Strategie per la strutturazione a livelli

Peter Eeles

White paper del software Rational

TP 199, 08/01

Rational®
the software development company

Indice

Abstract

È possibile adottare una serie di tecniche per scomporre sistemi software. La strutturazione a livelli è una di queste e verrà descritta nel presente documento. Tali tecniche riguardano principalmente due problematiche: gran parte dei sistemi sono troppo complessi per essere interamente compresi e sono necessarie prospettive diverse di un sistema per utenze diverse.

la strutturazione a livelli è stata adottata in numerosi sistemi software e viene illustrata in molti testi e anche in RUP (Rational Unified Process). Malgrado ciò, questa tecnica spesso non viene compresa a fondo e non viene applicata correttamente. Il presente documento chiarisce il concetto di strutturazione a livelli e illustra gli impatti delle varie strategie.

Che cosa si intende per “Strutturazione a livelli”?

Cominciamo con la spiegazione dell'espressione "Strutturazione a livelli". Il termine livello si riferisce all'applicazione di un pattern di architettura conosciuto più generalmente con il nome "Livelli", descritto in numerosi test ([Buschmann], [Herzum], [PloP2]) e anche in RUP. Un pattern rappresenta la soluzione a un problema comune che esiste in un dato contesto. Una panoramica del pattern Strutturazione a livelli si trova nella tabella 1.

Tabella 1: Panoramica del pattern "Livelli"

	Pattern Strutturazione a livelli
Contesto	Un sistema che richiede la scomposizione
Problema	Un sistema troppo complesso da comprendere nella sua totalità Un sistema difficile da mantenere Un sistema i cui elementi meno stabili non vengono isolati Un sistema in cui la maggior parte degli elementi riutilizzabili sono difficili da identificare Un sistema che deve essere realizzato da vari team
Soluzione	Strutturazione a livelli del sistema

Uno degli esempi più comuni della strutturazione a livelli è il modello OSI 7, definito dall'ISO (International Standardization Organization). Questo modello, mostrato nella Figura 1, definisce una serie di protocolli di rete; ogni livello si focalizza su un aspetto specifico della comunicazione e si basa sulle funzioni del livello sottostante. Il modello OSI 7 utilizza una responsabilità basata sulla strategia di strutturazione a livelli: ogni livello ha una responsabilità particolare. Tale strategia viene descritta nei dettagli più in là.

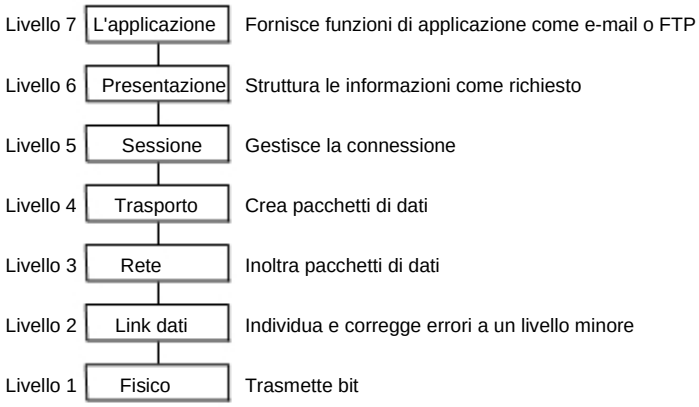

Figura 1: Modello OSI 7 (Strutturazione a livelli basata sulla responsabilità)

La Figura 2 mostra un altro esempio di strutturazione a livelli basata sulla responsabilità.

- Il livello Logica di presentazione contiene elementi utili a offrire una resa grafica interpretabile dall'uomo, come ad esempio l'interfaccia utente.
- Il livello Logica di business contiene elementi che eseguono una sorta di elaborazione del business e che applicano regole di business.
- Il livello Logica di accesso ai dati contiene elementi responsabili dell'accesso ad una fonte di informazioni, come un database relazionale.

Si osservi che i livelli possono essere modellati in vari modi, come descriveremo più in là. Per il momento, rappresentiamo esplicitamente un livello che utilizza un pacchetto UML con lo stereotipo ½layer╗.

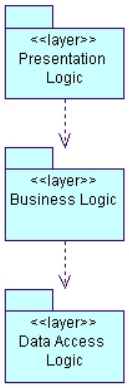

Figura 2: Strutturazione a livelli basata sulla responsabilità

I livelli mostrati in questo specifico esempio di strutturazione a livelli basata sulla responsabilità sono spesso chiamati “tier” e sono comuni nello sviluppo di sistemi distribuiti in cui si parla di tier 2, tier 3 e tier n.

Un aspetto importante della Figura 2 è la direzione delle dipendenze, poiché implica una regola caratteristica dei sistemi a livelli; un elemento di un particolare livello può accedere soltanto ad elementi dello stesso livello o dei livelli sottostanti[1]. In questo esempio, gli elementi del livello Logica di business non possono accedere agli elementi del livello Logica di presentazione. Inoltre, gli elementi del livello Logica di accesso ai dati non possono accedere agli elementi del livello Logica di business. Tale struttura viene spesso chiamata DAG (Directed Acyclic Graph). È diretta in quanto le dipendenze sono unidirezionali e un percorso aciclico di dipendenze non è mai circolare.

In altri termini, occorre essere più precisi sul significato di ciascun livello quando si definisce una strategia di strutturazione a livelli, affinché gli elementi siano situati correttamente nel livello appropriato. Un errore nell'assegnazione di un elemento al livello appropriato diminuirà il valore della strategia da applicare al primo punto. Dato che ogni strategia di strutturazione a livelli viene discussa nei dettagli, si forniscono informazioni generali sul significato di ciascun livello.

Modellazione dei livelli

Poiché vengono contemplate varie strategie di strutturazione a livelli, è chiaro che bisogna comunicare ogni strategia utilizzando modelli specifici (e quindi elementi UML specifici). Un modello rappresenta la descrizione completa di un sistema da una determinata prospettiva. La Figura 3 mostra un esempio di quattro modelli che rappresentano varie prospettive del sistema preso in considerazione:

[1] Sebbene la notifica di un evento possa risultare in un messaggio con l'invio di un elemento di un livello verso un elemento di un livello superiore, non esiste una dipendenza esplicita in questa direzione.

- Modello Caso d'uso: cattura i requisiti del sistema
- Modello Analisi: cattura l'analisi dei requisiti del sistema
- Modello Progettazione: contiene la progettazione del sistema
- Modello Implementazione: contiene l'implementazione del sistema

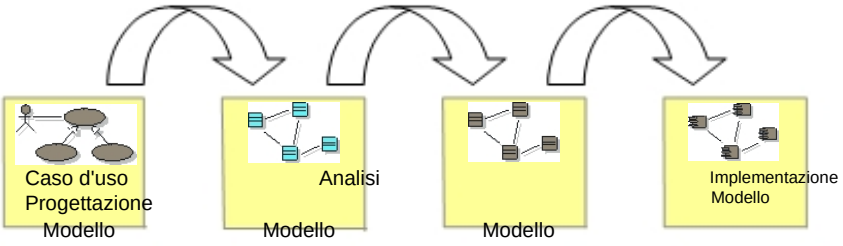

Figura 3: quattro modelli che rappresentano il perfezionamento graduale.

I modelli aggiuntivi includono:

- Modello Distribuzione: include gli aspetti relativi alla distribuzione di un sistema
- Modello Dati: cattura gli aspetti permanenti di un sistema

Strategie per la strutturazione a livelli

La strutturazione a livelli può essere basata su numerose caratteristiche. Questa sezione discute la strutturazione a livelli in base alle seguenti caratteristiche:

- responsabilità
- riutilizzo

La rappresentazione di ogni strategia sarà considerata nel momento in cui la strategia verrà illustrata.

Strutturazione a livelli basata sulla responsabilità

Probabilmente la strategia di strutturazione a livelli più comunemente utilizzata è quella basata sulla responsabilità. Questa particolare strategia può migliorare lo sviluppo e la manutenzione di un sistema poiché varie responsabilità del sistema sono isolate le une dalle altre. Ad esempio (vedere Figura 2), un sistema può essere strutturato a livelli in base alle seguenti responsabilità:

- logica di presentazione
- logica di business
- logica di accesso ai dati

Ognuna di queste responsabilità può essere rappresentata con un livello, come mostrato dalla Figura 4, che illustra alcuni esempi di contenuti per ciascun livello. Consideriamo qui tre concetti in un sistema di elaborazione ordini (Cliente, Ordine, Prodotto). Ad esempio, il concetto Cliente include quanto segue:

- Classe CustomerView: responsabile della logica di presentazione associata ad un cliente, come la resa grafica interpretabile dal cliente nell'interfaccia utente
- Classe Customer: responsabile della logica di business associata ad un cliente, come la validazione di dettagli sul cliente

= Classe CustomerData: responsabile della logica di accesso ai dati associata ad un cliente, come il fatto di rendere permanente lo stato di un cliente

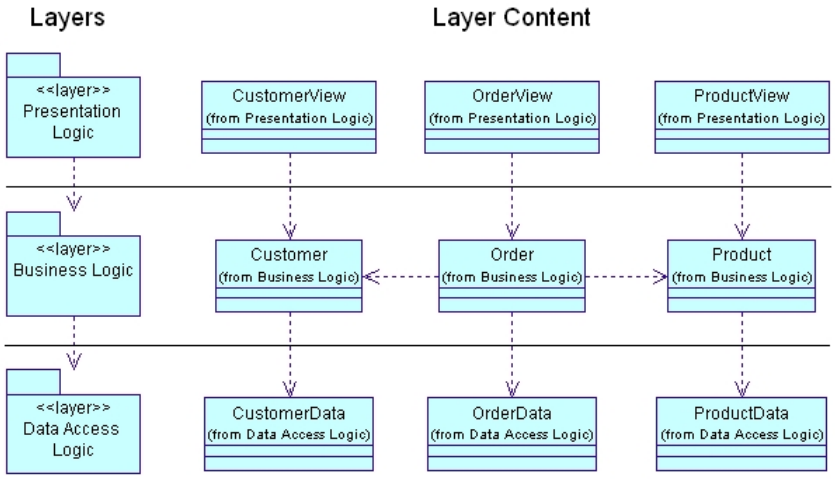

Figura 4: livelli e contenuti relativi alla strutturazione a livelli basata sulla responsabilità

Prenderemo ora in considerazione alcuni "assunti" riguardanti questa particolare strategia di strutturazione a livelli.

Assunto 1: i livelli e i tier sono diversi

Questo particolare assunto è solitamente motivo di confusione. Di fatto, un tier è un livello, anche se si tratta di un livello basato su una particolare strategia di responsabilità. Ad aggravare questa confusione contribuisce il fatto che i concetti di tier possono essere applicati in varie situazioni, come mostrato dalla tabella 2.

Tabella 2: definizioni di tier

L'applicazione	Livelli (tier)
tier 2	logica di presentazione combinata e logica di business logica di accesso ai dati
tier 3	logica di presentazione logica di business logica di accesso ai dati
tier n	logica di presentazione logica di business distribuita logica di accesso ai dati

Assunto 2: i livelli (tier) implicano una distribuzione fisica

Un'altra convinzione erronea è che la strutturazione logica a livelli implica una distribuzione fisica. Considerare una strutturazione a 3 livelli. Anche se alcuni elementi si trovano in uno dei livelli, ogni livello può essere applicato in molti modi come illustra la tabella 3, che utilizza nomi spesso impiegati per descrivere una distribuzione fisica particolare (come “client minore”).

Tabella 3: applicazione della strutturazione a 3 livelli

L'applicazione	Livelli	
	Lato Client	Lato Server
Singolo sistema	logica di presentazione logica di business l ogica di accesso ai dati	
Client minore	logica di presentazione logica di business	logica di business logica di accesso ai dati
Client maggiore	logica di presentazione	logica di accesso ai dati

È anche vero che un singolo sistema può utilizzare più di una strategia di distribuzione fisica, dove alcuni elementi possono essere classificati come rappresentanti una distribuzione di “client minore” e altri una distribuzione di “client maggiore”. Solitamente, la scelta si basa su requisiti non funzionali, come le prestazioni.

Modellazione di strutturazione a livelli basata sulla responsabilità

Come vedremo, l'applicazione di questa strategia può influenzare il modello di progettazione, il modello di implementazione e il modello di distribuzione. Il modello Progettazione è generalmente strutturato utilizzando uno o due approcci.

Il primo approccio mostra gli elementi “contenuti” nel livello. Il risultato è suggerito nella Figura 5, uno screenshot del browser di Rational Rose che include:

- classi di presentazione (CustomerView, OrderView e ProductView) all'interno di una Logica di presentazione pacchetto
- classi di logica di business (Cliente, Ordine e Prodotto) all'interno di un pacchetto di Logica di business
- classi di logica di accesso ai dati (CustomerData, OrderData e ProductData) all'interno del pacchetto di logica di accesso ai dati

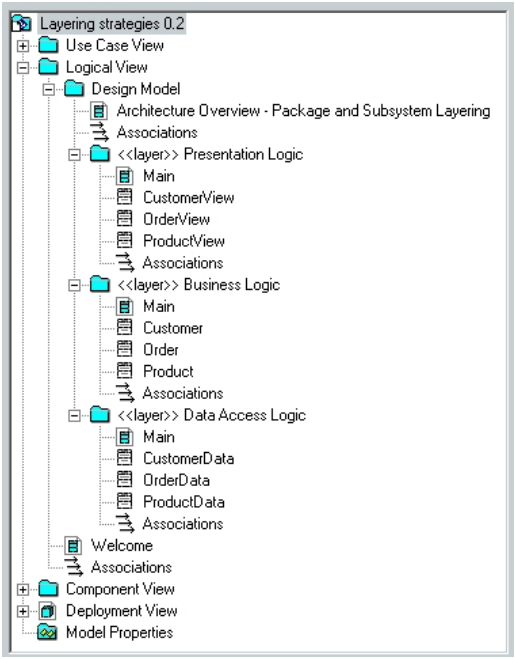

Figura 5: elementi contenuti nei livelli

Il secondo approccio include il concetto di un componente business (in questo caso, Cliente, Ordine e Prodotto) come cittadino di classe A, in cui gli elementi principali della problematica sono concetti inerenti al dominio supportato dal sistema. Ad esempio, il concetto Cliente può includere elementi relativi alla logica di presentazione, di business e di accesso ai dati. Il concetto di componente business è ulteriormente discusso in [Eeles] e [Herzum]. Questo modo di pensare si riflette nella struttura del modello illustrato dalla Figura 6. In questo esempio la strutturazione a livelli è suggerita dai nomi degli elementi. Tutte le classi View (come ad esempio CustomerView) presuppongono una logica di presentazione e tutte le classi Data (come CustomerData) un livello di logica di accesso ai dati. I nomi di classi non qualificate (come Cliente) sono inclusi in un livello di logica di business.

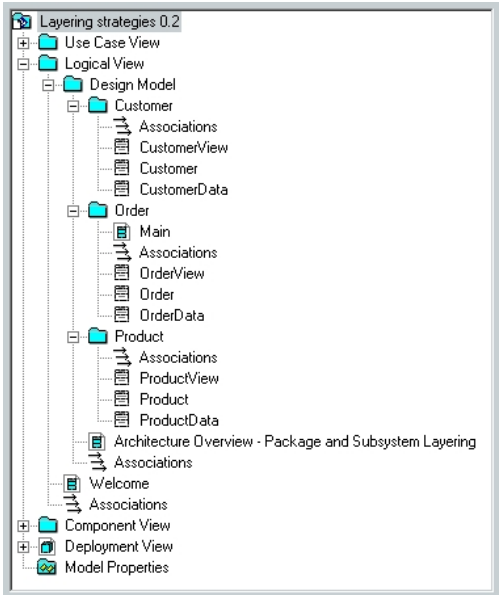

Figura 6: strutturazione a livelli implicita in ogni pacchetto del componente business.

La strutturazione a livelli può anche essere rappresentata in maniera esplicita se ogni pacchetto definisce un componente business, come illustrato nella Figura
7. Questa strutturazione è preferibile quando numerosi elementi sono concentrati nel livello di un certo componente business. Sebbene in questo esempio sia stato descritto soltanto il pacchetto del componente business Cliente, i pacchetti Ordine e Prodotto avrebbero una struttura simile.

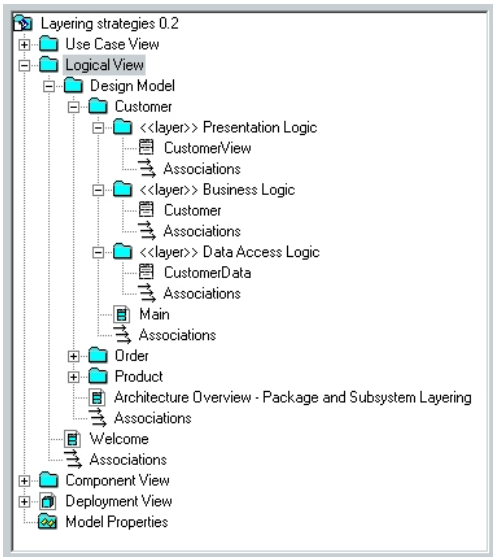

Figura 7: strutturazione a livelli esplicita in un pacchetto del componente business

Una strategia di strutturazione a livelli basata sulla responsabilità influenza solitamente il modello Implementazione, oltre al modello Progettazione, quando occorre una suddivisione fisica degli elementi che implementano ogni responsabilità. Ad esempio, consideriamo un sistema che illustra una distribuzione fisica di "client minore": è utile identificare le unità di implementazione richieste per supportare l'esecuzione su un client e quelle per supportare l'esecuzione sul server. In questo esempio, gli elementi del livello della logica di presentazione si trovano in un'applicazione distribuita su un client e tutti gli elementi del livello della logica di business e del livello della logica dati sono in un'altra applicazione distribuita su un server.

Questo scenario indica un modello di implementazione, come mostra la Figura 8, che presenta l'immagine di un browser Rational Rose e di un diagramma di componente in cui si visualizzano gli elementi dell'applicazione distribuita sul client. In questo esempio troviamo una mappatura diretta tra una classe del modello Progettazione e un componente UML del modello Implementazione. Notare, comunque, che tale mappatura dipende spesso dalla tecnologia di implementazione impiegata.

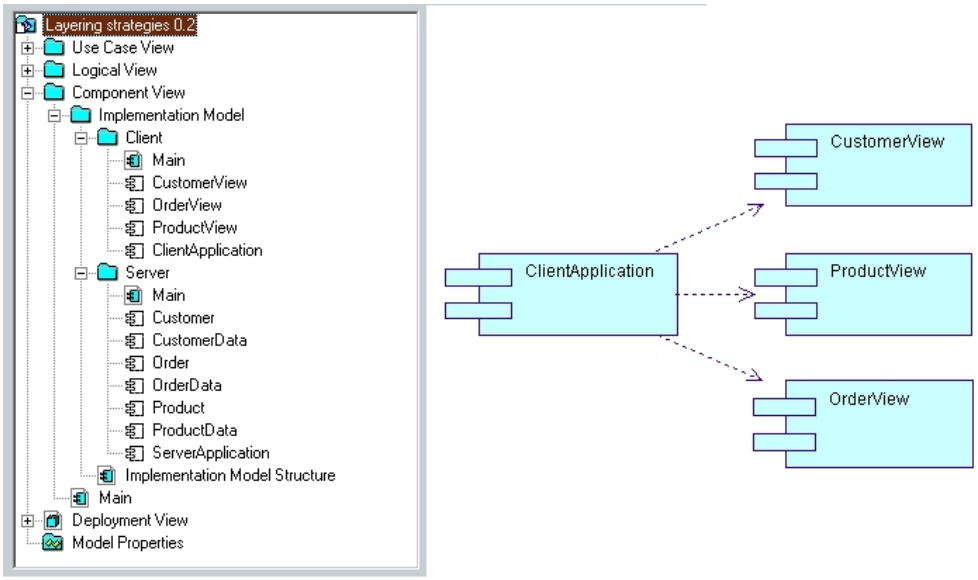

Figura 8: strutturazione a livelli implicita nel modello Implementazione

Parimenti, una strategia di strutturazione a livelli basata sulla responsabilità può anche influenzare il modello Distribuzione qualora fosse necessario descrivere la distribuzione fisica delle responsabilità. Nella Figura 9, seguendo l'esempio sopra illustrato, possiamo osservare che sono stati definiti sei nodi. Ognuno dei tre nodi Client ospita un processo ClientApplication. Il nodo FrontEndServer ospita un processo LoadBalancer, responsabile della distribuzione delle richieste del client a uno dei due nodi Server. Ogni nodo Server ospita un processo
ServerApplication.

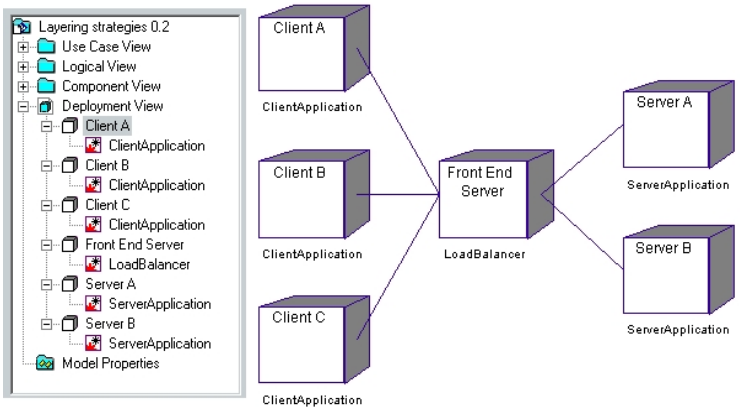

Figura 9: modello Distribuzione che descrive la distribuzione fisica delle responsabilità

Modellazione basata sul riutilizzo

Un'altra strutturazione a livelli frequente è quella basata sul riutilizzo. Questa strategia è particolarmente indicata per organizzazioni in cui si vogliono riutilizzare dei componenti. L'impatto di tale strategia di strutturazione a livelli consiste nella visibilità circa il riutilizzo dei componenti, in quanto questi ultimi sono raggruppati secondo il livello di possibile riutilizzo. Un esempio di strutturazione a livelli, derivato da una strategia descritta in [Jacobson], viene illustrato dalla Figura 10. Troviamo qui tre livelli: Base, Business e Applicazione.

= Il livello Base mostra elementi da applicare nelle organizzazioni (come Math). Tali elementi saranno ampiamente riutilizzati.

= Il livello Business include elementi da applicare ad un'organizzazione specifica, ma che sono indipendenti dall'applicazione (come Address Book). Questi elementi saranno riutilizzati in applicazioni della stessa organizzazione.

= Il livello Applicazione contiene elementi per un particolare tipo di applicazione o progetto (come Personal Organizer). Tali elementi sono i meno riutilizzabili.

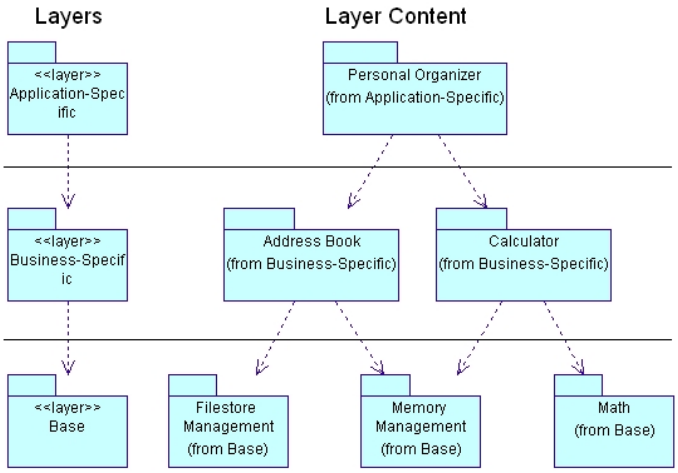

Figura 10: esempio di strutturazione a livelli basata sul riutilizzo

Possiamo dedurre che gli elementi del livello Base sono i più riutilizzabili, mentre quelli dell'Applicazione sono più specifici al progetto e, quindi, meno riutilizzabili.

Modellazione dei livelli basati sul riutilizzo

L'applicazione di una strategia di riutilizzo influenza principalmente il modello Progettazione. La struttura di un modello Progettazione che include la strutturazione a livelli basata sul riutilizzo è semplice da contemplare e viene mostrata nella Figura 11, che riflette l'esempio della Figura 10.

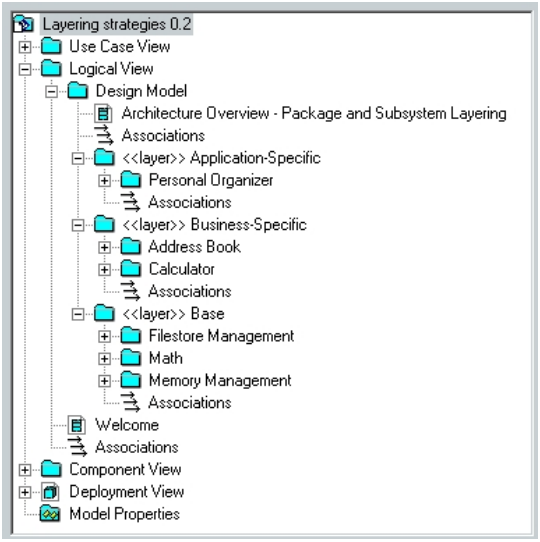

Figura 11: modello Progettazione che include strutturazione a livelli basata sul riutilizzo.

Altre strategie di strutturazione a livelli

Questo documento intende semplicemente fornire un' "idea" delle diverse strategie di strutturazione a livelli esistenti, prendendo come esempi le due strategie più comunemente impiegate. Tuttavia, simili approcci possono essere adottati per strategie che tengono conto di caratteristiche quali la sicurezza, la proprietà e l'insieme di capacità.

Strutturazione a livelli pluridimensionale

Le strategie precedentemente illustrate possono essere anche combinate per creare nuove strategie di strutturazione a livelli. L'esempio della Figura 12 mostra:

- due dei livelli basati sul riutilizzo presi dall'esempio precedente
  - applicazione
  - business
- tre livelli basati sulla responsabilità (tier)
  - logica di presentazione
  - logica di business
  - logica di accesso ai dati

Le dipendenze presenti nella strategia di strutturazione a livelli basata sul riutilizzo risultano solitamente da quelle tra elementi dei livelli della logica di business, come suggerisce la Figura 12, dove è visibile la dipendenza tra PersonalOrganizer e AddressBook.

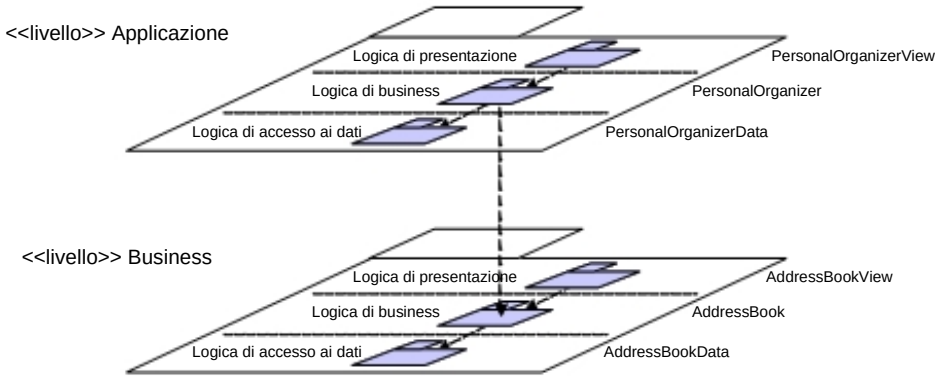

Figura 12: strutturazione a livelli pluridimensionale

Modellazione di livelli pluridimensionali

Consideriamo ora la rappresentazione degli aspetti pluridimensionali della strutturazione a livelli in un modello Progettazione bidimensionale. Prendiamo anche in esame una struttura in cui viene incorporato il concetto di componente business.

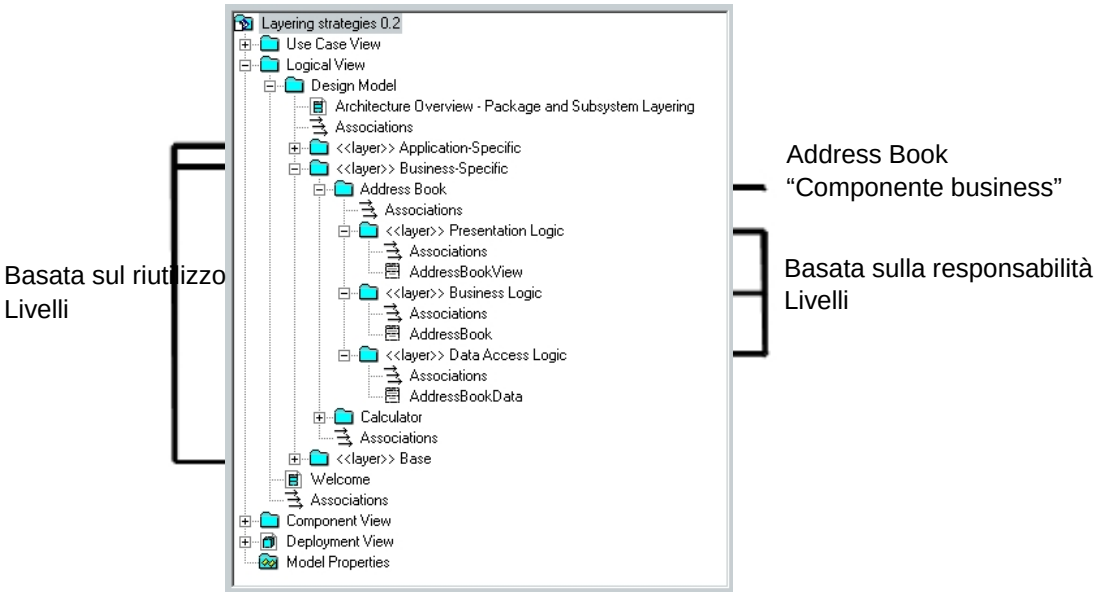

Figura 13: modello Progettazione che include strutturazione a livelli pluridimensionale

L'adozione di una strategia di strutturazione a livelli pluridimensionale richiede l'identificazione di una strategia principale. Nel nostro esempio, la strategia di strutturazione a livelli principale si basa sul riutilizzo. Il modello Progettazione è organizzato in primo luogo sulla strategia dei livelli Applicazione, Business e Base. Ciascuno di questi livelli è poi organizzato ulteriormente dagli elementi che si trovano in questi livelli; ad esempio, la Figura 13 mostra il livello Business che contiene Address Book e Calculator. Ognuno di questi elementi è poi organizzato a sua volta in funzione di una strategia secondaria: la strutturazione a livelli basata sulla responsabilità. Ad esempio, il pacchetto Address Book contiene tre livelli: Logica di presentazione, Logica di business e Logica di accesso ai dati.

Ciascuno dei livelli contiene tutti gli elementi che si trovano su questo livello:

- il livello logica di presentazione contiene la classe AddressBookView
- il livello logica di business contiene la classe AddressBook

= il livello logica di accesso ai dati contiene la classe AddressBookData

Conclusioni

Una delle decisioni più importanti che un architetto deve prendere consiste nella scelta della strategia appropriata di strutturazione a livelli, la quale influenzerà notevolmente la struttura dei modelli prodotti. Ancora più significativo è comunque il fatto che i benefici del business,
come la possibilità di manutenzione e il riutilizzo, possono essere direttamente supportati dalla strategia di strutturazione a livelli scelta. Ad esempio, i sistemi di più facile manutenzione possono essere sviluppati se si scandiscono le diverse responsabilità del sistema adottando una strategia di strutturazione a livelli basata sulla responsabilità. Inoltre, gli elementi riutilizzabili del sistema possono essere chiaramente identificati mediante una strategia di strutturazione a livelli basata sul riutilizzo.

Ringraziamenti

L'autore desidera ringraziare i suoi collaboratori Kelli Houston, Wojtek Kozaczynski, Philippe Kruchten, Bran Selic e Catherine Southwood (tutti di Rational Software) per le loro preziose osservazioni sulla bozza del documento.

Bibliografia

[Buschmann] Buschmann, Frank, et al. A System of Patterns. 1996. New York: John Wiley & Sons. ISBN 0-471-95869-7.

[Edwards] Edwards, Jeri. 3-Tier Client/Server at Work. 1999. New York: John Wiley & Sons. ISBN 0-471-31502-8.

[Eeles] Eeles, Peter e Oliver Sims. Building Business Objects. 1998. New York: John Wiley & Sons. ISBN 0-471-19176-0.

[Herzum] Herzum, Peter e Oliver Sims. The Business Component Factory. 2000. New York: John Wiley & Sons.

[Jacobson] Jacobson, Ivar, et al. Software Reuse. 1997. Reading, Massachusetts: Addison-Wesley. ISBN 0-201-92476-5.

[PLoP2] Vlissides, John, James Coplien e Norman Kerth. Pattern Languages of Program Design 2. 1996. Reading, Massachusetts: Addison-Wesley. ISBN 0-201-89527-7.

Rational®
the software development company

Sedi principali:

Rational Software
18880 Homestead Road
Cupertino, CA 95014
Tel: (408) 863-9900

Rational Software
20 Maguire Road
Lexington, MA 02421
Tel: (781) 676-2400

Numero verde: (800) 728-1212
E-mail: info@rational.com
Sito Web: www.rational.com
Sito internazionale: www.rational.com/worldwide