Progettazione di Applicazioni Web

Slides parzialmente tratte da materiale di Giansalvatore Mecca (*Tecnologie di Sviluppo per il Web*) e Marty Hall (*http://www.coreservlets.com*)

Argomenti della lezione

- Sviluppo delle applicazioni
  - Processo di sviluppo
  - Formalismi grafici di supporto
    - diagrammi UML (cenni)
  - Scelta dell'architettura
- Sviluppo di applicazioni WEB
  - Estensioni di UML per applicazioni WEB
  - Tecnologia J2EE e architetture di riferimento

Parte I: Sviluppo delle applicazioni

Il Processo di Sviluppo

- Studio di Fattibilità
- Analisi dei Requisiti
- Progettazione
- Sviluppo
- Test
- Installazione ed Uso
- Manutenzione

Formalismi di supporto allo sviluppo di applicazioni

- Diagrammi DFD
  - modella i processi, evidenziando gli archivi utilizzati
  - usati per la progettazione di applicazioni su DB
- UML (Unified Modelling Language)
  - un linguaggio per la modellazione di sistemi (software, ma non solo)
    - specifica
    - progettazione
    - documentazione
  - contiene vari tipi di diagrammi
    - consideriamo solo un sottoinsieme

Diagrammi di UML (alcuni)

- Diagrammi dei Casi d'Uso

  specificano il comportamento del sistema dal punto di vista degli utenti o degli altri sistemi con cui esso interagisce (come il sistema agisce e reagisce)
- Diagrammi delle Attività

  forniscono la sequenza di operazioni "elementari" che definiscono un'attività più complessa
- Diagrammi delle Classi

  descrivono le classi di oggetti che compongono il sistema, cioè gli aspetti che accomunano diversi oggetti nella loro struttura e nel loro comportamento

Esempio1: Indovina il Numero

Specifiche:

- Per cominciare il gioco, l'utente immette il suo nome
- Il calcolatore sceglie un numero casuale tra 1 e 100
- Il giocatore tenta di indovinarlo effettuando tentativi
- Per ciascun tentativo, il calcolatore risponde con un messaggio del tipo
  - "Prova con un numero più alto"
  - oppure "Prova con un numero più basso"
  - oppure "Hai indovinato"
- Il gioco deve tenere traccia di:
  - numero di tentativi effettuati dall'utente
  - record di tentativi (minimo numero di tentativi impiegati per indovinare dai vari utenti)

Esempio 2: Il SI dell'ACI

Specifiche:

- E' necessario gestire dati relativi ad una collezione di autovetture e ai loro proprietari
- Per ciascun proprietario è necessario rappresentare codice Fiscale, nome e città di residenza
- Per ciascuna autovettura è necessario rappresentare targa, modello, cilindrata e riferimento al proprietario
- L'applicazione deve consentire di effettuare le seguenti operazioni:
  - inserimento dei dati di un proprietario;
  - inserimento dei dati di un'automobile;
  - stampa dei dati dei proprietari e delle relative automobili

Analisi dei Requisiti

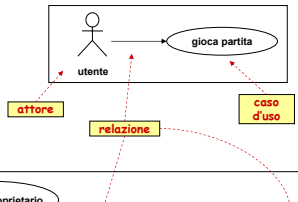
- Obiettivo
  - capire come funziona la realtà di interesse
  - capire come deve funzionare l'applicazione
  - studiare le funzioni dell'applicazione
  - studiare i dati dell'applicazione
- Un'utile strumento
  - i Casi d'Uso di UML ("Use Cases")

Casi d'Uso

- Caso d'Uso
  - funzionalità dell'applicazione utilizzabile dagli utenti
  - elencare i casi d'uso consente di
    - riassumere le funzionalità dell'applicazione
    - dividere l'analisi del sistema in blocchi
- Diagramma dei Casi d'Uso
  - strumento di UML per la rappresentazione dei casi d'uso
  - corrisponde al diagramma di contesto dei DFD

Diagramma dei casi d'uso: Esempi

- Indovina il Numero
  - un unico caso d'uso
  - un unico utente
- Il SI dell'ACI
  - vari casi d'uso



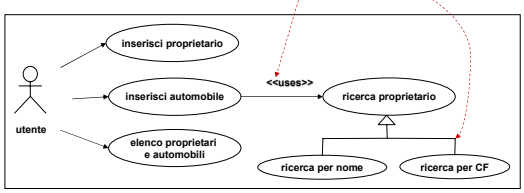
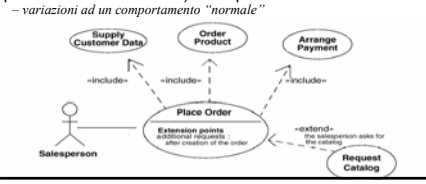


Diagramma dei Casi d'Uso: tipi di relazioni

- Partecipazione = freccia fra attore e caso d'uso
  - l'attore "compie" i casi d'uso cui è collegato
- Relazione = freccia tra due casi d'uso
  - uso (etichettata con <<*uses*>> o <<*includes*>>)
    A <<*includes*>> *B*: ogni istanza dello Use Case A includerà anche il comportamento specificato per lo Use Case B
  - estensione (etichettata con <<*extends*>>)
    A <<*extends*>> *B*: ogni istanza di B può includere (sotto condizioni specificate nell'estensione) il comportamento definito da A.
    - *variazioni ad un comportamento "normale"*



Casi d'Uso

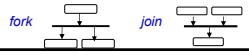
- Per ciascun caso d'uso
  - è necessario analizzare le operazioni corrispondenti
- Descrizione del caso d'uso
  - descrizione testuale
  - diagramma delle attività (Activity Diagram)

Descrizione dei Casi d'Uso

- Esempio: Gioca Partita
  - Attore: Utente
  - Obiettivo: Giocare una Partita
  - Pre-condizione: nessuna
  - Post-condizione: il giocatore ha indovinato il numero
- Descrizione testuale:
  - L'applicazione legge il nome dell'utente.
  - L'applicazione sceglie un numero a caso tra 1 e 100.
  - L'utente inserisce i suoi tentativi.
  - Per ogni tentativo, l'applicazione risponde con un messaggio del tipo "Prova con un numero più alto" oppure "Prova con un numero più basso".
  - Quando l'utente indovina l'applicazione verifica se ha battuto il record di tentativi, nel qual caso aggiorna il record, oppure lo ha eguagliato, e stampa un messaggio sullo schermo.

Diagrammi delle Attività

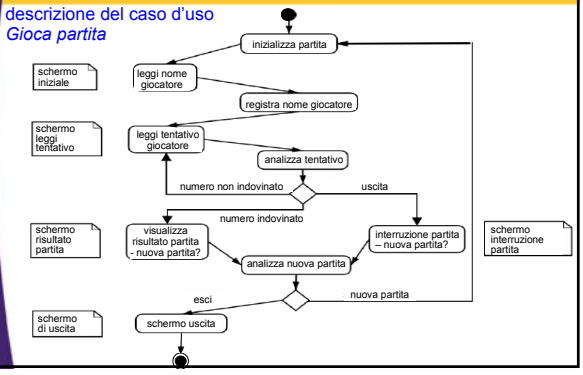
- Descrizione grafica dei casi d'uso
  - utile per evidenziare il flusso di controllo e l'interazione con l'interfaccia
  - simile ad un diagramma di flusso tradizionale
- Elementi
  - stati (i nodi del diagramma)
    - stati di attività 

    - inizio 
    - fine 
  - transizioni:
    - semplici
    - decisione 
    - concorrenza 

Diagrammi delle Attività

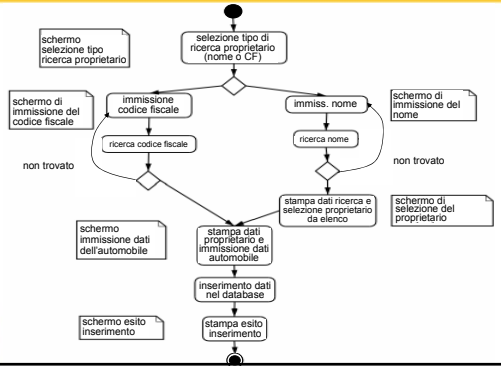
Esempio: Indovina il Numero

descrizione del caso d'uso
Gioca partita



Diagrammi delle Attività

Inserisci Automobile



Diagrammi delle Attività: corsie

Corsie
- associano le attività agli attori che devono eseguirle
- definiscono le responsabilità

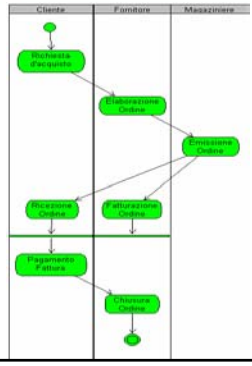


Diagramma di sequenza

- Scenario
  - uno specifico percorso all'interno di un caso d'uso
  - un caso d'uso comprende un insieme di percorsi alternativi che, durante la messa in opera del sistema, saranno seguiti a seconda delle condizioni che si verificheranno
- Diagramma di sequenza
  - rappresentazione grafica di uno scenario
  - descrive le interazioni degli oggetti ordinate secondo una sequenza temporale
- Descrizione dei casi d'uso con diagrammi di sequenza
  - la descrizione di un caso d'uso può essere resa più intuibile descrivendo vari scenari alternativi trattati all'interno di esso

Diagramma di sequenza

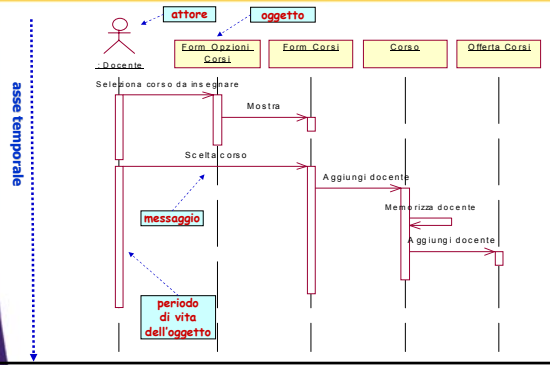
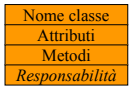


Diagramma delle Classi

- Descrive la struttura statica del sistema esaminato
  - evidenzia le classi, la loro struttura e le loro relazioni
  - può essere usato per descrivere
    - tipi di oggetti gestiti presenti nel sistema (*concettualizzazione*)
    - classi dell'implementazione software
- Ogni classe è rappresentata da un rettangolo
  - in cui sono indicati il nome della classe, i suoi attributi, i suoi metodi e le sue responsabilità (descrizione informale)



  - è possibile indicare la visibilità di attributi e metodi
    public (**+**), private (**-**), protected (**#**)

Diagramma delle Classi: relazioni tra Classi e tra Oggetti

Tre tipi di relazioni

- Generalizzazione - Specializzazione
- Composizione (Aggregazione)
- Associazione

Diagramma delle Classi

Esempio di Gen-Spec

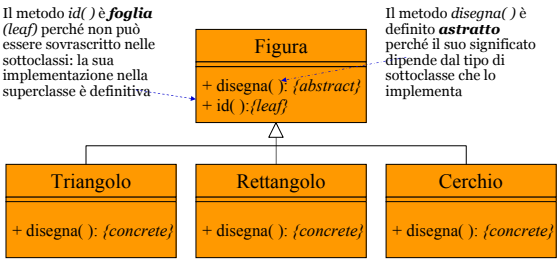
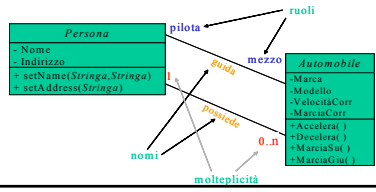


Diagramma delle Classi

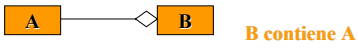
Associazione

- Esprime un legame "semantico" tra le classi
- Caratteristiche
  - nome
  - ruolo delle parti associate (facoltativo)
  - molteplicità (cardinalità della relazione)
    - notazione "inversa" rispetto al modello E/R
  - direzionalità ("verso di percorrenza" della relazione)



Composizione (aggregazione)

- Consente di rappresentare situazioni in cui una classe di oggetti si presenta come aggregazione di altri oggetti
  - l'oggetto composto può essere descritto come "somma" degli oggetti che lo compongono
- Due tipi di composizione
  - Lasca: il contenuto può esistere indipendentemente dal contenitore

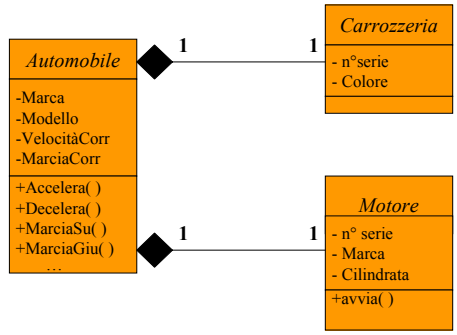


B contiene A

  - Stretta: l'oggetto contenuto esiste solo se esiste un'istanza corrispondente dell'oggetto contenitore

X contiene Y

Composizione: Esempio



Progettazione

- Segue l'analisi dei requisiti
- Guida la fase di sviluppo
- Obiettivi
  - definire l'architettura dell'applicazione
  - definire la struttura della base di dati
  - definire la struttura e i metodi delle classi

Architettura delle applicazioni

- Tipi di componenti di un'applicazione
  - Interfaccia (*View*)
    - gestisce comunicazione con l'utente
    - possibilità: console, grafica, HTML
  - Logica di controllo (*Control*)
    - gestisce il flusso di esecuzione fra le varie componenti
  - Modello (*Model*)
    - rappresenta lo stato dell'applicazione: dati legati all'esecuzione dell'applicazione
  - Persistenza
    - alcune informazioni devono essere memorizzati in modo persistente al di là delle singole esecuzioni
    - possibilità : base di dati, file di testo, file XML
- Varie architetture possibili

  le componenti concettuali possono essere implementate con componenti software distinte o meno

Architettura applicazioni: esempi

Indovina il Numero

- Interfaccia

  comunicazione con l'utente
- Logica di controllo

  gestisce i tentativi dell'utente
- Dati

  stato della partita e record attuale
- Persistenza

  il record del gioco deve essere persistente (file)

Il SI dell'ACI

- Interfaccia

  comunicazione con l'utente
- Logica di controllo

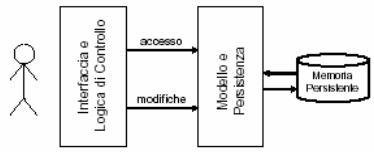
  gestisce le varie operazioni
- Dati

  automobili e proprietari
- Persistenza

  DB delle automobili e dei proprietari

Architettura di Base

- Architettura semplice
  - interfaccia e controllo assieme
  - modello e persistenza assieme
- Adatta per piccole applicazioni



Architettura di Base

- Interfaccia e Controllo
  - l'applicazione è fatta di una sequenza di schermi
    - schermo iniziale: menu di accesso ai vari casi d'uso
  - ogni schermo analizza l'input dell'utente e gestisce il flusso di controllo corrispondente
  - ciascuno schermo chiama il successivo
- Modello e Persistenza
  - il modello è fatto di un insieme di componenti che mantengono lo stato dell'applicazione
  - gli oggetti del modello sono modificati dall'interfaccia
  - gli oggetti del modello contengono codice per la gestione della persistenza (es: chiamate JDBC)

Architettura di Base

- Linee Guida
  - l'interfaccia ha accesso al modello (accede ai dati e li aggiorna)
  - il modello non ha accesso all'interfaccia
  - il modello ha accesso alla base di dati
  - l'interfaccia tipicamente non ha accesso alla base di dati (esegue le operazioni sul modello)
- JavaBeans
  - In un'applicazione Java, il modello è tipicamente composto di JavaBeans
  - permettono di disaccoppiare controllo e interfaccia dalla persistenza
    - i dati estratti dalla base di dati sono memorizzati in bean
    - i dati da inserire nella base di dati provengono da bean

Architettura di base: esempi

- Indovina il Numero
  - Un modulo per l'interfaccia e la logica di controllo
  - JavaBean
    - "Partita" che mantiene lo stato della partita
    - "Record" che mantiene il record, gestendo la persistenza in un file
- SI dell'ACI
  - Moduli per l'interfaccia e la logica di controllo
  - classe Proprietario.java
    - *codiceFiscale*, *nome*, *cittaDiResidenza*, lista delle sue automobili
    - inserimento nella base di dati
    - caricamento della lista delle automobili
  - classe Automobile.java
    - targa, cilindrata, modello, riferimento al proprietario
    - inserimento nella base di dati
  - classe Interrogazione.java
    - operazioni sul DB non riconducibili a proprietario o automobile

Architettura di base: critica

- Difetti
  - mischia interfaccia e controllo
  - modifiche all'interfaccia richiedono modifiche al controllo
    - l'interfaccia cambia abbastanza frequentemente
    - talvolta serve avere lo stesso controllo per interfacce diverse
- Sarebbe opportuno
  - rendere interfaccia e controllo indipendenti
  - poter cambiare l'interfaccia senza intaccare il controllo
- Soluzione: approccio "Modello-Vista-Controllo"
  - distingue i ruoli delle tre componenti dell'architettura

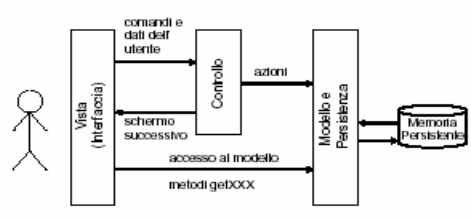
Architettura MVC

- Modello
  - componenti (JavaBeans) che rappresentano i dati e la logica del problema
- Vista
  - Interfaccia, sequenza di schermi per l'interazione con l'utente
- Controllo
  - logica per la gestione del flusso di controllo tra gli schermi e le operazioni sul modello

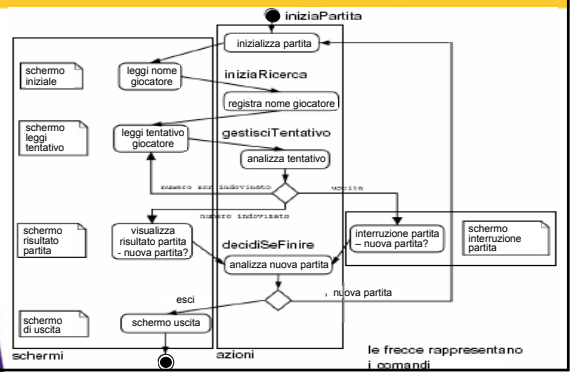
Architettura MVC

- Funzionamento
  - l'utente interagisce con gli schermi
  - ciascuno schermo raccoglie i dati dall'utente e invia comandi al controllo
  - il controllo analizza i dati dell'utente ed effettua le azioni relative sul modello
  - le azioni restituiscono un risultato
  - sulla base del risultato, il controllo s lo schermo successivo
  - lo schermo accede al modello per la visualizzazione
- Varie implementazioni possibili
  - vari modi di implementare il controllo
  - varie modalità di comunicazione tra vista e controllo

Architettura MVC



Architettura MVC: esempio

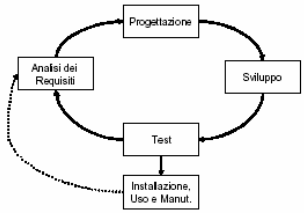


Architettura MVC

- Caratteristiche
  - interfaccia e controllo sono separate
  - l'interfaccia non modifica il modello
  - l'interfaccia non accede alla persistenza
- Vantaggi
  - la divisione tra schermi e azioni discende naturalmente dal diagramma delle attività
  - l'architettura complessiva risulta più modulare
  - è possibile cambiare l'interfaccia senza intaccare gli altri strati dell'applicazione
  - è possibile avere viste diverse sul modello
- Svantaggi
  - maggiore complessità del codice

Sviluppo di applicazioni: Metodologia di Riferimento

- Metodologia semplificata
  - adatta solo a piccole applicazioni
- Passi fondamentali
  - analisi dei requisiti
  - progettazione
  - sviluppo
  - test



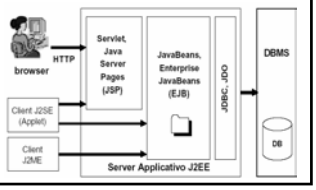
Sviluppo di applicazioni: Metodologia di Riferimento

- *I Passo*: Analisi dei Requisiti
  - Diagramma dei Casi d'Uso
    - descrizione dei Casi d'Uso
  - Diagramma delle Attività dei casi d'Uso
  - Selezione degli schermi dell'Interfaccia
  - Modello Concettuale dei Dati
- *II Passo*: Progettazione
  - Scelta dell'architettura dell'applicazione
  - Progetto della base di dati
  - Progetto delle classi dell'applicazione

Parte II: Sviluppo di applicazioni WEB

Tecnologia di riferimento

- J2EE
  - tecnologia orientata agli Oggetti per lo sviluppo di applicazioni Web
  - insieme di "componenti" (lato server) per la definizione della logica applicativa e dell'interfaccia
- Tipologie principali di componenti
  - JavaServer Pages
  - Servlet
  - JavaBeans
  - JDBC



Interfaccia

- Funzioni dell'interfaccia
  - interazione con l'utente
  - visualizzazione di dati
  - acquisizione di dati dall'utente
  - gestione degli "eventi" generati dall'utente
- Per le applicazioni Web: interfaccia basata su HTML
  - interazione con il browser
  - visualizzazione di messaggi in formato HTML
  - acquisizione dei dati con maschere HTML ("form")
  - due sole forme di evento
    - l'utente schiaccia il bottone di una form
      - richiesta *get* o *post* al server
    - l'utente seleziona un collegamento (…)
      - richiesta *get*

Logica Applicativa

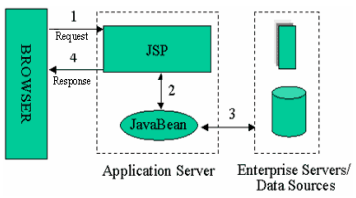
- Il server Web deve essere in grado di eseguire applicazioni (programmi)
  - Servlet e JSP, integrate con JavaBean e librerie
- Stato delle Sessioni
  - la sessione rappresenta una sequenza di operazioni effettuate da un singolo utente in una sessione di lavoro
    - Es.: "Indovina il numero" memorizza nome dell'utente, numero da indovinare, tentativi effettuati
- Stato dell'applicazione
  - per applicazioni multi-utente può essere utile mantenere dati sull'applicazione nel suo complesso, oltre ai dati di sessione (ogni sessione è relativa ad un singolo utente)
    - Es.: il numero di utenti che in quel momento stanno giocando

Architettura di applicazioni Web

- Due modelli di sviluppo basati sull'uso della tecnologia Servlet/JSP
  - si differenziano i tipi di componenti usati e per l'architettura complessiva dell'applicazione
- Modello JSP/1
  - implementazione dell'*"architettura di base"*
  - combina pagine JSP e Java Bean
    - separazione tra presentazione e logica applicativa
  - adatto per applicazioni di dimensioni medie
- Modello JSP/2
  - implementazione del pattern MVC
  - uso combinato di pagine JSP, Java Bean e Servlet
    - separazione tra presentazione, modello e logica di controllo
  - adatto per applicazioni più complesse con modalità di elaborazione e presentazione molto eterogenei

Modello JSP/1

- Interfaccia e controllo insieme
  - pagine JSP
- Modello
  - JavaBeans
- Persistenza
  - Base di dati

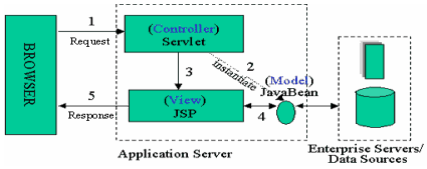


Modello JSP/1

- Flusso di controllo
  - ciascuno schermo (pagina web) riceve la richiesta
  - la analizza
  - se è di sua pertinenza produce la vista
  - altrimenti inoltra richiesta e risposta ad un altro schermo
- Non esistono cicli di controllo
  - il controllo passa sempre da uno schermo all'altro
- Utilizzo di modello e persistenza
  - servlet e pagine JSP istanziano i bean e ne chiamano i metodi
  - i bean non chiamano metodi di servlet e JSP
  - alcuni bean sono inseriti nella sessione o nello stato dell'applicazione e sono visibili da schermi diversi

Modello JSP/2

- Implementazione del pattern MVC
- Servlet per elaborazione e controllo (Control)
  - processa la richiesta originale, effettuando il grosso dei calcoli
  - memorizza i risultati in bean
- JSP per livello di presentazione (View)
  - recupera i bean creati dal servlet
  - usa i dati dei bean per costruire il contenuto della risposta



Estensioni di UML per il Web

- Stereotipi per descrivere gli schermi di applicazioni Web
  - aiutano a progettare il flusso tra gli schermi
- Rappresentano
  - pagine sul server (JSP)
  - pagine sul client (codice HTML)
  - le loro relazioni

Estensioni di UML per il Web

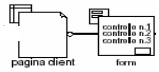
- Stereotipo di *Pagina Server*
  - rappresenta codice eseguito sul server (es: pagina JSP) per costruire dinamicamente la risposta HTML da inviare al client
  - è uno stereotipo di classe
    per le pagine JSP gli attributi e i metodi sono definiti mediante dichiarazioni

- Stereotipo di *Pagina Client*
  - rappresenta una pagina HTML visualizzata dal browser
  - è uno stereotipo di classe

Estensioni di UML per il Web

- Stereotipo di *Form*
  - rappresenta una maschera (form HTML) visualizzata dal browser all'utente
  - stereotipo di classe
    - attributi: i controlli della form

  - appartiene ad una pagina



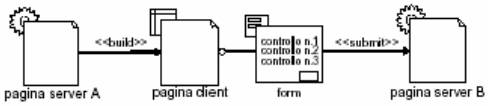
UML per il Web: Stereotipi di associazione

<<build>>

una pagina server costruisce una pagina client

<<submit>>

l'azione di una form esegue una pagina server



UML per il Web: Stereotipi di associazione

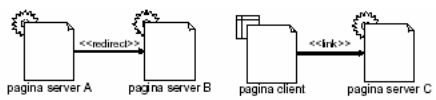
<<redirect>>

una pagina server inoltra il controllo ad un'altra

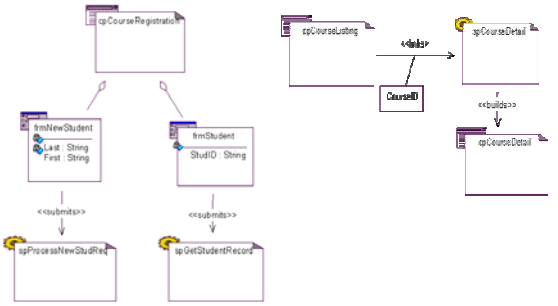
<<link>>

link da una client page a una client o server

- corrisponde al tag anchor
- è possibile rappresentare i parametri da passare con la richiesta



UML per il Web: Esempi



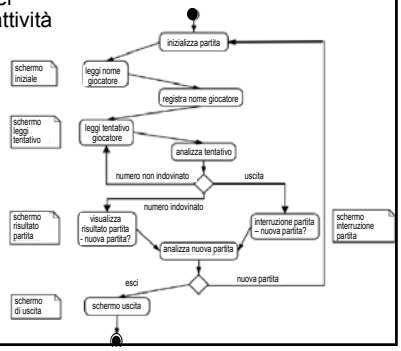
UML per il Web: diagramma delle pagine

- E' un diagramma delle classi (pagine)
  - pagine server
  - pagine client
  - form
  - associazioni
- Descrive
  - la relazione tra codice sul server e schermi prodotti sul client
  - l'effetto degli eventi scatenati dall'utente

Esempio: Indovina il Numero

Schermi indicati nel diagramma delle attività

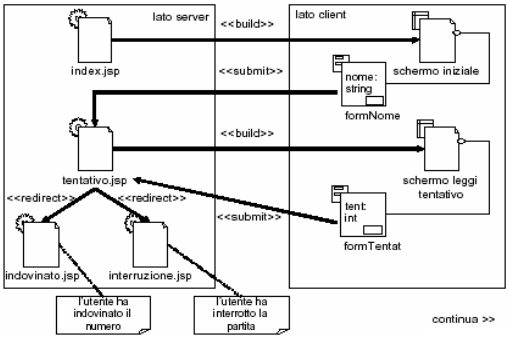
- iniziale
- leggi tentativo
- risultato partita
- interruzione
- uscita



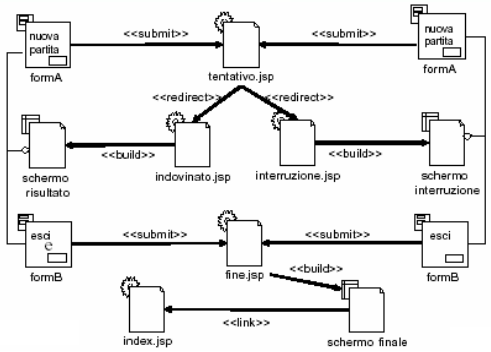
Esempio: *Indovina il Numero*

- Cinque pagine JSP
- Ciascuna costruisce uno schermo
  - index.jsp (schermo iniziale)
  - tentativo.jsp (schermo leggi tentativo)
  - indovinato.jsp (schermo risultato partita)
  - interruzione.jsp (schermo interruzione)
  - fine.jsp (schermo di uscita)

Esempio: diagramma delle pagine



Esempio: diagramma delle pagine



Metodologia di sviluppo di Applicazioni Web

- La fase di analisi è immutata
  - casi d'uso, attività, modello concettuale
- In fase di progettazione
  - si progetta l'architettura dell'applicazione
  - si progetta la base di dati (se presente)
  - si progettano le classi di modello e persistenza
  - si progetta l'interfaccia e il controllo

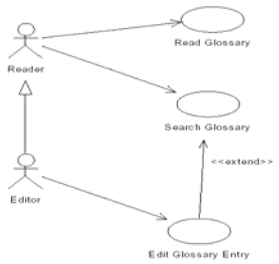
Progettare Interfaccia e Controllo

- Due attività
  - progetto del diagramma UML delle pagine
  - progetto della grafica delle pagine client
- Diagramma UML delle pagine
  - quali pagine server
  - quali pagine client (schermi)
  - quali form e con quali parametri
  - quali associazioni

Esempio: Glossario

- L'applicazione permette di accedere a un glossario
  - Il glossario contiene un insieme di termini con le relative definizioni
- Due tipi di utenti:
  - Lettore: accede in lettura al contenuto del glossario
  - Editore: può modificare le definizioni dei termini
- La home page contiene
  - un link per ogni lettera dell'alfabeto
  - una form con due campi di input: Termine e Descrizione
- Il lettore può sottomettere una richiesta di ricerca inserendo del testo in Termine o in Descrizione
  - Il sistema visualizza l'elenco di tutte le entrate del glossario che verificano i criteri di ricerca

Glossario: Casi d'uso



Nel resto dell'esempio considereremo solo la parte dell'applicazione destinata ai lettori generici
- casi d'uso *Read Glossary* e *Search Glossary*

Diagramma delle classi (solo operazioni di lettura e ricerca)

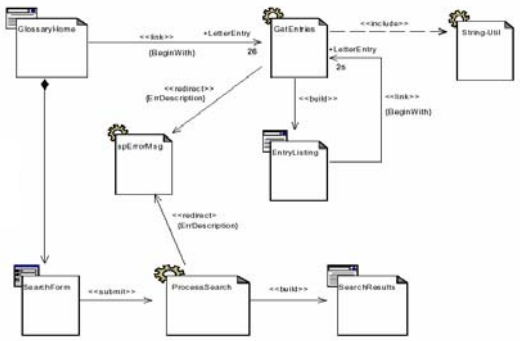


Diagramma delle classi: dettaglio