Caratteristiche, problematiche e tecnologia dei sistemi informativi basati sul Web

Features, problems and technology of Web-based Information Systems

Nicola Aloia, Cesare Concordia, Francesco Furfari, Vittorio Miori

CNUCE – Istituto del Consiglio Nazionale delle Ricerche
Via S. Maria 36, 56126 Pisa
Telefono 050-31562997 - Fax 050-3138091/2(G3/G4)
E-mail: Nicola.Aloia@cnuce.cnr.it

Sommario

La diffusione crescente del World Wide Web e la capillare penetrazione della tecnologia ad esso connessa sta generando notevoli cambiamenti nella gestione ed organizzazione delle informazioni di moltissime attività, con un notevole impatto sul sistema informativo di molte aziende. D'altra parte la crescente affidabilità di strumenti e tecniche generati dalla "rivoluzione Web" rende sempre più conveniente basare lo sviluppo di nuove applicazioni su questa tecnologia, soprattutto per i grossi vantaggi forniti in termini di scalabilità e distribuzione. Nel presente articolo vengono discusse le caratteristiche fondamentali di sistemi informativi basati sulla tecnologia Web e illustrate le problematiche connesse al loro sviluppo ed alla loro gestione. Vengono altresì presentati alcuni strumenti tecnologici utili per la loro realizzazione.

Abstract

The ever-increasing growth and penetration of the World Wide Web and associated technologies is bringing about significant changes in how information is organized and managed, with far-reaching consequences for the information systems of a great number of organizations. Moreover, the increasing reliability of tools and techniques stemming from the "Web revolution" is rendering it ever more convenient to base the development of new applications on this technology, especially because of the great advantages it affords in terms of scalability and distribution. The present article deals with the fundamental characteristics of information systems based on Web technology and illustrates the problems that need to be tackled in their development and management. It also show some useful technological tools for WIS implementation.

Keywords:

Web Information System, Internet, Open Hypermedia System, Public Access WIS.

1. Introduzione

Nella metà degli anni 80, il progresso tecnologico col conseguente abbattimento dei costi di componenti e apparecchiature elettroniche determinò l'introduzione e la diffusione di sistemi di calcolo in moltissime realtà organizzative promuovendo l'automazione di processi produttivi e di trattamento dell'informazione. La necessità del sistema informativo nelle organizzazioni è andato sempre più affermandosi, in sincronia col crescente fabbisogno informativo. Mano a mano che l'informazione stessa

diveniva una merce, la cui produzione, gestione e distribuzione era necessario automatizzare, il ruolo del sistema informativo estendeva i suoi compiti al di là delle sue funzioni fondamentali di supporto del sistema di regolazione e controllo dell'organizzazione.

Il rapido e crescente sviluppo del World Wide Web come strumento di accesso ad Internet, determinato principalmente dalla comparsa di clienti multi-protocollo, come Mosaic, Lynx e Netscape, ha generato una nuova rivoluzione informatica offrendo a tanti nuovi soggetti la possibilità di utilizzare tecnologie, a costi contenuti, per allargare le loro attività e distribuire informazione in un ambito molto più vasto. All'inizio il WWW è stato utilizzato, da aziende ed organizzazioni, principalmente come ambiente di diffusione delle informazioni. Nel corso degli anni i "siti Web" si sono raffinati ed arricchiti di servizi trasformandosi in strumenti sempre più complessi, che spesso affiancano il sistema informativo, là dove presente, o ne costituiscono il nucleo iniziale, sebbene a volte in maniera confusa. Nuove esigenze di applicazioni realizzate per il Web, ad esempio la possibilità di accedere basi di dati aziendali, e nuove possibilità derivanti da realtà emergenti, ad esempio l'e-Business, affiancati dalla maturazione di strumenti utilizzabili nel Web rende sempre più necessario una ripensamento dei sistemi informativi per le organizzazioni. Nel presente lavoro vengono discusse le caratteristiche fondamentali di sistemi informativi in cui le componenti fruibili tramite Internet sono strettamente integrate col sistema dell'organizzazione. Vengono altresì considerati i vantaggi derivanti dall'utilizzo del Web come infrastruttura per la creazione dei sottosistemi componenti il sistema informatico e analizzate le nuove problematiche che determinano. Il resto dell'articolo è così organizzato: nel primo paragrafo è presentata una breve e concisa rassegna di articoli in cui abbiamo trovato spunti interessanti. Il successivo paragrafo presenta le caratteristiche fondamentali dei sistemi informativi e delle applicazioni basate sul Web; segue, nel paragrafo 4, una discussione sulla realizzazione di Web Information Systems (WIS) ed un'analisi delle problematiche derivanti dalla loro implementazione. Il paragrafo 5 presenta la tecnologia, che a nostro parere, si presenta al momento più adatta per la realizzazione di WIS; seguono le conclusioni ed i riferimenti bibliografici..

2. Attività correlate

Una articolata trattazione delle caratteristiche e delle problematiche generali relative ai WIS si può trovare nel nº.7 di Communications of the ACM uscito nel Luglio 98 (Vol. 41), tra gli articoli segnaliamo un'analisi delle problematiche specifiche dei Public Access WIS [Kam 98]. Sul numero di agosto 98 della stessa rivista c'è un interessante articolo che fa il punto sullo stato della tecnologia Web evidenziandone le carenze ed i problemi irrisolti [Fie 98], un'analisi complessiva sulle tecnologie Open Hypermedia System applicate al Web può essere trovata in [Len 97]. Molto importanti per lo sviluppo dei WISs sono gli studi condotti da Alberto Mendelzon sulla formalizzazione dei linguaggi di interrogazione per il Web; al suo gruppo di lavoro si deve la specifica di WebSQL [Men 97], un linguaggio dichiarativo ad alto livello per la ricerca di informazioni sul Web, e la realizzazione di WebOQL [Aro 98], un sistema per l'estrazione di dati da documenti semistrutturati (http://www.cs.toronto.edu/~gus/weboql/index.html). Una analisi abbastanza approfondita dell'evoluzione dei sistemi informativi aziendali nell'epoca del commercio elettronico è contenuta in [Kal 99]. Nel libro vengono inoltre proposte metodologie per la progettazione e realizzazione di sistemi informativi aziendali finalizzati all'e-Business.

3. Sistemi informativi e Web

Un sistema informativo è un insieme di procedure organizzate uomo/macchina atte a provvedere l'informazione necessaria al funzionamento di una organizzazione. Il sistema informativo non si identifica

con il sistema informatico che è la componente del sistema informativo per l'elaborazione automatica dell'informazione. In un sistema informativo si possono riconoscere alcuni sottosistemi fondamentali, che sono compresi nel sistema informatico proprio [Sch 77]:

- Il sottosistema di gestione delle comunicazioni.
- Il sottosistema di gestione delle interazioni uomo/macchina
- Il sottosistema di gestione dei dati.

Nei sistemi informativi tradizionali gli strumenti informatici utilizzati hanno il compito di facilitare la comunicazione, la fruizione e l'elaborazione delle informazioni e sono vincolati dalla struttura organizzativa e dalle architetture hardware e software adottate.

Negli ultimi anni la tecnologia Web è cresciuta a ritmi impressionanti trasformando i vari "*siti*" da semplici finestre nell'area marketing a piattaforme capaci di supportare tutti gli aspetti di un'attività organizzata. L'opportunità di utilizzare la piattaforma Web, per la realizzazione di sistemi informativi, offre notevoli benefici, sebbene ponga nuove problematiche per la loro implementazione. Un sistema informativo basato sul Web (WIS) è un sistema la cui implementazione e gestione avviene tramite l'utilizzo della tecnologia Web [Isa 98]. Esiste una chiara differenza tra un WIS ed un insieme di pagine Web. Nei primi le varie componenti (es. gestore di basi di dati, processi delle transazioni, ecc.) sono strettamente integrate e consentono di svolgere attività preposte al buon funzionamento di una organizzazione. Rispetto ad un sistema informativo tradizionale, un WIS è un sistema aperto in cui le informazioni assumono forme sempre più complesse, grazie alle caratteristiche intrinseche della tecnologia Web, si può pensare che nei WIS sia presente una tendenza a contrastare il decadimento (entropia) e quindi che abbiano una maggiore capacità di adeguarsi con efficienza ai mutamenti.

La realizzazione di un WIS deve contemplare gli stessi principi di disciplina richiesti da un sistema informativo tradizionale. Nel prossimo paragrafo analizziamo vantaggi e svantaggi derivanti dall'utilizzo della tecnologia Web per la realizzazione di sistemi informativi.

4.Web-based Information Systems

Una caratteristica fondamentale dei WISs è la potenzialità di raggiungere una comunità enormemente più ampia di quella delle tradizionali realizzazioni cliente/servente basati su rete proprietarie, consentendo l'accesso ad informazioni e funzioni anche ad utenti occasionali. Tale possibilità è enfatizzata dalle caratteristiche intrinseche della tecnologia Web: standard aperti e software molto spesso gratuito e facilmente disponibile. Questa caratteristica risulta estremamente interessante per organizzazioni che abbiano una struttura distribuita su un territorio molto ampio o che intendono allargare la propria attività al cosiddetto e-Business [Kal 99]. Un'altra area rilevante per la realizzazione di WISs è costituita dalle istituzioni pubbliche. Queste raccolgono, generano e distribuiscono, in generale, una grande quantità di informazioni, la cui conoscenza può risultare molto importante per i cittadini e per le imprese. Questi sistemi, chiamati Public Access WIS (PAWIS) [Kam 98] hanno una caratteristica da cui non si può prescindere: l'*accesso universale,* cioè la possibilità di utilizzare i servizi o di accedere ai documenti pubblici deve essere fornita a tutti, a prescindere dall'esperienza e dalle dotazioni hardware dell'utilizzatore.

La progettazione e realizzazione di un WIS richiede nuovi approcci, rispetto a quelli utilizzati nei sistemi informativi tradizionali, e sebbene in tante situazioni la tecnologia fornisce un valido supporto con indubbi vantaggi, d'altro canto emergono nuove problematiche. Consideriamo i tre principali sottosistemi informatici (gestione della comunicazione, gestione dei dati, interfaccia uomo-macchina), analizzandone i vantaggi derivanti dall'utilizzo della tecnologia Web.

Gestione della comunicazione. La rete Internet ed il protocollo TCP/IP, ormai standard de-facto, non pongono limiti geografici o hardware/software all'espansibilità del sottosistema di gestione delle comunicazioni. Espandere il sottosistema di gestione delle comunicazioni di un WIS, i cui nodi si trovano ad esempio all'interno di un edificio, aggiungendo un nodo che si trova in un'altra città (o in un altro continente) non richiede costi elevati o procedimenti particolari, la tecnologia è la stessa usata nell'installazione locale. La possibilità di scalare facilmente il sistema da uso locale a uso remoto è uno dei principali pregi dei WISs rispetto ai sistemi tradizionali.

Interazione uomo-macchina. Le più importanti caratteristiche del sottosistema di gestione delle interazioni uomo-macchina di un WIS sono costituite dall'universalità dei browser grafici e dalla possibilità di utilizzare codice (mobile o script-embedded) multipiattaforma per la realizzazione della componente cliente del sistema. Questo non solo semplifica notevolmente lo sviluppo delle interfacce utente, che non sono più vincolate ad architetture hardware o software, ma facilita anche la distribuzione e l'installazione delle stesse [Alo 98]. Un altro elemento, altrettanto importante, è quello legato all'utilizzo nei browser della cosiddetta *mimecap Interface*. Grazie ad essa i browser sono in grado di riconoscere la natura dei dati ed attivare l'opportuno programma o plug-in, qualora non siano in grado di effettuarne il rendering. Questo porta ad una scalabilità quasi illimitata del sottosistema di interfaccia, con l'opportuno plug-in/programma si possono visualizzare dati di qualsiasi natura, riducendo al minimo l'intervento dell'utente o dello sviluppatore.

Gestione dei dati . Per quanto riguarda il sottosistema di gestione dei dati di un WIS, la situazione è molto più complessa. In un sistema informativo tradizionale, di solito, esso è realizzato tramite l'uso di un DBMS, che implementa lo schema della base di dati. In tale situazione, ogni proprietà dello schema ha una corrispondenza diretta con una proprietà del database implementato (cioè l'attributo di una relazione o una database procedure). In un WIS, la corrispondenza univoca tra schema e proprietà del database non è più fissata a priori a causa della presenza di URL [Ber 98]. Le URL possono essere viste come particolari tipi di dato [Men 97], che oltre ad avere proprietà ed operazioni proprie, ereditano proprietà ed operazioni dei tipi di dato da essi linkati e del tipo di server/protocollo cui fanno riferimento. Le URL possono essere interpretate come puntatori a informazioni memorizzate in gestori dei dati che possono essere DBMS, file system, programmi (CGI o servlet) che generano documenti *on the fly* etc. In questa situazione possiamo considerare il sottosistema di gestione dei dati, non più costruito su un insieme fissato a priori di tipi di dati, ma estensibile dinamicamente con nuovi tipi. Possiamo quindi considerare il sottosistema di gestione dei dati di un WIS composto da un multi-gestore di dati, che contempla, tra gli altri, di sicuro anche i DBMS. In questa nuova visione, l'implementazione dello schema è distribuito non solo dal punto di vista della locazione nella rete, ma anche per quanto riguarda i gestori di dati coinvolti. L'introduzione di URL tra i tipi di dati arricchisce enormemente le possibilità offerte dal sottosistema di gestione dei dati, sebbene comporti nuove problematiche che tratteremo più approfonditamente nel sottoparagrafo successivo.

4.1 Problematiche dei WIS

Affinché sia possibile realizzare un WIS efficiente ed efficace è necessario affrontare alcune problematiche non presenti nei sistemi informativi tradizionali; ci limiteremo qui a descrivere solo le problematiche relative alla componente informatica. La progettazione di un WIS deve contemplare non solo la distribuzione delle informazioni ma anche l'integrazione con altri strumenti per svolgere compiti complessi, come supporto al lavoro cooperativo o al sistema di regolazione e controllo di un'organizzazione. La percezione stessa, da parte dell'utente, del WIS come un sistema integrato di servizi e funzionalità è un elemento fondamentale per la sua accettazione e per il suo successo. Di seguito elenchiamo alcune problematiche, molte delle quali sono presenti nelle tradizionali applicazioni

basate sul Web (i cosiddetti Siti), a cui la tecnologia WIS dovrebbe essere in grado di fornire risposte adeguate.

Meccanismi di ricerca. Dove cercare? Quali termini di ricerca usare? Come capire se il risultato è il migliore possibile? La realizzazione di un buon *motore di ricerca* per un WIS deve contemplare la possibilità di definire facilmente ed efficacemente filtri ben realizzati tramite interfacce utenti chiare che "guidino" nella compilazione di query prive di ambiguità. Il *motore* deve fornire suggerimenti alternativi in caso di ricerche con esito negativo e contemplare la possibilità di affinamenti successivi dei filtri di ricerca.

Lentezza nel trasferimento dati. L'adozione di opportune tecniche di bilanciamento del carico tra cliente e servente tramite l'utilizzo di codice mobile ed opportuni protocolli di comunicazione possono migliorare le prestazioni del sistema.

Supporto per il lavoro cooperativo. Un sistema informativo basato sul Web deve supportare sia la collaborazione asincrona (condivisione dei dati tra più soggetti, ciascuna impegnato in una sessione di lavoro individuale), che sincrona (condivisione simultanea dei dati tra più soggetti impegnati in una sessione di lavoro collettiva). Gli strumenti utilizzati vanno dalla semplice posta elettronica a sofisticati sistemi di teleconferenza. L'integrazione tra Web e strumenti di lavoro collaborativo è ad oggi ancora nella fase iniziale, la scelta tra le diverse soluzioni dipende dal particolare tipo d'interazione che si vuole ottenere e da altri fattori quali l'esperienza degli utenti e/o la dotazione hardware. I maggior problemi per la realizzazione di una soluzione efficiente, derivano dalle peculiarità del protocollo HTTP, che utilizza un modello di comunicazione strettamente client/server. Questo modello è adatto per l'accesso alle risorse ma non è sicuramente adatto ad attività di lavoro cooperativo. Ad esempio nel modello client/server il client deve eseguire accessi per verificare l'avvenuta modifica delle risorse. Il polling può risultare inefficiente nei casi in cui le risorse da controllare siano tante ed i cambiamenti poco frequenti. La realizzazione di un ambiente di lavoro cooperativo basato sull'aggiornamento di documenti HTML risulta quindi una soluzione poco funzionale. Fino ad ora il problema è stato aggirato utilizzando plug-in sul client o add-on sul server, si avverte comunque la mancanza di un meccanismo nativo per la notifica.

Meccanismi di caching locale. I meccanismi di accumulazione locale implementati dagli attuali browser sono mirati essenzialmente a limitare, se possibile, il trasferimento di pagine da visualizzare. Questi meccanismi sono abbastanza poveri e limitano notevolmente il loro utilizzo off-line, cioè senza la necessità di avere una connessione attiva. Un WIS deve prevedere dei sofisticati meccanismi di accumulazione locale delle informazioni mirati all'utilizzo delle applicazioni il più possibile off-line, soprattutto per quelle situazioni in cui la connessione alla rete avviene su linea commutata.

Gestione dei link. La gestione ed il controllo dei link è un aspetto molto importante nella realizzazione dei grandi sistemi hypermediali [Hal 88]. Negli Open Hypermedia System i link rappresentano le relazioni, esistenti nel modello, tra i diversi documenti che compongono il sistema, rendendo possibile la "navigazione". Nei WISs, come abbiamo visto nel paragrafo precedente, i link sono una componente del sottosistema di gestione dei dati. La tecnologia Web presenta attualmente dei limiti che impediscono una efficiente gestione dei link. Il limite più grosso è sicuramente legato alla loro natura *embedded* nel formalismo HTML, ossia il fatto che siano inseriti nel testo del documento. Questa caratteristica comporta i seguenti problemi:

- è molto difficile realizzare degli strumenti per la gestione, il controllo e/o l'aggiornamento dei link
- senza l'interpretazione del contesto non è semplice risalire dai link alle relazioni esistenti tra le diverse parti del sistema.

Questo rende molto difficile sia la realizzazione di ricerche basate sulle relazioni tra i diversi documenti sia ricavare "*a posteriori*" le proprietà degli oggetti puntati. Questo secondo punto è molto importante, infatti una corretta implementazione del sottosistema di gestione dei dati può avvenire solo conoscendo tutte le proprietà dei dati coinvolti. Sono state suggerite alcune soluzioni per superare questo limite come ad

esempio l'uso dei metalevel link [Tak 98], oppure dei first-class link [And 97], sviluppati per gli Open Hypermedia System, ma sino ad ora nessuna di esse è stata implementata con successo. Probabilmente si dovrà attendere il pieno uso di un formalismo più strutturato, come XML, per poter ovviare a questi inconvenienti.

Sicurezza ed autenticazione. La sicurezza e la gestione sicura dei dati sono tra i principali problemi da risolvere. In un sistema informativo tradizionale ci sono più tipologie di utenti ciascuna con specifici "diritti", inoltre vengono eseguite transazioni che spesso coinvolgono dati critici, è necessario quindi implementare accuratamente i meccanismi di protezione e di gestione della sicurezza. In un WIS l'implementazione dei meccanismi di sicurezza può risultare complessa. Se infatti i firewall o altri meccanismi di protezione software possono risultare efficaci nell'impedire tentativi di intrusione nei computer o nelle reti, essi non forniscono alcun aiuto per la sicurezza delle transazioni ed in generale per la sicurezza delle funzioni di un sistema informativo. Affinchè un WIS possa considerarsi sicuro è necessario che siano implementate le seguenti condizioni:

- **Confidenzialità** la comunicazione o il flusso delle informazioni tra più parti deve essere ristretto alle parti stesse
- **Authentication** ogni soggetto deve essere sicuro dell'identità del suo interlocutore o dell'origine dell'informazione cui sta accedendo
- **Integrità dei dati:** i dati non devono poter essere modificati durante il trasferimento tra i soggetti di una comunicazione
- ***Accesso ai servizi selettivo:*** è auspicabile che un soggetto "veda" solo i servizi a cui può accedere

È possibile oggi implementare in tutto o in parte alcune di queste condizioni grazie ai protocolli sicuri ed al criptaggio dei dati, tuttavia la tecnologia Web non fornisce alcun supporto nativo e molto è lasciato alle singole implementazioni.

Accessibilità. La possibilità che le informazioni ed i servizi di un WIS siano acceduti dal maggior numero di utenti possibile è un requisito molto importante non solo per i WISs pubblici. È necessario affrontare il problema secondo due aspetti: l'aspetto tecnico (ad esempio fornire delle alternative alla navigazione grafica) e l'aspetto "cognitivo" [Nie 90] (ad esempio evitare che, seguendo una catena di link, l'utente si "perda" raggiungendo punti lontani dal suo obiettivo). Per risolvere le problematiche legate all'ultimo aspetto occorre rendere evidente in ogni momento il contesto in cui l'utente sta operando attraverso una accurata progettazione e realizzazione sia della struttura dei documenti che del sistema di interazione. Nell'analizzare il primo aspetto operiamo una distinzione: in un WIS possiamo trovare interfacce per la presentazione delle informazioni ed interfacce per l'accesso ai servizi. Per quanto riguarda la fruizione delle informazioni esistono delle linee guida [W3c 99] proposte dal Web Accessibility Initiative (WAI) del Web Consortium ed ormai universalmente accettate. In molti casi la loro implementazione consente di ottenere un grande livello di accessibilità alle informazioni presenti in un WIS. Il discorso è diverso per l'accesso ai servizi. Spesso questo avviene tramite interfacce realizzate con codice mobile, garantire la piena accessibilità ai servizi può comportare allora notevoli problemi. Alcuni passi avanti sono stati effettuati di recente sia attraverso l'integrazione dei client Web con strumenti di *access technology* (la tecnologia che consente alle persone disabili l'utilizzo del computer), sia attraverso l'implementazione di API ad hoc nei principali linguaggi utilizzati per la realizzazione di codice mobile [Alo 99].

5. Tecnologie per i WIS

L'interesse sempre crescente dei ricercatori e le grosse spinte che arrivano dall'area business lasciano ben sperare in una risposta esauriente alle problematiche esposte nel paragrafo precedente. In questo

paragrafo più che fornire una risposta ai quesiti posti, ci limitiamo a presentare alcuni strumenti tecnologici che possono proficuamente aiutare nello sviluppo di WIS.

5.1 Java e i WIS

La tecnologia Java è oggi ampiamente utilizzata nella realizzazione di WIS: si pensi agli applet o all'impiego di servlet e delle API JDBC [Alo 98] [Alo 99] per realizzare la *business logic* della componente *server-side* del sistema. Esiste tuttavia un'altra caratteristica di Java ancora poco utilizzata che lo rende adatto alla realizzazione di WIS, cioè la possibilità di accedere ad oggetti remoti (distributed object). È possibile cioè realizzare dei programmi in cui gli oggetti comunicano, tramite la rete, indipendentemente dalla loro localizzazione. L'utilizzo di tecnologie di gestione distribuita di oggetti, come CORBA (tramite il protocollo IIOP) e RMI (tramite i protocolli JRMP/IIOP), permette di realizzare delle applicazioni raffinate che consentono di superare in parte i limiti legati alla unidirezionalità del protocollo HTTP. Ad esempio è possibile implementare facilmente dei meccanismi di *call-back* tra server e client che evitino il polling continuo per l'accesso alle risorse. Questo impatta notevolmente sui tre sottosistemi che compongono un WIS:

Gestione della comunicazione: attraverso un uso combinato dei protocolli HTTP e IIOP o JRMP è possibile arricchire le caratteristiche di espandibilità e modularità dei WIS con la bidirezionalità e l'accesso a metodi distribuiti.
Interazione uomo-maccina: utilizzando la tecnologia RMI o CORBA, gli applet Java diventano parte di un'applicazione distribuita, con i conseguenti vantaggi.
Gestione dei dati: utilizzando la comunicazione tra oggetti è possibile rendere più efficace e puntuale l'interazione con il gestore dei dati. Se il gestore dei dati, inoltre, utilizza la tecnologia ad oggetti, allora si ottiene il vantaggio del trattamento uniforme delle classi Java e delle classi del DBMS.

Come esempio di utilizzo di questa tecnologia, in Figura 1 è descritta l'architettura generale da noi adottata nella realizzazione di un WIS, denominato INFEA, per il Ministero dell'Ambiente. La tecnologia per la programmazione distribuita adottata è RMI, la scelta è stata dettata dal fatto che tutti gli oggetti del sistema sono realizzati in Java. Il sistema realizzato ha una architettura client/server in cui la *presentation logic* è implementata utilizzando il linguaggio Java, il formalismo HTML ed il linguaggio Javascript, la *business logic* e il *data access* sono implementati tramite applicazioni scritte in Java, i dati sono memorizzati in un DBMS relazionale. Nel sistema INFEA sono presenti due tipologie di client:

- Applet Java
- Documenti HTML

Gli applet sono utilizzati nei casi in cui abbiamo ritenuto necessario che il client contenesse, oltre alla logica di presentazione, anche una parte della business logic. È il caso delle interfacce di inserimento/modifica/cancellazione dei dati ad esempio, nelle quali oltre alla verifica della correttezza delle informazioni avvengono, in tempo reale, interazioni con il server per facilitare i compiti dell'utilizzatore. Quando invece la funzione preponderante del client è quella di presentazione si è adottato il formalismo HTML, anche in questi client sono implementate delle funzionalità minime di controllo per mezzo del linguaggio Javascript. Si è cercato inoltre di garantire per quanto possibile l'accessibilità, replicando, quando necessario, le funzionalità degli oggetti distribuiti attraverso l'utilizzo di servlet e del protocollo CGI. Il server presenta due tipologie di programmi: *servlet* ed *oggetti remoti*. I servlet vengono invocati tramite il Web Server ed eseguono esclusivamente funzioni di interrogazioni del DBMS. Gli oggetti remoti vengono riferiti, tramite il protocollo RMI, direttamente dai client. Negli oggetti remoti sono implementate le funzionalità di gestione e modifica dei dati e di autenticazione dei client. I servlet e gli oggetti remoti

utilizzano librerie comuni per l'interazione con il DBMS. Prima di essere trasferiti tra client e server (e viceversa), i dati vengono strutturati secondo il formalismo XML, allo scopo di avere una maggiore modularità ed un maggiore controllo sulla correttezza delle operazioni (vedi pragrafo successivo).

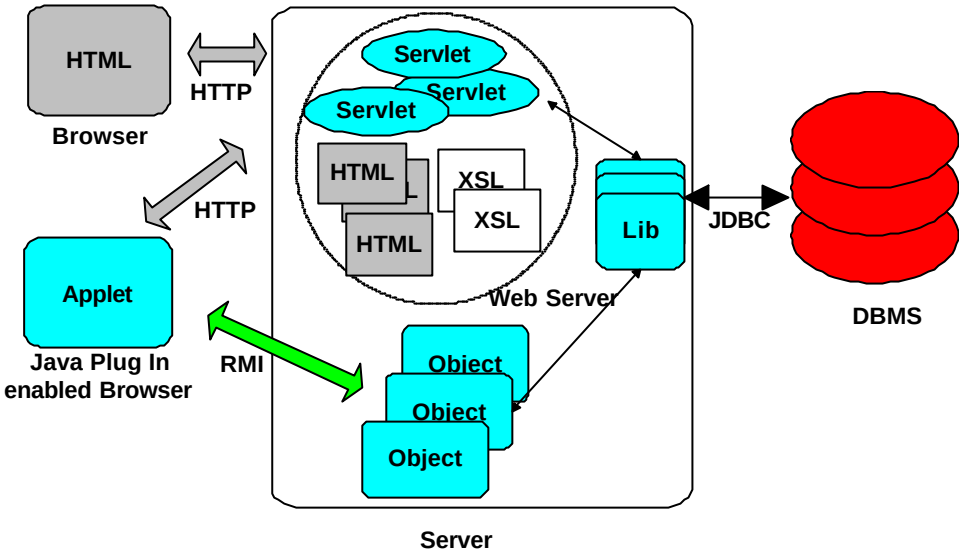

Figura 1 Architettura del sistema INFEA

5.2 Data Centric vs Document Centric

La scelta del tipo di gestore dei dati di un WIS è influenzata da molti fattori, uno dei più importanti è sicuramente la natura delle informazioni che deve contenere, ossia se si tratta di *dati strutturati* o *documenti* poco (o per niente) strutturati. Nel primo caso è sicuramente consigliabile l'uso di un DBMS nel secondo caso, è preferibile usare un *content management system*, più adatto alla gestione di documenti non (o poco) strutturati. Le informazioni del sistema informativo Infea rientrano sicuramente nella prima categoria, da qui la scelta di un DBMS. Poiché le informazioni in un WIS sono visualizzate principalmente usando browser Web, il problema da affrontare consiste nel trasferimento dei dati da un universo *data-centric* (il DBMS), adatto alla loro gestione-elaborazione, ad un universo *document-centric*, il Web, adatto per la loro visualizzazione. Questo trasferimento avviene estraendo e 'collezionando' i dati in modo da esplicitare le relazioni codificate nello schema della base di dati. Esistono molti modi per fare questo, il nostro tentativo è stato quello di utilizzare un formato intermedio che consenta di unire le caratteristiche dei due universi (*data-centric* e *document-centric*) allo scopo di potenziare e semplificare la gestione delle comunicazioni client/server. In questo modo si ottiene la possibilità di implementare meccanismi di caching, di verificare errori di trasmissione, di gestire *on the fly* cambiamenti delle interfacce per il rendering. Inoltre c'è un importante aspetto legato alla gestione di sistemi WIS (anche dal punto di vista della manutenzione del software), che consiste nella separazione tra la gestione del contenuto dei documenti e la loro modalità di presentazione. Un formalismo che consente di implementare tali specifiche è l'eXtensible Markup Language (XML). XML è un formalismo sviluppato dal W3C [W3c 98] principalmente per la strutturazione di documenti Web, ma che comincia ad essere utilizzato anche come formalismo di "*data description*".

5.3 XML e DBMS

Il primo passo nella implementazione di un tale approccio consiste nel definire un mapping opportuno tra la struttura della base dei dati e la struttura dei documenti XML. Non esiste una unica soluzione a questo problema e ci sono diversi tipi di approccio [Roz 00] [Buc 99], che però possono essere ricondotti a due categorie: *template-driven* e *model-driven*.

Template-driven mapping

Non viene creato un mapping predefinito tra struttura del documento XML e struttura del db. Il meccanismo si basa su comandi inseriti (embedded) in un template del documento; questi comandi vengono interpretati ed eseguiti da un processo middleware. Ad esempio consideriamo il seguente template in cui uno statement SQL select è incluso nell'elemento <SqlStmt>:

```
<?xml version="1.0"?>
<Riunioni>
<Titolo> Prossime riunioni progetto INFEA:</Titolo>
< SqlStmt >
SELECT Tema, DataOraInizio FROM Riunioni WHERE Tema LIKE '%INFEA%'
</SqlStmt >
<Commento>La partecipazione è riservata ai componenti della redazione</ Commento >
</Riunioni>
```

Una volta processato il template potrà diventare:

```
<?xml version="1.0"?>
<Riunioni>
<Titolo >Prossime riunioni progetto INFEA:</Titolo>
<Record>
INFEA: Definizione questionari EA 10 Maggio 2000-10:00AM
</Record>
< Record>
INFEA: Incontro con responsabili laboratori 1 Giugno 2000-08:00AM
</Record>
< Commento >La partecipazione è riservata ai componenti della redazione</Commento >
</Riunioni>
```

Questo approccio risulta estremamente flessibile per il trasferimento da database relazionali a doumenti XML ma poco adatto al contrario.

Model-Driven Mappings

In questa categoria rientrano quei casi che tendono ad imporre al documento XML un modello più o meno complesso dipendente dalla struttura del database (o viceversa). Due sono i modelli più comunemente adottati dai packages in circolazione: nella prima tipologia, il documento XML viene modellato come un insieme di relazioni (tables) ad esempio:

```
<database>
```

```
  <relazione1>
    <record1>
      <campo1>...</campo1>
      <campo2>...</ campo2>
      ...
    </record1>
    ...
  </relazione1>
  ...
</database>
```

Nel nostro esempio il termine *relazione* può ingenerare confusione, infatti esso può essere il *result set* di una query, che può comprendere campi di più relazioni del db. La seconda tipologia di modelli struttura il documento XML in un albero, i cui nodi sono oggetti del database, le cui proprietà vengono codificate come attributi XML. Questo modello pur fornendo un mapping diretto con database ad oggetti (o gerarchici) può essere adottato con opportune trasformazioni anche per database relazionali. Ad esempio nel caso del database di Infea, un Ente potrebbe essere così modellato (parzialmente):

Altri modelli di strutturazione possono essere adottati, ad esempio un documento XML può essere usato per rappresentare un grafo orientato, comunque questi modelli non sembrano avere largo utilizzo.

5.3.1 Problematiche generali

Di seguito elenchiamo alcune problematiche riscontrate nell'implementazione del sistema INFEA, che tuttavia hanno valenza di generalità. L'approccio da noi adottato è del tipo *model driven*, in cui il documento viene modellato come un insieme di relazioni che rispecchiano la struttura delle interfacce di visualizzazione.

Tipi di dato

Nel formalismo XML non esistono i tipi, fatta eccezione per "unparsed entities", per cui tutti i dati di un documento sono espressi in forma testuale. Il compito di trasformare i dati dal formato testo al formato specifico definito nel db (e viceversa) è lasciato al middleware.

Binary data

Due possono essere i modi per memorizzare dati binari: *unparsed entities* e *Base64 encoding* (una codifica MIME che consente la trasformazione di dati binari in un sottoinsieme del set US-ASCII).

Null data

Nel formalismo XML il concetto di *null data* può essere espresso attraverso l'opzionalità degli elementi o degli attributi, se il modello adottato prevede che un campo venga trasformato in un elemento e questo campo ha valore null, l'elemento non viene inserito nel documento. Per evitare il verificarsi di errore di inserzione nel db o di validazione del documento XML bisognerà controllare che elementi (o attributi) opzionali di un documento XML siano mappati in campi *nullable* (e viceversa).

Set di caratteri

Un documento XML può contenere tutti i caratteri *Unicode,* eccezion fatta per alcuni caratteri di controllo. Bisogna perciò fare molta attenzione, nella realizzazione di codice middleware, alla trasformazione di dati che non adottano la codifica in ASCII puro.

***Memorizzazione di simboli* Markup**

È possibile ovviamente memorizzare elementi in un database come stringa, il problema sorge in fase di estrazione dei dati, in cui non è possibile, ad esempio, determinare a priori se un *markup* "<" è l'inizio di un elemento o una parte del testo. Tra le soluzioni che possono essere adottate, consigliamo l'uso di flag particolari (bisogna però stare attenti alle altre applicazioni che usano il db).

5.4 XML e Java: SAX & DOM

Le caratteristiche intrinseche della tecnologia Java (multipiattaforma, portabilità del codice etc.), ne hanno fatta la candidata ideale per le implementazione di librerie specifiche per l formalismo XML. Esistono diverse librerie disponibili gratuitamente, che possono essere ricondotte a due tipologie di architettura: l'implementazione di API per la realizzazione di parser e processor *event-driven* e l'implementazione delle specifiche Document Object Model (DOM).

> **API event driven:** Java è uno dei linguaggi di implementazione delle Simple API for XML (SAX). Si tratta di librerie sviluppate dai membri della lista XML-DEV@ic.ac.uk e disponibili anche in Perl, Python e C++. Queste librerie sono alla base di quasi tutti i parser o processor XML attualmente in commercio e vengono distribuite gratuitamente. Si tratta di parser *event-driven* che leggono il documento XML sequenzialmente ed intercettano particolari eventi (inizio/fine documento, inizio/fine elemento etc) a cui è possibile associare una azione semantica.
> **DOM:** l'implementazione di queste API è stata realizzata, tra gli altri, dal W3C [W3c 98a] (in Java, Perl, C++, Phyton, ECMA Script). Tramite esse è possibile trasformare un documento XML in un albero il cui parsing consiste nella visita dei nodi.

La scelta tra i due approcci è legata alle particolari esigenze di programmazione: in genere si tende a preferire l'approccio *event-driven* quando la struttura del documento non è molto annidata e il mapping adottato non è dipendente da contesto, in caso contrario DOM può risultare una scelta migliore. Nelle attuali implementazioni delle librerie disponibili gratuitamente, c'è un fattore molto importante che può influenzare la scelta: DOM richiede molta memoria. Nulla vieta comunque di adottare una soluzione "mista". Ed è quella che abbiamo adottato per INFEA: sul server (dove la risorsa memoria non è critica) la

struttura del database è rappresentata attraverso il suo modello ad oggetti e contiene le regole di mapping, mentre sul client i dati in arrivo vengono analizzati ed elaborati tramite un parser event-driven.

5.6 XML e rendering: XSLT

Un requisito essenziale nella realizzazione di WIS, consiste nella separazione tra la gestione del contenuto dei documenti Web e la loro presentazione; questo consente di poter generare presentazioni diverse, degli stessi dati, a seconda delle necessità. Per implementare questa specifica, in INFEA, abbiamo utilizzato il formalismo: eXtensible Stylesheet Language Tranformations (XSLT) [W3c 99b]. Sono stati perciò creati dei file di stylesheet (anche se il termine stylesheet è improprio, esso viene utilizzato comunemente per indicare questo tipo di file), che vengono associati *on the fly* ai documenti XML. Se la richiesta delle informazioni arriva da un browser Web, un processore XSL genera, secondo il procedimento descritto, dei file XHTML e li invia al browser. Se la richiesta arriva da un applet, i dati vengono restituiti direttamente in formato XML, e visualizzate con il rendering opportuno (Fig. 2).

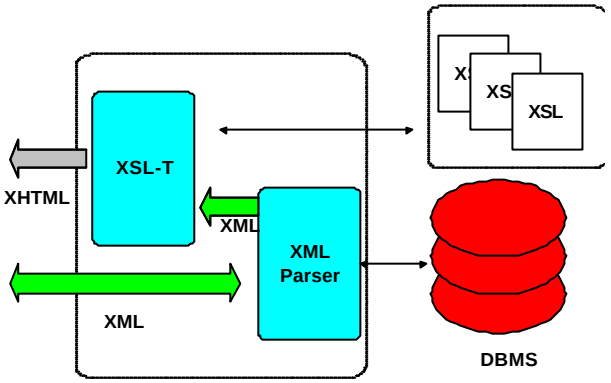

Figura 2 INFEA: data transfer

6. Conclusioni.

Nel presente lavoro sono state analizzate le caratteristiche di un sistema informativo basato sul Web. Sono stati discussi i vantaggi dell'uso della nuova tecnologia, rispetto all'implementazione di un sistema informativo tradizionale e considerate le nuove problematiche che essa comporta. Sono stati altresì presentati gli strumenti tecnologici più rilevanti per la realizzazione di WIS, facendo di tanto in tanto dei riferimenti ad un sistema informativo che stiamo sviluppando per conto del Ministero dell'Ambiente.

Bibliografia

[Alo 98] Aloia N., Concordia C. "*Accesso a Basi di Dati via Web*" Proc. of SEBD '98 "Sistemi evoluti per Basi di Dati", Ancona giugno 1998, (pp. 3-18)

[Alo 99] Aloia N., Concordia C., Miori V. **"**Web Architectures for Database Access" Proc. of WebNet '99, Honolulu, Hawaii, ottobre 1999, (pp. 1473-1475)

[Alo 99a] Aloia N., Concordia C., Furfari F. Miori V *"Considerazioni per la realizzazione di sistemi informativi basati su Web accessibili ai disabili* " Atti del 6th Convegno Nazionale Informatica, Didattica e Disabilità – Andria (BA), novembre 1999

[And 97] Anderson K., *"Integrating open hypermedia systems with World Wide Web"* in proceedings of the 8th ACM Conference on Hypertext (Southampton, England, Apr.6-11). ACM Press, New York,1997, pp.157-166.

[Aro 98] Arocena G., Mendelzon A. "*WebOQL: Restructuring documents, databases, and Web"* in Proceedings of the 14th International Conference on Data Engineering (Orlando, Florida) 1998.

[Ber 98] Berners-Lee T., Fielding, R. and L. Masinter, *"Uniform Resource Identifiers (URI): Generic Syntax and Semantics"*, RFC 2396, August 1998.

[Buc 99] Buck Lee "*Modeling Relational Data in XML*", white paper, 199 http://www.extensibility.com/xml_resources/modeling.htm

[Fie 98] Fielding E., Whitehead J., Anderson K., Bolcer G., Oreizy P., Taylor R. *"Web Based Development of Complex Information Products"* Communication of the ACM 41 (7) pp.84-91, luglio 1998.

[Hal 88] Halasz F. *"Reflections on NoteCards: Seven Issues for the Next Generation of Hypermedia Systems* ", Communication of the ACM, 31 (7) pp 836-852, July 1988.

[Isa 98] Isakowitz T., Bieber M. and F. Vitali: *"Web Information Systems",* Communication of the ACM, Vol. 41, No. 7, July 1998.

[Kal 99] Kalakota R, Robinson M. "*e-Business Roadmap for Success* "(Addison-Wesley 1999)

[Kam 98] Kambil A., and Ginsburg M. *"Public Access Web Information Systems: Lessons from the Internet EDGAR project",* Communication of the ACM 41 (7) pp.91-98, 1998

[Len 97] Lennon J. A. *"Hypermedia systems and applications: World Wide Web and beyond"* (Berlin, Springer, 1997)

[Men 97] Mendelzon A., Milo T., *"Formal Models of Web Queries"* in Prooceedings of the ACM PODS Conference, Tucson, Arizona, Giugno 1997, per informazioni aggiornate: *http://www.cs.toronto.edu/~websql/*.

[Nie 90] Nielsen Jakob *"The art of navigating through Hypertext"* Communication of the ACM 33(3) 1990, pp 296-310

[Sch 77] Schoderbek P., "*Management Systems*", Wiley 1977

[Roz 00] Rozenshtein D., Bondur T. Abramovich A. "*Tree & Graph Processing in SQL*", Preview edition, Sql Forum press, 2000.

[Tak 98] Takahashi Kenji, *"Metalevel Links: More Power to Your Links"* Communications of the ACM 41 (7) pp103-105, luglio 1998

[W3c 98] W3C Specifications www.w3.org/xml, Febbraio 1998

[W3c 98a] W3C Recommendation www.w3.org/DOM/ Ottobre 1998

[W3c 99] W3C Recommendation "*Web Content Accessibility Guidelines 1.0*" http://www.w3.org/TR/WAI-WEBCONTENT/, Maggio 1999.

[W3c 99a] W3C Recommendation www.w3.org/TR/xslt Novembre 99