Web Application Engineering
Project Management & Design

cristian lucchesi
IIT-CNR

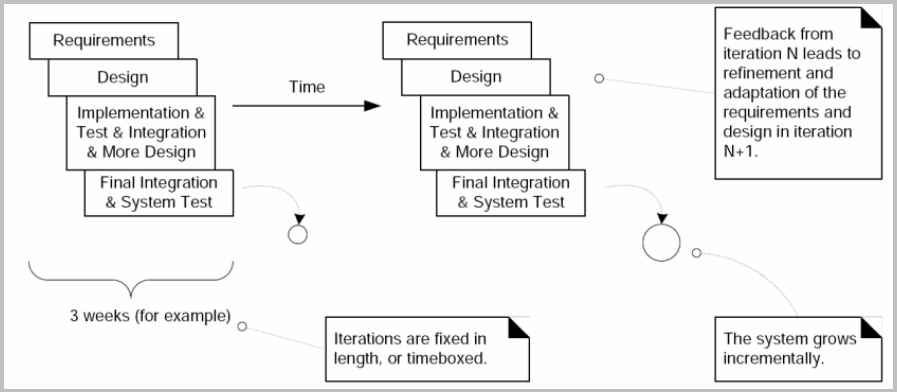
Pescara, 15-16 Maggio 2007
Alei – Ud'A

Caratteristiche Modello evolutivo

Sviluppo iterativo e incrementale

- lo sviluppo è organizzato in mini-progetti brevi, chiamati iterazioni
- il risultato di ciascuna iterazione è un sistema software eseguibile, ma parziale
- ciascuna iterazione comprende molte attività: requisiti, progettazione, implementazione, integrazione, verifica, ...
- il software parziale prodotto consente di ottenere feedback tempestivo dalle parti interessate (clienti e sviluppatori)
- il sistema cresce in modo incrementale da un'iterazione alla successiva, adattandosi ai requisiti, anche sulla base del feedback delle iterazioni precedenti
- il processo converge verso un sistema completo dopo varie iterazioni

Modello evolutivo



Configuration e project Management

- è necessario gestire le varie iterazioni del ciclo di sviluppo in modo controllato
- le attività di controllo e supporto allo sviluppo ed ai cambiamenti sono chiamate Web Configuration Management (WCM) e comprendono:
  - identificazione dei componenti del sistema
  - controllo delle versioni del codice
  - controllo dei cambiamenti richiesti ed effettuati
  - gestione della tempistica e del piano di lavoro
- esistono già delle pratiche consolidate nello sviluppo tradizionale chiamate Software Configuration Management (SCM) e Project Management, queste possono essere utilizzate nel WCM per gestire correttamente i progetti

Trac

- Trac è un web-based software project management e bug/issue tracking system che enfattizza la semplicità di utilizzo e la bassa burocrazia
- fornisce:
  - un wiki integrato
  - un'interfaccia al sistema di versionamento integrata con il sistema
  - la gestione del tracciamento dei bug e delle richieste tramite il meccanismo dei ticket
  - la gestione della roadmap, della timeline e dei componenti
- Trac è distribuito con licenza BSD, ed è scaricabile gratuitamente all'indirizzo http://trac.edgewall.org/

Object orientation

- object orientation è il prodotto di anni di esperienza nel cercare un modo efficacie di sviluppare il software
- è di sicuro il metodo più utilizzato nella maggior parte dei sistemi software attuali
- è un tecnica di modellare un sistemi del mondo reale in un oggetti software
- l'oggetto è il concetto fondamentale, è un modello software di un'entità o un concetto del mondo reale
- l'object orientation è utilizzata anche nella maggior parte delle Web application

Object Oriented Design

- progettare un sistema OO consiste in:
  - identificare gli oggetti contenuti dal sistema
  - definire comportamento e responsabilità dei vari oggetti
  - identificare come gli oggetti interagiscono
- con l'Object Orientation si può:
  - produrre programmi più semplici da progettare e quindi da capire
  - avere oggetti singoli che spesso possono essere implementati e debuggati indipendentemente
  - avere librerie di oggetti più facilmente riusabili e adattabili a nuovi design
  - avere programmi più facilmente modificabili e resistenti all'introduzione di bug durante la modifica ed il mantenimento

B.E.Wampler, The Essence of Object Oriented Programmin with Java and UML

Analisi degli oggetti

Fasi:

- identificazione delle classi
- identificazione delle associazioni
- identificazione degli attributi
- ereditarietà
- iterazione del procedimento

Analisi degli oggetti

cont.

Identificazione delle classi:

- elencare le classi candidate a partire dal Problem Statement e/o dai casi d'uso evidenziando tutti i sostantivi che ne fanno parte
- evitare costrutti tipici dell'implementazione (a meno che la problematica non sia prettamente informatica)
- eliminare le classi "scorrette"

Analisi degli oggetti

cont.

Criteri di eliminazione delle classi:

- classi ridondanti: due classi esprimono la stessa informazione (es: cliente ed utente)
- classi irrilevanti: classi che sono marginali all'interno del Problem Statement
- attributi: i nomi che descrivono oggetti singoli dovrebbero essere riformulati come attributi (es: età di una persona)
- operazioni: se un nome descrive un'operazione applicata ad oggetti e non viene esso stesso manipolato (es: telefonata)
- ruoli: il nome di una classe dovrebbe riflettere la sua reale natura e non il ruolo che ricopre in un'associazione (es: marito)

Analisi degli oggetti

cont.

Identificazione delle associazioni tra le classi

- estrarre tutti i verbi candidati dalla formulazione del Problem Statement e/o dei casi d'uso
- qualunque dipendenza verbale tra due o più classi è un'associazione
- un riferimento da una classe all'altra è un'associazione

Analisi degli oggetti

cont.

Identificazione degli attributi

- gli attributi sono proprietà di singoli oggetti (es: età, nome, colore, peso ...)
- gli attributi, generalmente, corrispondono a sostantivi seguiti dal complemento di specificazione (es: il colore dell'auto)
- se un attributo descrive uno stato interno di un oggetto che è invisibile all'esterno, eliminare l'attributo dal modello

Analisi degli oggetti

cont.

Ereditarietà

- organizzare le classi usando l'ereditarietà per condividere le proprietà comuni
- l'ereditarietà può essere raggiunta in due modi:
  - generalizzando aspetti comuni di classi esistenti in una superclasse
  - specializzando classi esistenti in sottoclassi

Analisi degli oggetti

cont.

Procedimento iterativo

- l'analisi degli oggetti è costituita da continue iterazioni successive
- nel caso ci siano parole che possono essere soggette ad interpretazioni differenti è necessario compilare un glossario per tutte le classi rilevate nella fase precedente

Analisi degli oggetti

cont.

Risultato dell'Analisi degli Oggetti

- un diagramma delle classi (UML) che dovrebbe rappresentare gli oggetti del dominio (senza riferimento ad aspetti implementativi)
- oppure direttamente le classi nel linguaggio di programmazione prescelto
- alcuni oggetti necessari per il controllo dell'applicazione potrebbero non essere parte dell'analisi iniziale (es.: interfacce)
- alcuni oggetti di controllo potrebbero essere introdotti come risultato dell'analisi dinamica

UML in due righe

- Unified Modelling Language (UML) è un linguaggio di modellazione di supporto a tutte le attività relative alla produzione di software
- si basa su simboli grafici e costrutti testuali

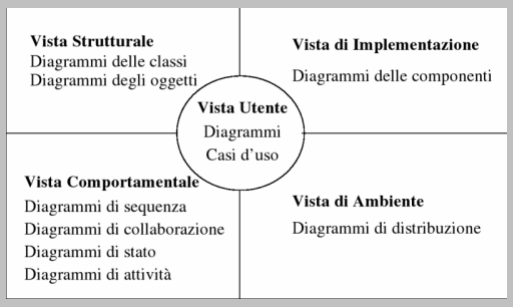
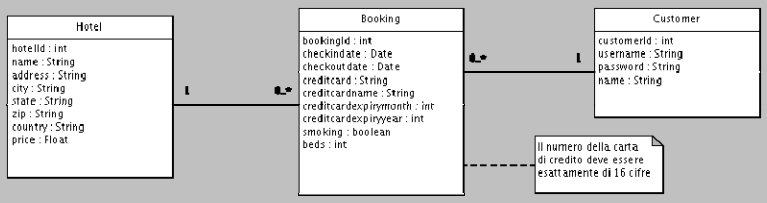


Diagramma delle classi

- esprime le classi che rappresentano gli oggetti con gli attributi ed i metodi (non presenti nell'esempio)
- esprime le associazioni tra le classi con la relativa cardinalità



Pattern

- è importante comprendere gli errori più comuni perché:
  - può aiutare a non ripeterli
  - aiuta lo sviluppo di una metodologia sempre più forte
- le metodologie più stabili nascono da un reale contesto di lavoro e sono definite durante il loro uso
- i pattern sono la descrizione della soluzione di un problema ricorrente, soluzione nata dall'esperienza nell'uso comune

Pattern software

- lo scopo di un pattern software è:
  - condividere una soluzione provata e ampiamente applicabile ad un particolare problema di progettazione,
  - in una forma standard che possa essere facilmente riusata
- una possibile descrizione strutturata:
  - nome
  - interessi - la situazione in cui il pattern può essere applicato
  - problema e forze in gioco
  - soluzione
  - conseguenze - risultati (positivi e negativi) e compromessi

Categorie Pattern

- architetturale
  - esprime una schema per l'organizzazione strutturale
  - fornisce un insieme di tipi di elementi predefiniti, specifica le loro responsabilità, comprende consigli per organizzarli
  - esempi: layers, client/server, peer-to-peer
- design
  - fornisce uno schema per raffinare gli elementi di un sistema software o le relazioni tra di essi
  - descrive una struttura che ricorre comunemente, risolve un problema di progettazione generale
  - esempi: Singleton, Adapter, Proxy
- idioma
  - è un pattern di basso livello, specifico di un linguaggio di programmazione
  - per esempio come implementare il Singleton in Java

Ruolo dei pattern

- alcuni dei ruoli svolti dai pattern:
  - deposito di conoscenza
  - esempi di buone pratiche
  - un linguaggio per discutere problemi di progettazione
  - un aiuto alla standardizzazione
  - incoraggiamento alla generalità
- il ruolo principale dal punto di vista delle architetture software?
  - riduzione del rischio
  - incremento della produttività, della standardizzazione e della qualità

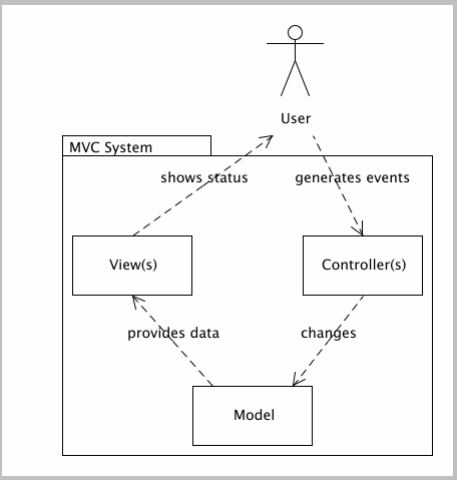
Model View Controller (MVC)

Model-view-controller (MVC) is an architectural pattern used in software engineering. In complex computer applications that present lots of data to the user, one often wishes to separate data (model) and user interface (view) concerns, so that changes to the user interface do not affect the data handling, and that the data can be reorganized without changing the user interface. The model-view-controller solves this problem by decoupling data access and business logic from data presentation and user interaction, by introducing an intermediate component: the controller.

From Wikipedia, the free encyclopedia

Model View Controller (MVC)

cont.



- l'MVC è un pattern architetturale, la comunicazione tra gli oggetti non è ulteriormente specificata

Model View Controller (MVC)

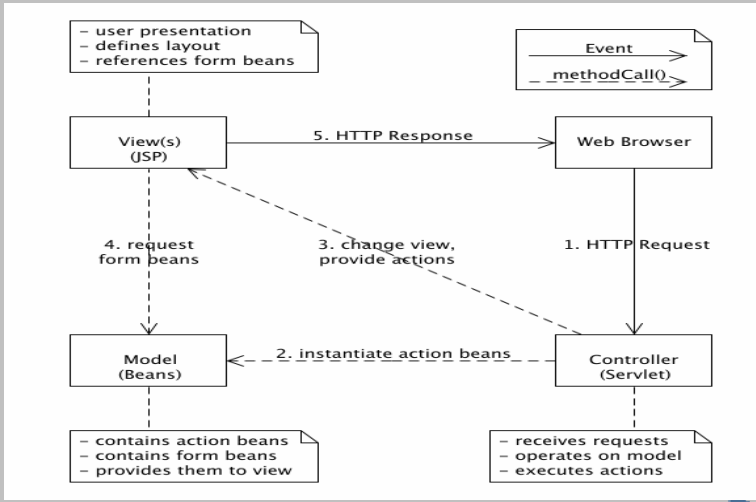
cont.

- il cuore dell architettura MVC sta nella **Presentazione Separata**
- viene definita una chiara separazione tra:
  - gli oggetti del dominio (model objects)
  - gli oggetti di presentazione: gli elementi dell'interfaccia grafica che vediamo a schermo (view objects)
- l'MVC ha le sue radici in SmallTalk, dove era originariamente applicato per trattare l'input tradizionale dell'utente, l'elaborazione e l'output mostrato all'utente in un interfaccia grafica

MVC: partecipanti e responsabilità

- **Model**
  - il modello rappresenta i dati e le regole che gestiscono il loro accesso e la loro modifica
- **View**
  - mostra il contenuto del modello
  - accede ai dati del modello e specifica come mostrarli
  - ha la responsabilità di mantenere coerente la presentazione quando i dati rappresentati dal modello cambiano. Può essere ottenuto tramite due diversi modelli:
    - Push model - la view registra se stessa presso il modello per la notifica dei cambiamenti
    - Pull model - è responsabilità della view di chiamare il modello quando ha bisogno di dati più aggiornati
- **Controller**
  - trasla le interazioni con la view in azioni eseguite sul modello
  - in una applicazione web le interazioni con la view sono l'invio di richieste HTTP GET o POST dopo aver selezionato un link o inviato una form
  - le azioni sul modello includono l'aggiornamento dei dati e l'attivazione di procedure
  - in funzione delle interazioni dell'utente e del risultato delle operazioni sul modello il controller risponde selezionando la view appropriata

MVC: Model 2 in Java



MVC: vantaggi e svantaggi

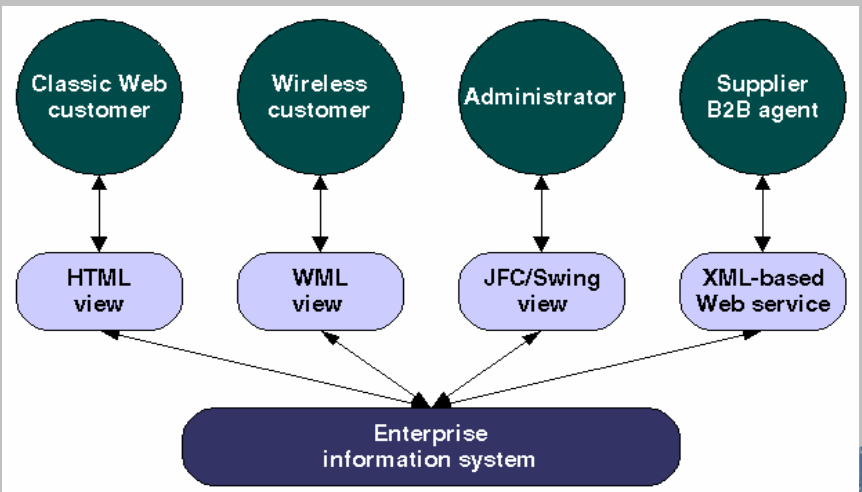
Vantaggi

- ri-uso dei componenti del modello
  - la separazione di model e view permette a molte view di usare gli stessi oggetti del modello
  - i componenti del modello sono più facilmente implementabili, testabili e mantenibili perché ogni accesso ai dati del modello passano da questi componenti
- più facile supporto per nuovi tipi di client
  - per supportare un nuovo tipo di client è sufficiente scrivere nuove view e alcuni controller utiilizzando i soliti oggetti del model

Svantaggi

- aumento della complessità dell'architettura
  - questo pattern introduce alcune classi aggiuntive dovuto alla seperazione tra model, view e controller

MVC: vari tipi di utenti



Framework

- un software framework è un architettura ri-utilizzabile per un sistema software
- un framework di solito include:
  - programmi di supporto
  - librerie
  - un linguaggio di scripting
  - componenti software che permettono di "incollare" insieme i differenti componenti del progetto
- i framework facilitano lo sviluppo rapido di applicationi permettendo ai programmatori di utilizzare funzionalità ad alto livello

Scrivere le applicazioni Web

- Ci sono molti Web application framework che permettono di:
  - definire una descrizione ad alto livello del programma
  - concentrarsi sul framework invece che sugli aspetti a basso livello
  - mantenere il codice più semplice
  - utilizzare librerie già pronte (e testate!) per risolvere problematiche comuni
  - diminuire le problematiche di sicurezza utilizzando metodi consolidati
  - promuovere l'utilizzo di best practices come per esempio "GET after POST"
  - ridurre il numero di errori nel programma

Java MVC Frameworks

- Struts è una dei primi web-application framework che fa largo uso del pattern MVC
  http://en.wikipedia.org/wiki/Struts
- Shale è un framework MVC che comprende funzionalità per costruire associare facilmente backing bean con le view, robusta validazione sia client che server side, template per le view
  http://shale.apache.org/
- The Spring Framework è un nuovo Java EE application framework. Spring utilizza un approccio multi-tier
  http://en.wikipedia.org/wiki/The_Spring_Framework
- Java Server Faces (JSF) framework è un Java EE standard web-application framework
  http://en.wikipedia.org/wiki/Java_Server_Faces
- …

PHP MVC frameworks

- Zend Framework, PHP 5 based MVC framework
  http://framework.zend.com
- Zoop Framework, PHP 4/5 MVC framework
  http://zoopframework.com
- Symfony Framework, PHP 5 MVC Framework
  http://www.symfony-project.com
- Switch Board with Routing, PHP 5 MVC Framework with Routing
  http://www.danielslaughter.com/projects/
- CakePHP, webapplication framework modellato successivamente ai concetti del Ruby on Rails
  http://cakephp.org
- PHP on Trax
  http://www.richardkmiller.com/files/john_peterson_trax_presentation.pdf

Post/Redirect/Get

- (PRG) is a common design pattern for web applications, to help avoid duplicate form submissions and allow applications to behave more intuitively with browser bookmarks and the reload button. After a web user submits a form to a server, the server typically generates an HTML page as a response. To the user, this looks like an ordinary web page, but because it was generated by an HTTP POST request, it cannot be bookmarked, and attempting to reload/refresh the page in the browser could cause the form information to be resubmitted, possibly with unexpected results (such as a duplicate purchase). To avoid this problem, many web applications use the PRG pattern - instead of returning an HTML page directly, the POST operation returns a redirection command (using the HTTP 303 response code (sometimes 302) together with the HTTP "Location" response header), instructing the browser to load a different page using an HTTP GET request. The result page can then safely be bookmarked or reloaded without unexpected side effects.

Wikipedia

Riferimenti

- Trac site **http://trac.edgewall.org/**
- Design Pattern: Elementi per il riuso di software a oggetti
- Best Practice Software Engineering **http://best-practice-software-engineering.ifs.tuwien.ac.at/**
- Java BluePrints - Model-View-Controller **http://java.sun.com/blueprints/patterns/MVC-detailed.html**

grazie per l'attenzione

cristian.lucchesi@iit.cnr.it