

L'architettura software generale

L'architettura si basa su una struttura multi-tier che implementa i SERVIZI (funzionalità rese disponibili agli utenti) utilizzando una modellizzazione a componenti ("*Component Software Modelling*").

Più in particolare la tecnologia di riferimento sarà JAVA - XML ed il modello di riferimento per lo sviluppo dell'applicazione è quello previsto dal Model View Controller.

Il modello MVC si basa nella suddivisione funzionale degli oggetti dell'applicazione, in modo da disaccoppiarli fra loro il più possibile: Nell'architettura prevista per la piattaforma quindi ci sono oggetti che hanno a che fare con gli aspetti legati alla sua presentazione (view), altri che riguardano le regole legate alla business logic ed ai dati (model), altri ancora che accettano ed interpretano le richieste degli utenti e sono responsabili di controllare che tali richieste siano più o meno legittime per poi esaudirle se è possibile (controller).

Il model rappresenta quindi i dati dell'applicazione e le regole che governano le operazioni con cui tali dati vengono acceduti e modificati. Spesso esso rappresenta un'approssimazione software degli oggetti realmente presenti nel dominio dell'applicazione.

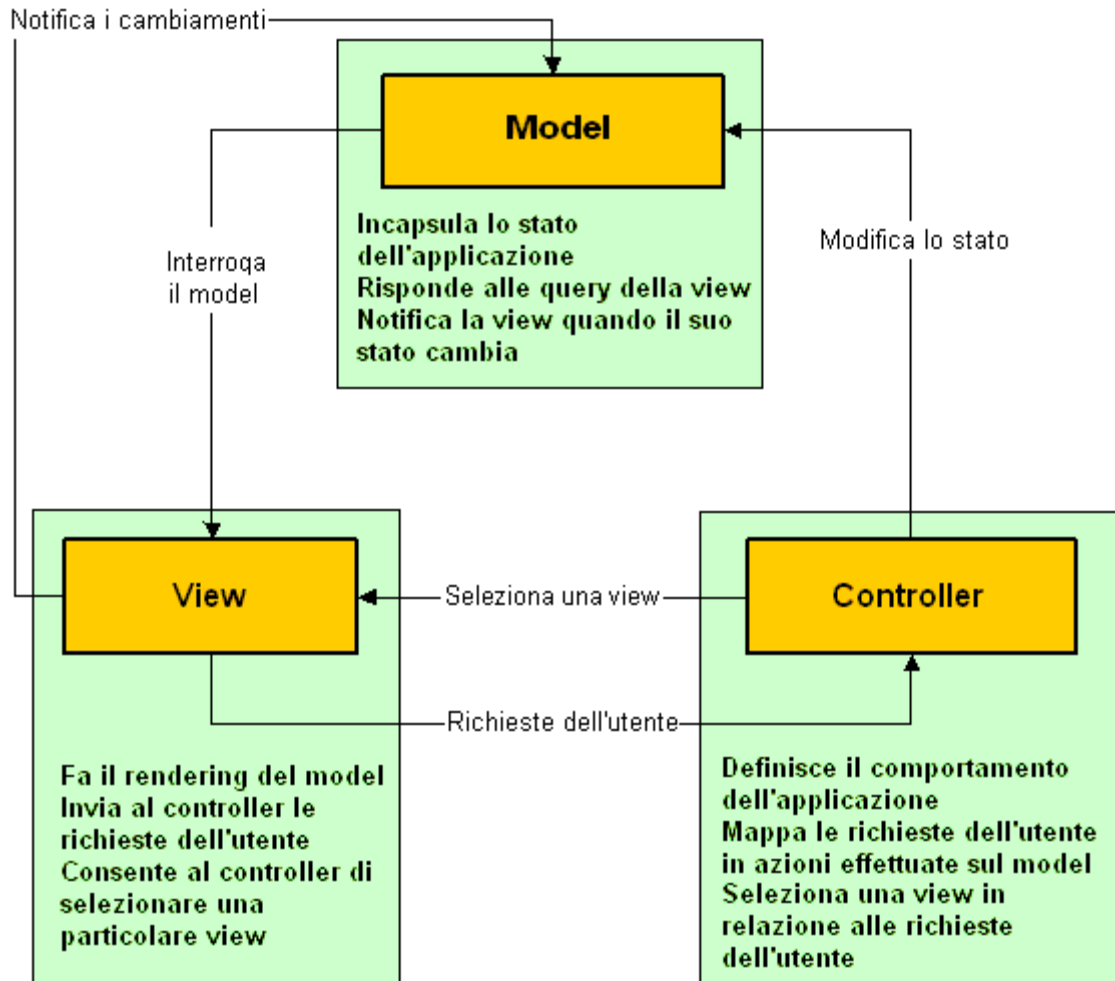
Il model notifica la view dei suoi cambiamenti e mette a disposizione della view un modo per interrogare il model circa il proprio stato.

La view ha invece il compito di effettuare il rendering del model. In altri termini, questa accede ai dati contenuti nel model e decide le modalità con cui questi dati debbano essere presentati. Quando il model cambia è responsabilità della view di mantenere la sua presentazione coerente con tali cambiamenti.

La view, infine, ha il compito di inviare le richieste dell'utente al controller.

Quest'ultimo definisce il comportamento dell'applicazione: esso interpreta le richieste dell'utente e le mappa in azioni che verranno eseguite sul model. In una web application di questo tipo, tali richieste vengono effettuate al web tier sotto forma di request get e post HTTP. Inoltre, in base alle richieste dell'utente ed agli effetti che tali richieste implicano sul model, il controller seleziona la view che verrà utilizzata per effettuare il rendering del model così modificato.

Più in particolare il modello ricalca la seguente rappresentazione:

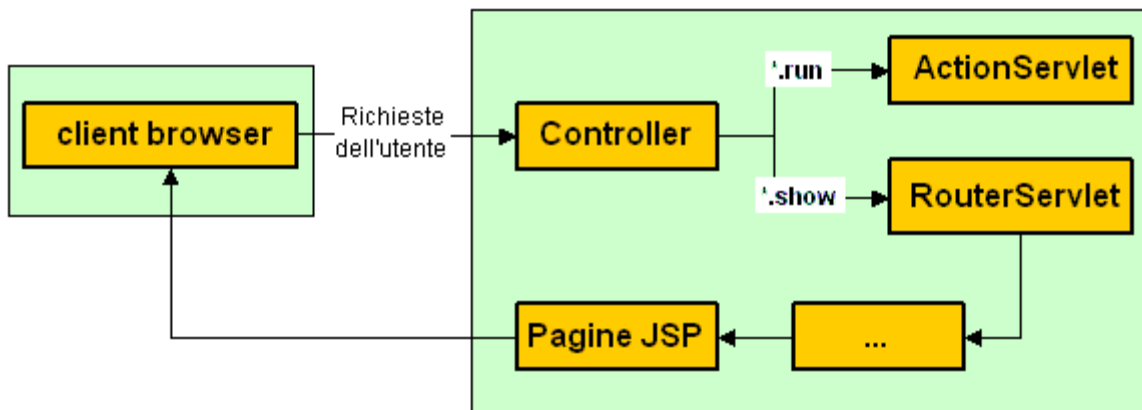


Come nella maggior parte delle applicazioni EJB-centriche, anche in questo caso, gli EJB di tipo entity, che riflettono i dati immagazzinati nella base di dati, rappresentano il model dell'architettura. Per quanto riguarda la view, i dati memorizzati nel model vengono replicati lato web da componenti JavaBeans. Questi componenti si registrano presso il controller per fare il listening degli eventi di aggiornamento del model: quando ricevono uno di tali eventi interrogano gli entity bean per allineare il proprio stato. Il rendering dei dati presenti nei componenti JavaBeans viene poi ovviamente effettuato mediante pagine JSP. Infine, per ciò che concerne il controller, esso è probabilmente la parte più complessa dell'applicazione. Mentre infatti la view è concentrata sul web-tier e il model risiede unicamente sull'ejb-tier, il controller necessita di estendersi su entrambi i livelli; proprio a causa di tale complessità può essere funzionalmente suddiviso nelle seguenti sottoparti: request processor: converte le request HTTP in eventi comprensibili dal resto dell'applicazione, consentendo di concentrare in un unico punto le processazioni specifiche del protocollo HTTP e quindi rendendo il resto dell'applicazione indipendente dal tipo di client utilizzato; web controller: inoltra gli eventi generati dal request processor all'EJB controller, assicurando che il risultato dell'aggiornamento del model, effettuato da quest'ultimo, venga propagato ai componenti JavaBeans ovvero alla view;.ejb controller: accetta gli eventi inviati dal web controller e modifica il model coerentemente con tali eventi; è anche responsabile di mantenere lo stato della sessione dell'utente all'interno dell'applicazione.

Le tre parti di MVC sono così organizzate:

- View: Una serie di pagine JSP e di fogli di trasformazione XSLT
- Controller: due servlet che si occupano dello smistamento del traffico

- Model: dei componenti JavaBeans o EJB dove è inserita tutta la business logic della applicazione.



I 3 livelli previsti sono dunque:

Presentation Layer (Front End)

L'interfaccia di fruizione degli applicativi verrà realizzata attraverso l'uso di pagine dinamiche costruite con tecnologia J2EE-JSP (Java Server Pages) e/o attraverso la renderizzazione HTML svolta dall'utilizzo della tecnologia XSLT.

La pagina JSP sarà costituita da codice HTML e da codice JavaScript.

Il client è costituito dal solo Browser standard che supporti HTML ver 4 o superiore.

Il livello di *Presentation* è rappresentato dai server Web che gestiscono la parte d'interfaccia verso l'utente generico Internet.

In quest'area si collocano tutte le componenti che hanno il compito di riconoscere l'utente e la modalità di accesso di volta in volta adottata (quindi un browser piuttosto che un terminale Wap o quant'altro), validano e profilano l'utente mediante il riconoscimento e la verifica di parole chiave e/o certificati digitali e creano, in base a queste informazioni, una presentazione del portale d'accesso personalizzata.

La verifica delle credenziali dell'utente potrà avvenire in diverse modalità e per diverse finalità; in particolare, vi sarà un primo riconoscimento per il solo scopo di fornire all'utente viste personalizzate del portale di accesso e per la gestione interna della profilazione (utente registrato), in più potrà essere richiesta un'identificazione tramite parole chiave per l'accesso ai servizi (utente riconosciuto).

Application layer (Business Logic)

L'implementazione della logica di business e l'astrazione dal database è costituito classi Java (servlet, JavaBeans o EJB, Web Services e utilizzo del protocollo SOAP) per l'accesso ai dati e la logica dell'applicazione.

All'interno degli Application Server saranno gestiti i meccanismi di autenticazione.

Una volta che l'utente sia stato "abilitato" alla navigazione e alla fruizione delle informazioni e dei servizi disponibili, le componenti presenti in quest'area avranno il compito d'instradare le richieste effettuate verso i fornitori del servizio e di garantire la continuità e la fruibilità dei servizi, mediante tecniche di ripartizione automatica dei carichi in commistione con le componenti di Business Logic.

Il livello di Business Logic rappresenta il vero e proprio cuore del sistema e racchiude la logica applicativa. A essa sono demandate tutte le funzionalità di accesso ai dati e quelle d'integrazione, trasformazione e

cooperazione verso sistemi omogenei e/o eterogenei, così come la gestione della consistenza della base concettuale.

Il compito principale di questa componente è quello di ricevere richieste provenienti dalla componente di Presentation, dopo opportune verifiche; le richieste arrivano corredate delle informazioni di autenticazione e autorizzazione dell'utente, che possono essere nuovamente utilizzate per nuovi controlli e/o filtri sulle informazioni da rendere disponibili.

Sulla base delle richieste pervenute avverrà l'accesso diretto ai dati o ai servizi necessari alla composizione della risposta per l'utente finale. L'accesso ai dati potrà avvenire in modalità transazionale tra sorgenti omogenee, per garantire quanto più possibile la consistenza e la coerenza dei dati, dovunque essi siano memorizzati.

Database Layer

Il livello Data Layer permette al Business Logic di reperire le informazioni sulle quali effettuare le elaborazioni; i componenti del Data Layer contengono la logica che permette sia l'interfacciamento verso sistemi di memorizzazione dati (data base) che verso file system gerarchici.

L'applicazione sarà costituito da un database nativo XML che verrà acceduto dal motore di Information Retrieval Extraway™. L'interfacciamento alle API di Extraway è garantito attraverso un framework Java che consente il dialogo in fase di query e di editing con la base dati XML nativa.

Infrastruttura di rete

L'architettura del servizio dovrà essere progettata in modo da garantire le seguenti condizioni:

- Alto grado di affidabilità
- Fruibilità (interfacce semplici utilizzabili anche da utenti non esperti)
- Elevati livelli di performance (tempi di risposta adeguati)
- Scalabilità lineare senza modifiche alla struttura software

La natura di questo tipo di applicazioni richiede l'utilizzo di una tecnologia che raggiunga l'eccellenza in termini di scalabilità, alta disponibilità ed integrazione degli ambienti preesistenti, per garantire non solo le performance, ma anche la massima affidabilità nella protezione dei dati (*reliability*), la continuità del servizio (*availability*) e la rapida individuazione e correzione degli errori (*serviceability*).

Tali considerazioni trovano la propria applicazione in un'architettura tecnologica che soddisfi i requisiti di espansibilità orizzontale e verticale.

Espansibilità verticale, ovvero un sistema che sia in grado di essere messo in produzione su un unico livello hardware e sia di espandersi sui *n* livelli definiti.

L'espansibilità orizzontale deve essere garantita dalla clusterizzazione dei servizi e/o dalla gestione degli stessi su macchine dedicate. Quindi l'applicazione dovrà poter essere gestita, in relazione alle esigenze di servizio, da un'unica macchina, da una macchina per ciascun livello infrastrutturale, da una coppia di macchine per livello (clustering per garantire l'alta affidabilità del servizio), da più coppie di macchine in cluster per ciascun servizio di ogni livello.

Il modello finale al quale giungere è quindi quello illustrato nella figura, e la progettazione iniziale del sistema sarà tale da non preconstituire vincoli al requisito posto di espandibilità verticale ed orizzontale.

Dal punto di vista sistemistico, esistono solo due livelli, quello di Front End e quello di Back End. Sul primo si installano i server che ricevono le connessioni direttamente dagli utenti generici sulla rete Intranet/Internet, quindi i server Web e i Multimedia Server.

Sulla rete di Back End sono invece posizionati i server che ricevono le connessioni solo dai server di Front End e da utenti particolari, connessi con la redazione attraverso una rete privata (Vpn).

Su questo livello saranno quindi posizionati gli application server i DB Server e il File server.

Questa soluzione ha come scopo principale quello di ottimizzazione del traffico di rete, dividendolo tra quello proveniente da Internet e quello applicativo destinato ai server di Back End. Garantisce inoltre una maggior sicurezza, in quanto esiste una vera e propria distinzione tra la reti visibili da Internet e non.

I tre livelli applicativi, invece, indicano i tre componenti che costituiscono la piattaforma e non il numero di macchine che la costituiscono (i tre livelli possono essere distribuiti su un numero indefinito di server). Il modello applicativo a tre livelli divide l'applicazione nei tre componenti logici descritti in dettaglio nel seguito (Presentation, Business Logic e Dati). Tali componenti comunicano tra loro utilizzando un'interfaccia di astrazione che nasconde le funzionalità del componente.

Nell'architettura a tre livelli, quel che si costruisce non è un'applicazione nella comune accezione del termine, ma è una collezione di moduli client e server che comunicano tramite un'interfaccia standardizzata: tali moduli, una volta combinati, si comportano come un sistema di applicazioni integrate. Ogni modulo è un oggetto condivisibile e riutilizzabile all'interno di altri sistemi applicativi. I vari moduli possono essere visti come oggetti, perciò presentano tutti i vantaggi della tecnologia a oggetti.

Un beneficio importante di questa architettura è che l'implementazione d'interfacce differenti del sistema, utilizzando le stesse regole di business, diventa molto semplice.

La separazione delle funzionalità applicative da quelle legate ai dati permette inoltre l'attivazione, in modo semplice, di tecniche di bilanciamento del carico (*load balancing*).

Architettura software di dettaglio

Per la realizzazione dell'infrastruttura relativa sia alla piattaforma di backoffice che a quella di consultazione il framework applicativo si basa sulle seguenti componenti software:

- Sistema operativo: piattaforma. La soluzione proposta prevede la piena compatibilità con diversi SO come Linux, Sun Solaris, Microsoft Windows 2003 Server, etc
- HTTP Server distribuito in configurazione di "Load Balance" su 2 Front-End WEB: si ritiene opportuno utilizzare Microsoft Information Server 6.0 quale Server Web di riferimento.
- FTP Server Attivato sui server di Database, attraverso cui gestire l'upload degli allegati digitali sullo Storage server
- Installazione del filtro Jakarta IISAPI che permette il tunnelling http tra i front-end Web e l'Application server.
- Installazione della Java 2 Enterprise Edition ver . 1.4.2_06
- Application Server Java: JBoss3.2.1-Tomcat 4.1.24 quale componente per la gestione della logica di business di tutta la piattaforma.
- Extraway Information Retrieval quale motore di ricerca di tutta la piattaforma, in grado di gestire le banche dati in XML nativo. Il motore è assolutamente "multipiattaforma" in quanto può essere installato su Solaris/SPARC, Linux, AIX, SCO Open Server, SCO Unixware, Windows 2K