Comparative analysis of software architectures: MVC, Layered and Pipe&Filter

Analisi comparative delle architetture software: MVC, Layered e Pipe&Filter

Tesi di Laurea di

Francesco De Patre

Relatore

Chiar.mo Prof. Ing. A. Poggi

Architettura Model-View-Controller (MVC)

Model-View-Controller (MVC) è un pattern architetturale per lo sviluppo software che suddivide un'applicazione in tre componenti principali: Model, View e Controller.

Il componente Model si occupa di rappresentare i dati e la logica di business, il componente View gestisce l'interfaccia utente e la presentazione dei dati, mentre il Controller coordina le interazioni tra Model e View. Questo modello favorisce la separazione delle responsabilità, promuove la modularità del codice e facilita lo sviluppo collaborativo dell'applicazione.

Vantaggi e Svantaggi dell'Architettura Model-View-Controller (MVC)

- Vantaggi:
  - Maggiore scalabilità
  - Migliore testabilità
  - Complessità ridotta

- Svantaggi:
  - La riusabilità compromessa se l'architettura non è sviluppata correttamente.
  - Possibilità di overhead di comunicazione
  - Rischio codice non strutturato e disorganizzato

Architettura Layered

L'architettura Layered è un layout strutturale che suddivide un'applicazione in diversi livelli o strati, ciascuno dei quali svolge compiti specifici:

- Il Presentation Layer gestisce le interazioni utente e la presentazione dei dati, fornendo un'interfaccia visiva per l'utente finale.
- L'Application Layer fornisce un'interfaccia tra il presentation layer e il domain layer, gestendo la logica di business dell'applicazione.
- Il Domain Layer rappresenta il nucleo concettuale e funzionale dell'applicazione, definendo le entità, la logica aziendale e le regole di accesso ai dati.
- Infine, l'Infrastructure Layer si occupa della gestione delle risorse di sistema fondamentali, come il file system, il database e le connessioni di rete.

Vantaggi e Svantaggi dell'Architettura Layered

- Vantaggi
  - Maggiore modularità
  - Chiara separazione delle responsabilità
  - Maggiore manutenibilità

- Svantaggi
  - Prestazioni ridotte
  - Difficoltà nell’incapsulamento dei dati

Architettura Pipe&Filter

L'architettura Pipe & Filter è un modello organizzativo per lo sviluppo software che gestisce il flusso dei dati attraverso una sequenza di componenti di elaborazione, noti come filtri, collegati tramite pipe.

I componenti principali di questa architettura includono:

- Pipe: Canali attraverso i quali i dati vengono trasmessi tra i filtri, consentendo il passaggio fluido dei dati attraverso il sistema.
- Filter: Componenti responsabili dell'elaborazione dei dati, eseguendo operazioni specifiche su di essi.
- Pump: Componente che controlla il flusso dei dati, leggendo i dati da una sorgente esterna e inviandoli al primo filtro nella pipeline.
- Sink: Componente specializzato che riceve i dati dal filtro precedente e li elabora in modo specifico, terminando il flusso di dati o inviandoli a un altro sistema per un utilizzo finale.

Questo modello favorisce la modularità, la riusabilità e la tolleranza ai guasti, consentendo una maggiore flessibilità e facilità nella gestione dei processi di elaborazione dei dati all'interno del sistema.

Vantaggi e Svantaggi dell'Architettura Pipe&Filter

- Vantaggi
  - Elaborazione parallela dei dati
  - Elevata tolleranza ai guasti
  - Testing indipendente dei filtri

- Svantaggi
  - Maggiore complessità
  - Problemi di scalabilità

Caso d’uso

Il caso d'uso illustra l'applicazione pratica delle architetture MVC, Layered e Pipe & Filter nell'automazione del processo di elaborazione dei dati meteorologici.

L’applicazione è suddivisa in tre versioni che adottano le diverse architetture.

Il caso d’uso si occupa di analizzare come ciascuna architettura gestisca il flusso dei dati e offra vantaggi distinti nell'ambito dell'elaborazione dei dati.

Analisi dei Risultati

Confrontando le analisi delle tre versioni, emergono significative disparità in termini di utilizzo di memoria, numero di classi impiegate, gestione dei thread e utilizzo della CPU.

L'architettura Pipe&Filter si distingue per un consumo di memoria e una quantità di classi impiegate intermedie rispetto alle versioni MVC e Layered, mostrando un utilizzo della CPU inferiore rispetto all'architettura MVC, ma superiore rispetto all'architettura Layered.

Conclusioni

In conclusione, le architetture software MVC, Layered e Pipe&Filter offrono approcci distinti alla progettazione di sistemi complessi, la scelta dell'architettura più adatta dipende dalle esigenze specifiche del progetto, comprese le considerazioni sulla complessità del dominio, i requisiti di performance, la scalabilità e la manutenibilità del sistema.

Ciascuna architettura presenta vantaggi e svantaggi unici, e la decisione finale dovrebbe essere guidata dalla comprensione approfondita dei requisiti del progetto e delle caratteristiche delle diverse architetture disponibili.