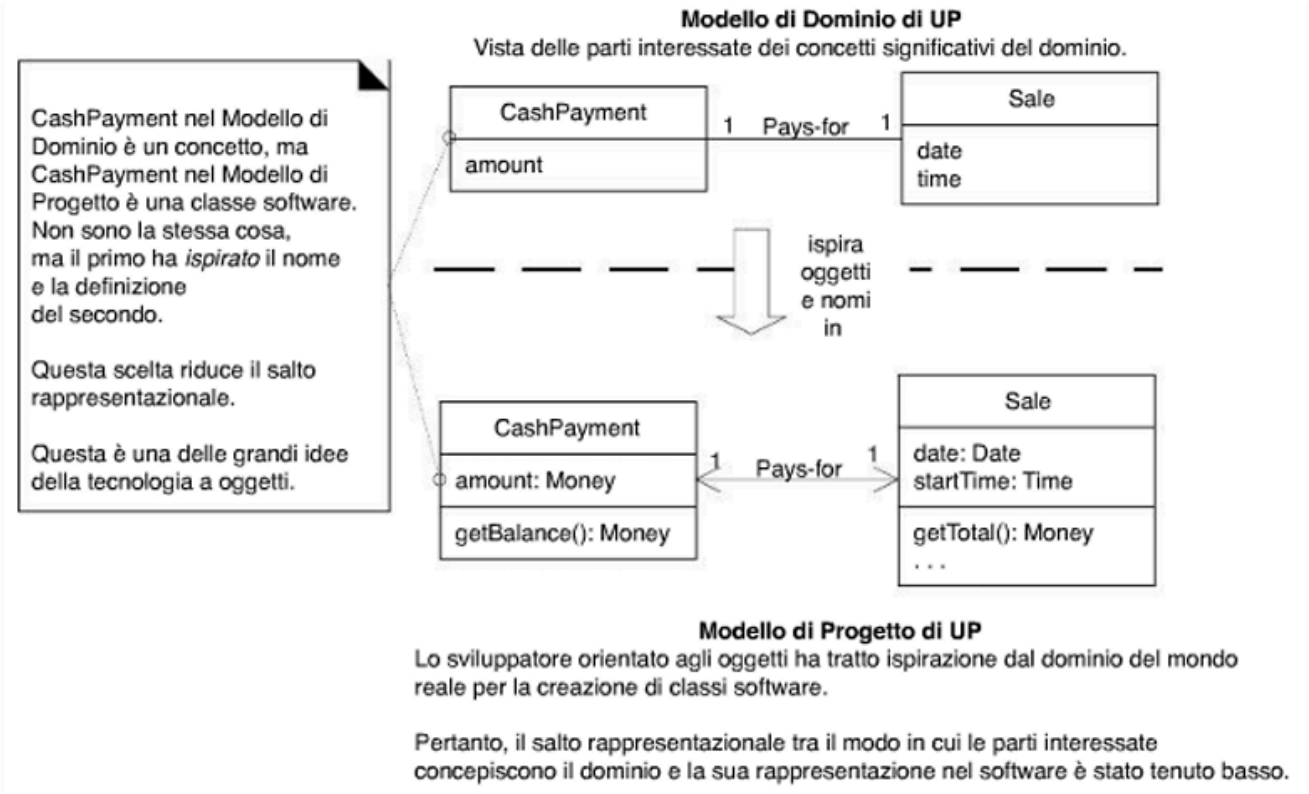
Una tipica architettura a strati:



Strato del Dominio

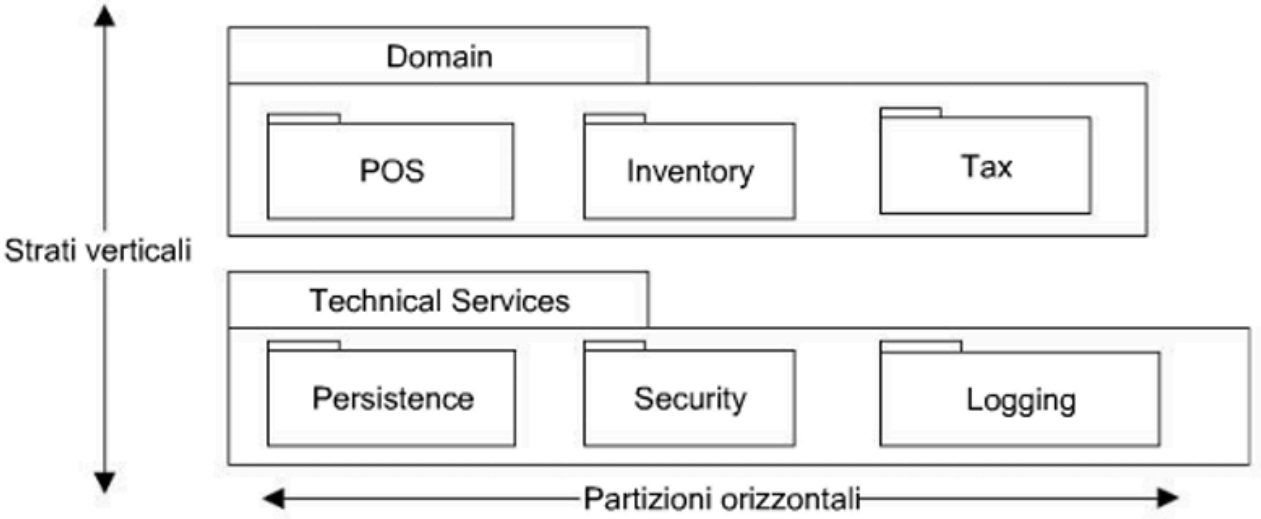
Per progettare la logica applicativa con gli oggetti, l'approccio consigliato è **creare oggetti software con nomi e informazioni simili al dominio del mondo reale**, e assegnare a essi responsabilità della logica applicativa.
Un **oggetto di dominio** è un oggetto software che rappresenta una cosa nello spazio di dominio del problema e ha una logica applicativa o di business correlata (es. un oggetto *Sale* è in grado di calcolare il suo totale).
Lo strato del dominio si ispira al modello di dominio per i nomi delle classi.

Creando uno strato del dominio in questo modo, si ottiene un "*salto rappresentazionale basso*" tra dominio del mondo reale e il progetto software.



Livelli, Strati e Partizioni

La nozione originaria di livello è di *strato logico*.
Si dice che gli **strati** di un'architettura rappresentano le *sezioni verticali*, mentre le **partizioni** rappresentano una *divisione orizzontale* di sottosistemi relativamente paralleli di uno strato.



Principio di Separazione Modello-Vista

Il principio di separazione Modello-Vista stabilisce:

1. **Non relazionare o accoppiare oggetti non UI con oggetti UI**: Gli oggetti non UI (es. oggetti di dominio) non devono essere connessi o accoppiati direttamente agli oggetti UI.
2. **Non incapsulare la logica dell'applicazione in metodi di UI**: Non inserire logica applicativa (es. calcolo delle imposte) nei metodi di un oggetto dell'interfaccia utente. Gli

oggetti UI dovrebbero solo inizializzare elementi UI, ricevere eventi UI e delegare le richieste di logica applicativa agli oggetti non UI.

Questo principio è fondamentale perché le finestre appartengono a un'applicazione specifica, mentre gli oggetti non UI possono essere riutilizzati in nuove applicazioni o relazionati a nuove interfacce.

Modello = strato di dominio	**Vista = strato UI**
è sinonimo per lo strato degli oggetti del dominio	è un sinonimo per gli oggetti dell'interfaccia utente

Gli oggetti del modello (dominio) non devono avere una conoscenza diretta degli oggetti della vista (UI), almeno in quanto oggetti della vista. Questo è un principio fondamentale del pattern **Model-View-Controller** (**MVC**).

> **MVC**
>
> Era un pattern di Smalltalk-80 relativo a oggetti dati (modelli), elementi della GUI (vista) e gestori degli eventi del mouse e della tastiera (controller).

- Le classi di dominio **incapsulano** informazioni e comportamento relativi alla logica applicativa.
- Le classi della vista sono relativamente leggere, responsabili di input/output e di catturare eventi GUI, ma non mantengono dati dell’applicazione né forniscono direttamente logica applicativa.

Vantaggi del principio di separazione Modello-Vista:

- Favorisce la definizione coesa dei modelli.
- Consente lo sviluppo separato degli strati del modello e dell’interfaccia utente.
- Minimizza l’impatto sullo strato del dominio dei cambiamenti dei requisiti UI.
- Consente di connettere facilmente nuove viste a uno strato del dominio esistente.
- Consente viste multiple, simultanee sugli stessi oggetti modello.
- Consente l’esecuzione dello strato di modello indipendente da quella dello strato UI (batch o a messaggi).
- Consente un porting facile dello strato di modello ad un altro framework UI.

SSD

Gli SSD mostrano le operazioni di sistema ma nascondono gli oggetti specifici della UI. Gli oggetti dello strato UI inoltreranno (o delegaranno) le richieste dallo strato UI allo strato del dominio.

I messaggi inviati dallo strato UI allo strato del dominio saranno gli stessi mostrati negli SSD.