Luca Cabibbo

Analisi e Progettazione del Software

Alcune idee sui sistemi software e la loro architettura

Capitolo 92

marzo 2017

Gli orchi sono come le cipolle.
Le cipolle hanno gli strati.
Gli orchi hanno gli strati.

Shrek

A P S * Architettura a strati

I sistemi informatici (ovvero, le applicazioni software) sono comunemente composti da diversi elementi software (classi, package, ...) – spesso questi sistemi vengono organizzati secondo un'*architettura a strati*

- un'applicazione è composta da una pila verticale di strati
- ogni strato comprende diversi elementi software – in particolare, uno o più package, ciascuno dei quali è composto da più classi
- comunemente, l'utente interagisce con l'applicazione tramite il suo strato più alto – che si occupa proprio di implementare l'interfaccia utente
  - le richieste dell'utente vengono via via propagate (ovvero, elaborate e trasformate in altre richieste) da questo strato verso gli strati più bassi
  - inoltre, le risposte prodotte vengono fatte risalire (ovvero, elaborate e trasformate in altre risposte) tra gli strati, fino ad arrivare all'utente

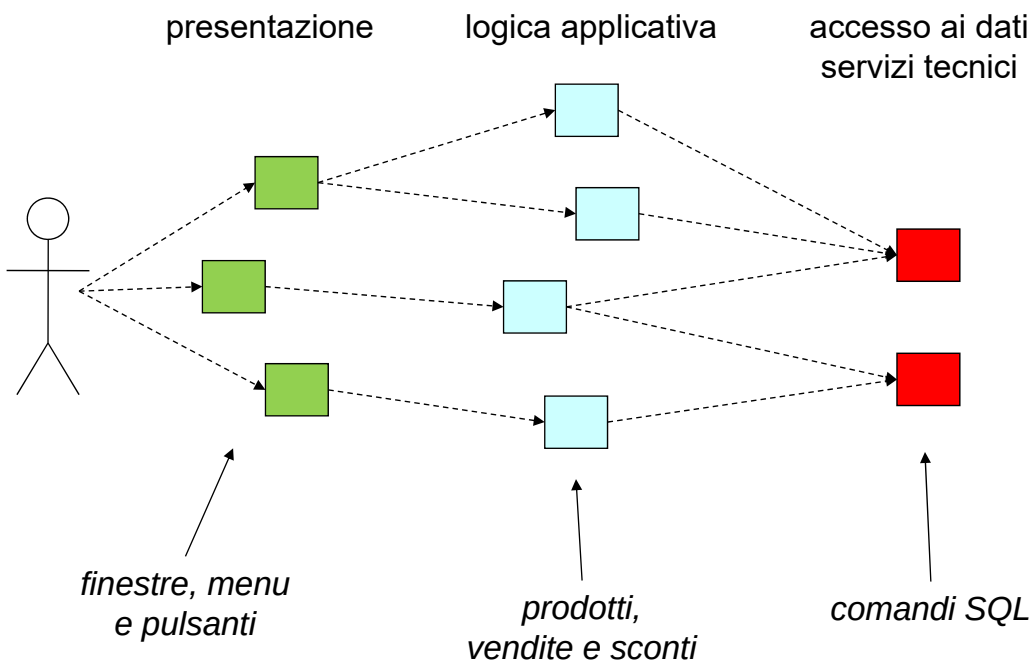
A P S * Architettura a strati

Una possibile scelta (minimale ma comune) per gli strati

- presentazione
- logica applicativa
- accesso alla base di dati e altri servizi tecnici

A P S * Architettura a strati

Architettura a strati



A P S * Sullo strato della logica applicativa

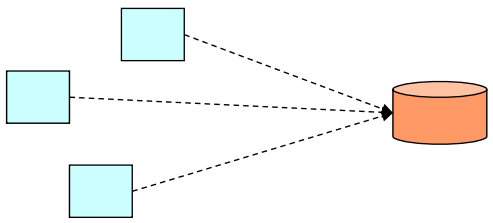
Esistono diverse strategie per l'organizzazione dello strato della logica applicativa di un sistema software – che si diversificano tra loro su come organizzano dati e operazioni

Attenzione: **la scelta della strategia adottata per lo strato della logica applicativa ha un impatto profondo sul codice, ma anche su come va svolta l'analisi e la progettazione**

A P S * Sullo strato della logica applicativa

Alcune strategie principali per l'organizzazione dello strato della logica applicativa

- **approccio "transazionale"**
  - i dati sono gestiti in una base di dati
  - le operazioni sono transazioni sulla base di dati – ciascun oggetto/classe della logica applicativa corrisponde a una procedura/transazione che l'utente può richiedere al sistema

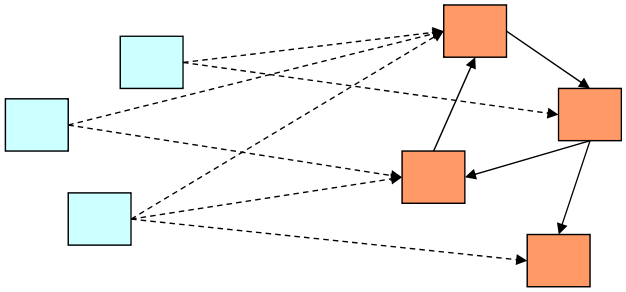


servizi (transazioni) base di dati

A P S * Sullo strato della logica applicativa

Alcune strategie principali per l'organizzazione dello strato della logica applicativa

- **approccio "procedurale"**
  - i dati sono gestiti in memoria principale
  - alcuni oggetti sono usati per rappresentare dati/informazioni
  - altri oggetti (separati) definiscono le operazioni che possono essere applicate alle informazioni



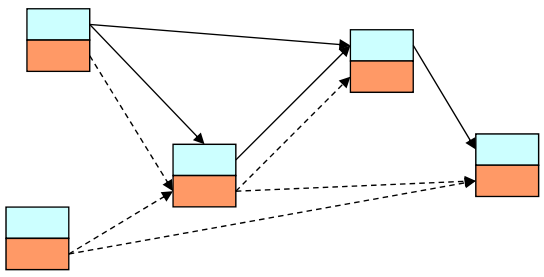
oggetti servizi (procedure)

oggetti dati (record e puntatori)

A P S * Sullo strato della logica applicativa

Alcune strategie principali per l'organizzazione dello strato della logica applicativa

- ***Domain Model***
  - secondo questa strategia, lo strato della logica applicativa è realizzato a oggetti – e la maggior parte di questi oggetti di dominio incapsulano sia dati che operazioni – ripartendosi le responsabilità del sistema
  - lo strato della logica applicativa viene anche chiamato **strato di dominio**



oggetti (con dati e operazioni)

A P S * Sullo strato della logica applicativa

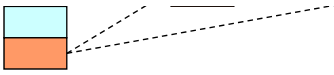
Alcune strategie principali per l'organizzazione dello strato della logica applicativa

- ***Domain Model***

va è
i di
osi le

trato

Per quanto riguarda l'organizzazione della logica applicativa, questo corso fa riferimento alla strategia *Domain Model*



oggetti (con dati e operazioni)

A P S * Applicazioni stand-alone e client-server

In generale, un'applicazione software si occupa di offrire ai suoi utenti

- l'esecuzione di un certo numero di *funzionalità*
- solitamente, relative alla gestione di alcune tipologie di *informazioni* (*dati*)

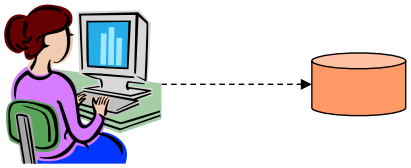
È utile distinguere tra due tipologie di applicazioni software

- **applicazioni stand-alone**
  - si tratta di applicazioni mono-utente, i cui dati non sono condivisi tra utenti diversi della stessa applicazione
- **applicazioni client-server**
  - si tratta di applicazioni che possono essere accedute in modo concorrente da più utenti, tramite un accesso in rete o sul web
  - queste applicazioni di solito gestiscono anche dati che devono essere condivisi dai diversi utenti

A P S * Applicazioni stand-alone

In un'applicazione stand-alone, mono-utente, i dati non sono condivisi tra utenti diversi della stessa applicazione

- l'applicazione è utilizzata da un utente, per gestire i propri dati, memorizzati localmente, nel computer dell'utente
- ad esempio, un'applicazione che consente all'utente di gestire la sua rubrica, memorizzata su un file del computer

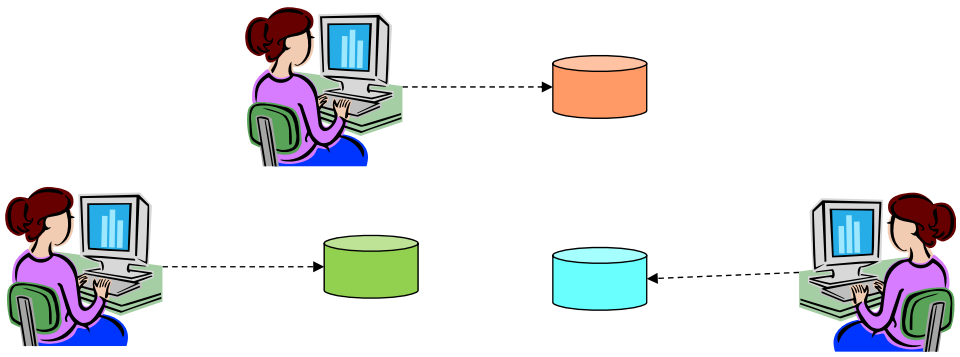


- tra i nostri studi di caso, è il caso del gioco del Monopoly

A P S * Applicazioni stand-alone

È anche possibile che la stessa applicazione stand-alone venga usata, contemporaneamente, da più utenti, ciascuno sul suo computer

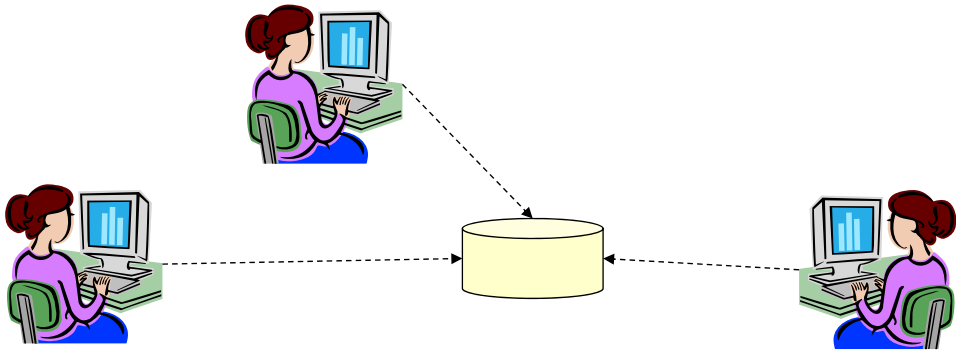
- in questo caso, ciascuna istanza/esecuzione dell'applicazione gestisce i propri dati – i dati di istanze/esecuzioni diverse dell'applicazione non sono mai "mischiati"



A P S * Applicazioni client-server

Un'applicazione client-server può essere utilizzata da più utenti (detti client) in modo concorrente – mediante un accesso in rete o sul web

- di solito, i dati gestiti dall'applicazione non sono memorizzati nel computer dell'utente, ma piuttosto vengono gestiti, in modo condiviso, in una base di dati (unica) per l'applicazione
- ad esempio, il portale dello studente



- tra i nostri studi di caso, è il caso del sistema POS – infatti nel negozio ci possono essere più registratori di cassa

A P S * Progettazione per applicazioni stand-alone

È utile discutere alcune implicazioni del tipo di applicazione software (stand-alone o client-server) sul modo di farne analisi e progettazione

- il caso di un'applicazione stand-alone è relativamente semplice
  - è infatti sufficiente pensare a una singola istanza/esecuzione dell'applicazione, dal punto di vista di un singolo utente
  - ad esempio, nel gioco del Monopoly si farà riferimento a una singola partita, giocata su un singolo tabellone, da un certo numero di giocatori virtuali (che "esistono" solo in quella partita)

A P S * Progettazione per applicazioni client-server

Il caso di un'applicazione client-server è più complesso

- ad es., un sistema per le prenotazioni agli esami – può essere usato in modo concorrente da due o più studenti (diversi) che vogliono prenotarsi a due o più esami (diversi, o anche lo stesso)
- da una parte, l'applicazione deve gestire alcuni dati condivisi tra tutti i suoi utenti/client
  - ad es., l'insieme degli studenti, dei corsi e degli appelli
- inoltre, per ciascun utente/client, l'applicazione deve gestire alcuni dati specifici per la conversazione/sessione con quel particolare client
  - ad es., quale studente sta usando il sistema? quale corso ha selezionato? quale appello ha selezionato?
- si noti il "per ciascun utente/client"
  - complessivamente l'applicazione deve gestire i dati delle conversazioni/sessioni con tutti i suoi utenti/client

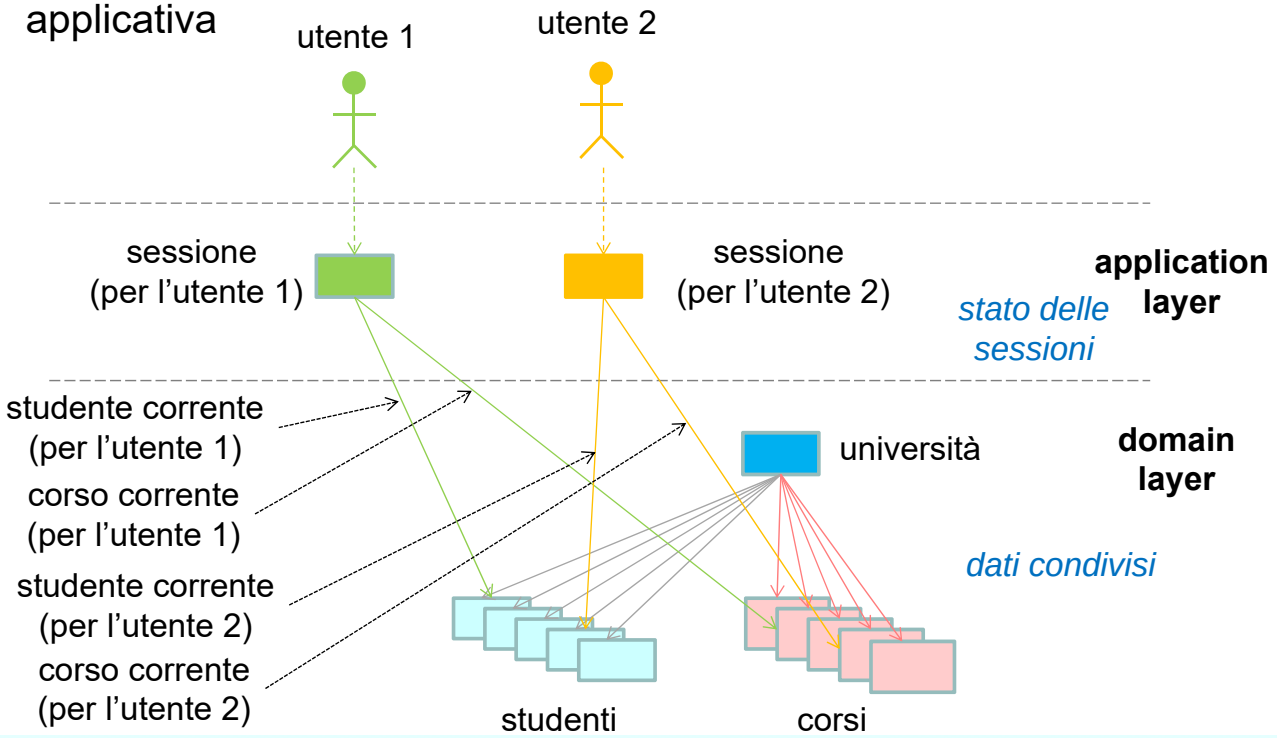
A P S * Progettazione per applicazioni client-server

Il caso di un'applicazione client-server è più complesso

- dunque, un'applicazione client/server deve gestire
  - i **dati condivisi** da tutti gli utenti – comunemente, questi dati vengono gestiti mediante una base di dati
  - i dati relativi allo stato delle conversazioni/sessioni con tutti i singoli utenti/client dell'applicazione – ovvero lo **stato delle sessioni** – questo può essere fatto in modi diversi, lato server oppure lato client
- per quanto riguarda lo stato delle sessioni, in questo corso ipotizziamo di utilizzare una tecnologia che consenta di ragionare (e scrivere i programmi) in termini di una singola conversazione/sessione – ovvero, in termini dello **stato della sessione di un singolo utente/client**
  - la gestione dello stato di tutte le sessioni per una molteplicità di utenti/client concorrenti è poi implementata dalla tecnologia

A P S * Progettazione per applicazioni client-server

Una soluzione comune è l'introduzione di un ulteriore strato ("application") tra quello di presentazione e quello della logica applicativa



A P S * Progettazione per applicazioni client-server

In pratica, nello sviluppo delle applicazioni client-server

- nell'analisi e progettazione, si devono fare ragionamenti opportuni (e separati) per quanto riguarda la gestione dei dati condivisi dell'applicazione e la gestione dello stato della sessione di un client
- inoltre, nella programmazione per queste applicazioni si dovranno usare delle opportune tecnologie, e si dovranno prendere in considerazione diversamente quelle parti del progetto che riguardano la gestione dei dati condivisi da quelle che riguardano la gestione dello stato della sessione
- alcuni approfondimenti su questo nei corsi di ***Sistemi informativi su web*** e di ***Architettura dei sistemi software***
  - ma anche, brevemente, alla fine di questo corso
  - purtroppo, questo argomento è considerato solo in modo parziale nel libro di Larman