Perché un post-relazionale
RDBMS e database post-relazionali, il perché di una scelta

ROBERTO TIRABASSI

Rappresentare il concetto di informazione e renderlo accessibile all'utente, nella maniera più semplice ed allo stesso tempo completa, costituisce una crescente esigenza da parte di coloro che si trovano ad essere fornitori e gestori di conoscenza. Il contributo in questione illustra una interessante soluzione, che sceglie di risolvere il problema della rappresentazione utilizzando il database post-relazionale, inteso come strumento alternativo al database relazionale, che con esso completa il panorama degli strumenti informatici di ricerca, gestione e conservazione dell'informazione. Infatti, se da un lato il database relazionale rappresenta il primo ed il più diffuso strumento attraverso il quale gestire e conservare i dati, è proprio nella necessaria operazione di normalizzazione che trova un limite, poiché il processo di frazionamento e di successiva riorganizzazione delle informazioni pone problemi di fragilità e complessità di ricostruzione dell'informazione completa originale. L'informazione, al contrario, necessita per sua natura di una rappresentazione di carattere strutturato, in grado di esprimere la varietà e la complessità dei legami che la accompagnano, e questa esigenza si accorda con le caratteristiche sia del database post-relazionale che dei database object-oriented*, strumenti di ricerca creati parallelamente ai post-relazionali ed orientati verso la programmazione ad oggetti. Inoltre, se aggiungiamo, a quanto è stato finora esposto, l'ulteriore necessità di rendere i dati fruibili e sicuri nel tempo, con uno sforzo minimo ed un costo contenuto, appare naturale come alla scelta del post-relazionale si affianchi quella di utilizzare standard aperti ed in particolare XML come linguaggio di rappresentazione dei dati. Sono queste le riflessioni che hanno portato la 3D Informatica, azienda con una ventennale esperienza nel campo dell'*information technology*, alla realizzazione del Progetto Extraway, ovvero alla creazione di un* native XML database*, nato per soddisfare le esigenze dei gestori dell'informazione. Extraway si presenta come un sistema di gestione delle informazioni che, grazie all'approccio post-relazionale, riesce ad assicurare la piena integrità nel tempo delle unità di informazione - denominate record - non come semplice serie di dati, bensì mantenendo la loro identità di oggetti auto-consistenti e strutturati, dotati di una serie di caratteristiche che ne delineano la reale sostanza. Essi sono infatti automaticamente corredati da metadati relativi ad elementi come utente, data, orario, contesto ecc. sigillati con un codice di integrità. Inoltre, Extraway si caratterizza non solo per la scelta di rappresentare le unità di informazione in formato XML, ma, diversamente dal comportamento di altri database, effettua la conservazione delle stesse su* file system *direttamente in questo formato in modo molto flessibile ed efficiente. Attraverso l'utilizzo di questo approccio, il*

record di informazione si caratterizza come l'unione indivisibile di dati, metadati ed eventuali allegati informatici, la cui evoluzione nel tempo viene tracciata capillarmente così da poter ricostruire l'intero percorso dell'unità di informazione, dalla creazione fino alla versione corrente.

In questo scenario Extraway si propone come lo strumento grazie al quale il concetto di informazione acquisisce orizzonti temporali ben superiori a quelli del software *che la gestisce.*

Parole chiave: XML - Database - Dato - Record - Informazione - Metadato - Documento - Integrità - Sicurezza - Autenticità - Architettura software

Ogni qual volta si parli, genericamente, di database, la nostra mente va ai database relazionali in quanto essi, oltre ad affondare le radici nelle origini del concetto stesso di database, sono indubbiamente i primi ed i più diffusi strumenti *software* in grado di dare risposta all'esigenza di conservare e rendere fruibili dei dati, mansione per la quale hanno rappresentato a lungo l'unica soluzione.

Parliamo infatti espressamente di "dati" quando pensiamo alle più comuni realizzazioni effettuate con database relazionali. L'insieme di questi dati assume un particolare significato e diviene di una certa utilità solo se visto nell'ambito di una precisa struttura d'archivio rappresentata in modo bidimensionale tramite tabelle e colonne, le quali devono essere combinate tra loro tramite legami complessi.

Il singolo dato, infatti, altro non è che una parte di un concetto più ampio, quello di "informazione", che assume una forma ricca, strutturata, articolata e che deve invece essere "normalizzata" per rientrare nei canoni di tabelle e colonne di un RDBMS [Relational Database Management System].

Perché questi "dati" assumano appunto il ruolo di "informazioni", e quindi risultino di effettiva utilità, è necessaria la realizzazione di applicazioni che guardino alla struttura del database come ad una realtà pluridimensionale ottenuta da una razionale combinazione di singoli scenari bidimensionali.

Compito di queste applicazioni è dare all'utenza una visione d'insieme di questa realtà attraverso la quale il dato destrutturato viene reso significativo dandone una rappresentazione sostanzialmente strutturata.

Quello che la mente concepisce come "informazione" deve essere frazionato e destrutturato per poter essere rappresentato correttamente da un RDBMS, per poi essere nuovamente organizzato in una forma complessa che lo renda nuovamente significativo e fruibile per gli utenti.

Quest'attività, che in alcune applicazioni risulta certamente sensata e funzionale, può incontrare limiti non semplicemente prestazionali, ma anche legati alla

complessità delle informazioni da rappresentare nell'ambito del database. Il compito della "normalizzazione" può risultare particolarmente complesso ed il risultato sin troppo distante dalla forma naturale che l'informazione avrebbe nel mondo reale.

Laddove si intenda privilegiare il ruolo di "informazione" che un insieme di dati può esprimere viene quindi naturale concepire i dati stessi in forma strutturata. Una simile esigenza trova una naturale soluzione nell'adozione di database post-relazionali ed ancor più nell'uso del linguaggio XML.

Parlando di *database post-relazionale* si vuole esprimere un concetto che ha assunto diversi significati nel tempo.

Se le prime esperienze post-relazionali hanno condotto alla realizzazione di sistemi che avevano origine dagli RDBMS e si discostavano da essi solo in piccola parte, la naturale evoluzione del panorama informatico, specie attraverso la diffusione di Internet, ha condotto una schiera di fornitori e fruitori di informazioni a richiedere funzionalità più ampie ed una rappresentazione sempre più ricca delle informazioni reperite.

Mentre i database relazionali hanno continuato a soddisfare le esigenze di chi doveva gestire "dati", i gestori e fornitori di "informazioni", e soprattutto di "conoscenza", hanno trovato in sistemi post-relazionali le risposte alle loro specifiche esigenze. La particolare attenzione alle funzionalità di *full text retrieval*, evoluta nel tempo nella direzione della multimedialità delle basi di dati, del Web semantico e delle ontologie, fanno dei database post-relazionali la realtà in grado di affiancarsi agli RDBMS per completare a 360° il panorama dei database.

Diamo a Cesare quel che è di Cesare: un database relazionale è semplicemente uno strumento "diverso" rispetto ad un database post-relazionale. I due svolgono compiti differenti, se pure simili, e si prestano a soddisfare differenti esigenze. Sarebbe irragionevole pensare che database relazionali e post-relazionali possano sostituirsi gli uni agli altri ed adattarsi a diversi obiettivi funzionali. Essi sono concepiti innanzitutto per rappresentare i dati ovvero le informazioni, in modo differente e, sulla base di questa diversa architettura, offrire diverse funzionalità.

Come detto, la diffusione di Internet e l'esigenza di distribuire informazioni e servizi tramite la rete ha portato alla crescita di un'ampia gamma di strumenti di sviluppo orientati alle applicazioni Web. Tali strumenti, nati nell'ambito della programmazione ad oggetti, hanno naturalmente un orientamento alla gestione dei dati che ne ricalca l'origine. Il dato è visto come oggetto strutturato non solo perché questa visione semplifica lo sviluppo delle applicazioni, ma anche e soprattutto perché in tal modo risulta maggiormente rappresentata la forma e l'articolazione che le "informazioni", o ancor meglio la "conoscenza", assumono nel mondo reale. Il processo di evoluzione che ha portato alla realizzazione di questi strumenti ha parallela-

mente condotto alla generazione di database orientati agli oggetti, detti anche ODBMS [Object Oriented Database Management Systems] in contrapposizione agli RDBMS.

Compito dei database post-relazionali, ed in particolare degli *object oriented database management system*, è quindi quello di unire il meglio di questi scenari, favorendo tanto la semplicità nello sviluppo di applicazioni per la fruizione delle informazioni quanto la gestione delle stesse attraverso un formato intuitivo ed in grado di rispecchiare la loro reale natura.

Altre esigenze si miscelano a quelle già esposte.

In primo luogo sarebbe auspicabile che le informazioni fossero fruibili nel tempo limitando al minimo gli sforzi, e quindi i costi, perché esse risultino intelligibili.

Una delle vie che intuitivamente va seguita per garantire il conseguimento di simili obiettivi consiste nella scelta di standard aperti per la rappresentazione dei dati.

Se ad essa uniamo quanto precedentemente detto, ovvero scegliamo di porre l'accento sul concetto di "informazione" e sulla sua descrizione come singola entità che rappresenta un'unità di informazione consistente e completa di ogni sua parte, la scelta del linguaggio XML per rappresentare queste unità si presenta come la più naturale.

Questa unità di informazione, che può trovare facilmente rappresentazione anche sotto forma di oggetto di un ODBMS, verrà di seguito identificata con il termine *record*. Un *record* rappresenta un'unione indivisibile di dati strutturati, arricchiti da metadati descrittivi ed eventuali "allegati informatici".

La scelta di rappresentare tali *record* tramite il linguaggio XML si rivela particolarmente adatta anche a garantire la "sicurezza" delle informazioni.

Il progetto eXtraWay, un *native XML DBMS*

Le scelte indicate sino ad ora sono quelle alla base del progetto eXtraWay di 3D Informatica. Ultimo di una lunga serie di esperienze nel mondo dell'*information technology*, il progetto eXtraWay sposa una ventennale attività nel settore dell'*information retrieval* con la filosofia XML, conducendo alla realizzazione di un *native XML database*.

Extraway è un insieme di moduli che consente di gestire e conservare nel tempo in modo sicuro *record* di informazione, cioè unità di informazione consistenti, complete, in modo da garantirne l'integrità, l'autenticità, la confidenzialità e l'intelligibilità nel tempo.

Per meglio comprendere cosa si intenda per *record* o *unità di informazione* e l'importanza ricoperta in particolare dalla sua integrità ed autenticità, approfondiamo il concetto di *documento* nell'accezione che tale termine assume nell'ambito del progetto eXtraWay.

Un documento è il risultato di un atto di volontà, la volontà di un singolo o di un'organizzazione di dare testimonianza di un evento o di un'intenzione, di comunicare a terzi una notizia o richiedere ad essi delle informazioni. Esso assume il suo reale significato quando è auto-consistente e completo in ogni sua parte e quando può rispondere ai più classici dei quesiti: chi, cosa, dove, come, quando e soprattutto perché. La sua completezza deve quindi essere garantita nel tempo così come la sua disponibilità ed intelligibilità.

Il compito di conservare questi oggetti auto-consistenti è pienamente svolto da sistemi post-relazionali come Extraway, che ha fatto di un simile approccio alle informazioni una delle sue principali caratteristiche.

In un sistema relazionale il documento, così come l'abbiamo inteso, sarebbe frazionato in molteplici tabelle. Solo tramite le loro relazioni reciproche sarebbe possibile ricostruire il documento nella sua forma originaria, o quanto meno in una forma ad essa prossima. Sempre in un simile scenario, interventi di modifica sui contenuti delle singole tabelle, avulsi dal concetto di documento, porterebbero lo stesso ad avere una forma ed un contenuto differenti da quelli originari, senza la possibilità di ricostruire la sua reale "sostanza". Ciò mostra chiaramente come l'approccio relazionale al *documento* non possa soddisfare pienamente il requisito dell'integrità.

Altra caratteristica fondamentale di Extraway consiste nell'aver scelto la rappresentazione dei record in forma XML. Ciò non vuol dire semplicemente che il documento viene "mostrato" in un *output* XML mentre si trova riposto in un qualche contenitore eventualmente in un diverso formato, comportamento che appartiene a molti altri database, relazionali e non, bensì che esso si trova fisicamente in formato XML nativo sul *file system*. Questa scelta offre la maggior garanzia di intelligibilità e disponibilità dei documenti nel tempo.

Il motore realizza la sua missione esponendo comandi di salvataggio ed accesso ai record, ovvero ai documenti, di tipo atomico, cioè non frazionabile, automaticamente corredati da metadati sia in codice che in chiaro, relativi all'utente, la data e l'orario, l'organizzazione corrente del produttore, del piano di classificazione, del sistema *software* e sigillati con un codice di integrità. Il record è quindi l'unione indivisibile e dimostrabile di dati, metadati ed eventuali allegati informatici.

Ogni cambio di stato subito da un record viene salvato da un sistema di *tracking* persistente, che permette la ricostruzione dell'immagine corrispondente del record a seguito di ogni azione, dalla creazione fino alla versione corrente.

La sicurezza nei formati standard aperti

Gli obiettivi della consistenza, completezza, integrità ed autenticità dei record sono da considerarsi un tutt'uno con quello della loro sicurezza.

La sicurezza informatica risulta dalla somma di più fattori, in primo luogo:

Confidenzialità:	solo soggetti autorizzati possono accedere ai record
Autenticità:	il record è quello che dichiara di essere
Integrità:	il record è consistente e completo
Disponibilità:	un record è accessibile e intellegibile a chi ne ha i diritti.

Confidenzialità

Se guardiamo con occhio critico alla sicurezza offerta da un qualsiasi database avremo di fronte a noi due possibili scenari. Uno di essi è rappresentato da un insieme di politiche di accesso e di protezione delle informazioni gestite applicativamente, l'altro consiste nella cifratura dei dati perché siano disponibili esclusivamente al loro "proprietario".

La sicurezza di queste informazioni mostra almeno due punti deboli. Nel primo scenario siamo comunque nelle mani dell'amministratore del database il quale ha accesso a tutti i dati e, come lui, tutte le persone che sono preposte ad amministrare queste informazioni. Per ragioni pratiche le *password* di accesso amministrative sono spesso a conoscenza di diverse persone ed il dato, in quanto tale, è sicuro nella misura in cui è "fidato" l'amministratore. Nel secondo scenario, per contro, l'informazione, che è effettivamente altamente protetta, corre il rischio di essere "persa" laddove la sua accessibilità è legata a fattori estremamente dipendenti dal singolo. Sarà sufficiente una *password* persa o dimenticata ed il documento non sarà più fruibile.

La tendenza ad ottenere la sicurezza nascondendo le informazioni, *security through obscurity*, lascia quindi il posto ad un diverso approccio. Compito dell'amministratore è garantire un accesso esclusivo alle informazioni da parte del motore Extraway e, non essendo necessario porre le informazioni in CLOB o BLOB di un database relazionale per metterle al sicuro da occhi indiscreti, l'uso di formati aperti è pienamente accettabile portando in dote altri vantaggi che risulteranno più evidenti in seguito.

Questo approccio ha comunque la necessità di proteggere l'accesso ad alcuni dati anche dallo sguardo degli amministratori. In questo caso la cifratura tramite algoritmi standard basati su chiavi che verranno consegnate a persona diversa dall'amministratore di sistema (ad esempio il responsabile dell'archivio, che tipicamente non

accede in sala server o comunque non ha le credenziali di amministratore di sistema) fa sì che solo l'intervento congiunto delle due figure permetta di decifrare il contenuto di tali record al di fuori delle politiche di sicurezza.

Autenticità

Si può presumere che un record sia autentico quando viene garantita anche a distanza di tempo tutta la catena di operazioni che ha subito da parte dei relativi utenti, documentandone la politica di accesso, i riferimenti temporali e quando è possibile individuando e circoscrivendo gli eventuali momenti di discontinuità nella sicurezza del sistema. A tale scopo Extraway registra in chiaro, in due allocazioni specifiche, sia i riferimenti alle operazioni di routine, sia gli eventi straordinari che possono abbassare temporaneamente il livello di sicurezza. Ciò in quanto l'autenticità dell'archivio, e quindi dei record, risulta tanto maggiore quanto tali eventi sono rilevabili indipendentemente dal programma, e quindi in chiaro ed in posizioni note, e siano conosciuti e circoscritti.

La capillare registrazione dei cambi di stato di ogni record, unitamente alla possibilità di identificare tutti i momenti di discontinuità, consentono appunto di dare ragionevoli garanzie di autenticità ad ognuno di essi, condizione che non può essere altrimenti garantita da sistemi privi di una tracciatura dettagliata.

Integrità

Un formato aperto, dotato di un sigillo noto, dà evidenza dell'integrità del record. Extraway completa ogni record con:

l'impronta degli eventuali allegati informatici secondo un formato noto (SHA1 o RIPMED160)

l'impronta *dell'intero* record (eccetto i suddetti allegati) secondo un formato noto.

Extraway verifica l'impronta dell'intero record ad ogni suo caricamento, mentre il controllo dell'impronta degli allegati viene fatto al momento dell'accesso ad essi. Un processo di *background* può comunque effettuare entrambi i controlli nei tempi in cui il sistema non è utilizzato.

L'aspetto da evidenziare è che l'amministratore può verificare l'integrità del record e dei file ad esso associati anche senza utilizzare Extraway, e quindi anche a distanza di tempo e su un altro sistema con comandi da *prompt*.

Anche in questo caso Extraway non nasconde, bensì dichiara.

Disponibilità

La disponibilità va riferita al servizio, ai record e alla loro intelligibilità per chi ne ha i diritti.

La disponibilità del servizio non è tanto determinata dal formato delle informazioni ma dalla robustezza del motore e dalle misure di prevenzione degli eventi dannosi (intrusioni, manipolazioni). Il formato in chiaro permette però agli amministratori di evidenziare con più modalità e strumenti la validità almeno formale di file e configurazioni.

La disponibilità delle informazioni e la loro intelligibilità beneficia in modo evidente dell'organizzazione e del formato aperto. Quando i dati ed i file associati sono in chiaro e fisicamente adiacenti il loro orizzonte temporale si estende oltre quello dei programmi che li producono e li gestiscono.

I record sono spesso un patrimonio primario di un'organizzazione e il loro mantenimento rischia di non essere economicamente sostenibile se richiede una conversione del pregresso ad ogni cambio di tecnologia *software*.

Supporto nativo del linguaggio XML

Extraway memorizza tutti i dati e metadati di ogni record in un'unica porzione indivisibile di un opportuno file XML. L'allocazione del file e la sua posizione in *sub-directories* è lasciata ad una politica di gestione scelta dall'amministratore dell'archivio, non dal programmatore.

Ogni porzione XML, a parte i casi in cui viene cifrata perché contenente dati sensibili, è intellegibile a chi possa accedere fisicamente al file, tipicamente l'amministratore del server o i conservatori delle copie di *backup*. Il motore di Extraway riconosce la struttura XML del record creando uno specifico indice per ogni diverso percorso dalla radice del record ai suoi elementi ed attributi ed ogni altro indice esteso frutto dell'elaborazione dei contenuti del record.

Durante il salvataggio del record, Extraway verifica che sia ben formato e controlla la sua validità rispetto ad un modello dichiarato (DTD o schema).

Gli approcci ai quali si deve far fronte sono normalmente due.

Strutture dati XML già esistenti, provenienti dal *porting* di basi di dati espresse in altro formato e quindi frutto di una conversione non sempre razionale o pensata per ottimizzare le prestazioni dell'applicazione che si andrà a realizzare. Gli stessi file da associare ai record identificati possono essere dislocati nel *file system* anche in modo scarsamente razionale. In un simile scenario è Extraway ad adattarsi alla natura dei file XML, a modellare la creazione dei

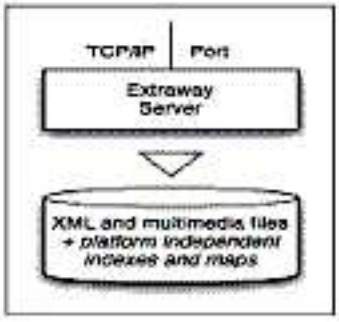
canali di ricerca sulla base di quanto disponibile con o senza la presenza di DTD atte a dichiarare il corretto formato delle informazioni. L'archivio risultante può avere limiti di portabilità ai quali si può porre rimedio con una riorganizzazione dei file d'archivio per mezzo delle politiche di distribuzione sul *file system* evidenziate nei paragrafi successivi.
Strutture realizzate espressamente per rappresentare le informazioni nel modo più coerente e razionale. Studiate e create con l'ausilio dei *tool* di Extraway, conducono alla realizzazione di archivi assolutamente portabili e ad applicazioni in grado di sfruttare pienamente le potenzialità degli indici che Extraway produce.

Architettura del *software* di base

Il cuore di Extraway è il motore. È costituito da un server multiprocesso, *multithread*, scritto in linguaggio C/C++ disponibile sui sistemi operativi più popolari:

Sparc Solaris
Linux x86
Linux on PowerPC
AIX on PowerPC
Windows
Mac OS X on Motorola
Mac OS X on Intel.

La figura successiva mostra il posizionamento del server Extraway.



Il processo è in ascolto su una porta TCP/IP configurabile. I file degli archivi sono di proprietà ed accesso esclusivo del server. I file sono composti da:

1. file di dati XML come *repository* dei record
2. allegati informatici associati ai record (testi, immagini, video)
3. mappe ed indici.

Tutti i file sono indipendenti dalla piattaforma, cioè possono essere utilizzati senza trasformazioni sui diversi sistemi operativi.

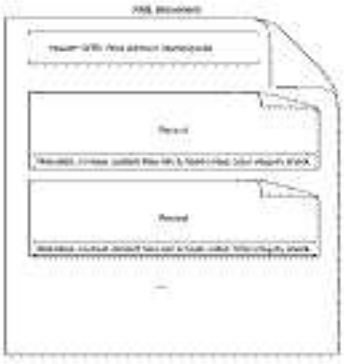
Come annunciato nel precedente paragrafo, la portabilità di un archivio può dipendere dall'organizzazione originaria delle informazioni e dei file a corredo. Di seguito vedremo come Extraway provvede a compiere una razionale distribuzione sul *file system* dei file ove le informazioni vengono memorizzate.

Architettura dei record

La base informativa non è una riga o un file XML, ma un *record* o *information unit*. Esso è l'aggregazione ottimale di informazioni per la quale è necessario assicurare l'integrità, i diritti di accesso, il *versioning* e la storia di accesso. La grande differenza con i database relazionali è che le operazioni sui record sono atomiche: i record sono intrinsecamente memorizzati, aggiornati e recuperati in un comando indivisibile. Ogni operazione viene salvata dal sistema di tracciamento di Extraway che registra tutte le manipolazioni di dati e contenuto, in modo da ricostruire tutte le sue versioni dal momento della sua prima sottomissione.

Pensato per realizzare prevalentemente applicazioni di natura documentale nelle quali la "fine" della vita utile di un documento consiste nel suo annullamento, il server non cancella record o allegati informatici ad esso associati se non per motivi di manutenzione straordinaria.

Il diagramma successivo mostra la disposizione dei record entro un file XML.

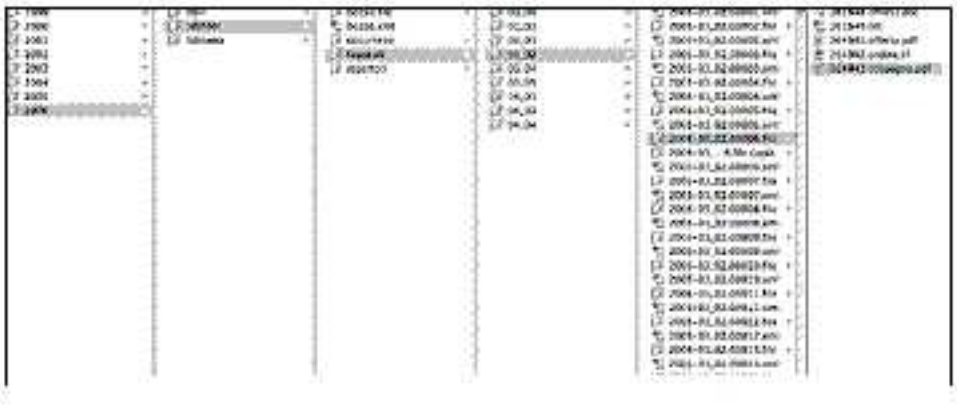


I record sono memorizzati all'interno di contenitori XML secondo una politica di allocazione definita dall'amministratore dell'archivio e indipendente dall'applicazione. L'amministratore può aggiornare la politica, cioè le regole di allocazione fisica dei record nei file senza impatto sull'applicazione.

Ci sono ovvi motivi di convenienza nell'impostare politiche che assemblano record che trattano la stessa attività o argomento nello stesso file. In questo modo un file XML contiene tutti i record della stessa procedura, ottimizzando le *performance* (la consultazione dei record della stessa attività viene velocizzata dalle funzionalità di *caching*) e la intelligibilità a lungo termine.

Ogni record è intrinsecamente legato ai file ad esso associati mediante impronte ed un codice di integrità totale. Anche per questi file la memorizzazione fisica tende a rispecchiare quanto già detto per i record. Ne consegue che l'organizzazione dei file su disco ricalca fedelmente l'organizzazione dei record nei file XML. I file sono adiacenti ai record cui sono associati e non memorizzati in un differente *repository*.

Una disposizione tipica è quella riportata in figura.



Nell'esempio, l'archivio è fisicamente organizzato in *directory*, una per anno. Nell'ambito delle registrazioni di ogni anno viene automaticamente creata una *directory* per ogni area produttrice di record. Per essa vengono create diverse *directory* secondo le tipologie di oggetti: bozze, documenti, fascicoli, repertori, ecc. Il prossimo livello è determinato dai valori di classificazione, per esempio 03_02 corrisponde a Titolo=3 e Classe=2 del piano di classificazione concordato per l'archivio in esame.

Da notare l'intelligibilità dall'alto verso il basso di archivi che oggigiorno facilmente arrivano a contenere centinaia di migliaia o milioni di record con i corrispondenti file associati.

Da notare inoltre che i nomi dei file sono auto-esplicativi:

2006-03.02.00006.xml

è il contenitore di tutti i record del fascicolo n. 6, titolo 3, classe 2, anno 2006 della corrente area.

La *directory* con lo stesso prefisso e posizione

2006-03.02.00006.file

ospita tutti i file associati ai record del fascicolo. Essi assumono un nome univoco, mantenendo il loro formato originale che, se proprietario, può essere automaticamente convertito secondo specifiche aperte (OpenDocument, PDF, ecc.).

Metadati

Ogni record viene arricchito automaticamente di alcuni metadati atti ad identificarne la vita utile e, laddove l'applicazione lo preveda, anche di altri metadati che assumono un particolare significato nell'ambito applicativo.

In particolare ogni record viene corredato dai seguenti metadati:

- Data e ora di creazione
- Utente creatore
- Data e ora di ultima modifica
- Ultimo utente modificatore
- Identificativo e versione del motore Extraway
- Identificativo e versione della struttura organizzativa
- Identificativo e versione del regolamento di accesso o manuale di gestione
- Identificativo e versione del piano di classificazione
- Codice di integrità.

Si noti che questi metadati vengono espressi sotto forma di *processing instruction* al fine di non causare impatti sul formato XML del record.

Qualora il formato del record venga disegnato seguendo i principî delle applicazioni documentali, in esso possono trovare posto altri metadati che tracciano capillarmente tutti gli interventi salienti avvenuti sul documento e sui file ad esso associati, indicandone data, ora e gli estremi degli operatori che li hanno compiuti.

Infine, utilizzando un modulo dell'ambiente, il *file conversion service*, è possibile aggiungere ulteriori metadati utili sia alla conservazione dell'intelligibilità che alle

operazioni di selezione per ogni allegato informatico, oltre ovviamente ad estrarne il testo per l'indicizzazione *full text*:

Impronta del file
Numero di pagine
Dimensione
Dettagli tecnologici (formato, compressione, risoluzione, durata, testo, sotto-titoli, ecc.).

Lo stesso strumento si presta alla trasformazione dell'allegato originario in altri formati standard (PDF, XML ed altri) utili particolarmente per la pubblicazione Web dei loro contenuti.

Funzionalità di ricerca

Come qualsiasi database, Extraway consente ricerche di informazioni secondo diverse modalità.

L'approccio a dati "noti" tramite ricerche su codici o altri valori specifici dei documenti interessa un preciso capitolo delle funzionalità di selezione ed una particolare modalità di ottimizzazione.

Particolare attenzione, però, viene posta nel reperimento di record appartenenti a basi di dati documentali la cui vastità, ricchezza ed eterogeneità dei contenuti richiedono di avere un approccio totalmente diverso.

Extraway è predisposto per supportare le funzionalità necessarie alla ricerca nell'ambito di basi di conoscenza, quindi la sua architettura privilegia la ricerca ed il recupero di record sulla base di criteri approssimati e progressivi in presenza di molti accessi concorrenti su grandi archivi. Oltre alle classiche funzioni di *information retrieval* (*vector space*, ricerche per somiglianza, *wild-card*, operatori booleani e di prossimità) Extraway supporta estensione dei termini mediante thesaurus, compresenza di termini in un dato bacino o *ancestor* XML, nonché la produzione dei termini più significativi di un sottoinsieme di record. A questi indici, per particolari applicazioni, vengono affiancati indici basati su basi di conoscenza, indici ricavati quindi dai significati e non semplicemente dai termini: un primo passo verso il concetto di ontologia negli *information retrieval*. Non finisce qui: là dove la collocazione dell'informazione nello spazio ha particolare rilevanza, possono essere realizzati anche indici spaziali per la ricerca di oggetti georeferenziati. Le modalità di reperimento ed accesso alle informazioni presenti nei database e la versatilità del loro utilizzo combinato rendono Extraway uno strumento ideale per la realizzazione di applicazioni orientate ad Internet.

Vengono inoltre gestiti a livello di motore i comandi più utili dell'interfaccia uomo-macchina:

Creazione di sottoinsiemi temporanei di record
Inclusione ed esclusione manuale di record
Combinazione booleana tra sottoinsiemi (intersezione, unione, differenza)
Definizione di ricerche ricorrenti
Copia e aggiornamento di collezioni di record per la consultazione pubblica.

La realizzazione di applicazioni con eXtraWay

Costruzione di applicazioni

Le applicazioni possono essere costruite in diversi modi.

Prima di tutto c'è la possibilità di creare applicazioni web senza conoscere linguaggi di programmazione, ma utilizzando un *tool* grafico per definire un modello, generare una interfaccia generica, personalizzarne eventualmente l'aspetto ed iniziare direttamente ad inserire record. Questa strada è rivolta ai non programmatori per definire applicazioni semplici ma efficaci.

Un altro modo è utilizzare il modellatore grafico XML per definire la struttura e le proprietà dell'archivio per poi realizzare un'applicazione richiamando le classi del motore mediante un'apposita libreria base in Java, chiamata Extraway Broker. Con l'ausilio di questo approccio possono essere realizzate *web application* di grande efficacia, con interfacce utente caratterizzate da una ricca *business logic* adeguata a soddisfare i casi più complessi.

Extraway dispone inoltre di un ricco insieme di servizi web che possono essere invocati dai più diffusi linguaggi di programmazione (Java, C#, PHP, ecc.).

Disegno del modello e dell'interfaccia

Extraway Model Designer è il modulo che permette di creare un archivio e di definire la struttura e le proprietà dei suoi record. La struttura viene descritta secondo una grammatica XML ed alcuni tipi di elemento vengono definiti *record*.

Prima di tutto il Model Designer è un *editor* grafico di DTD [Data Type Definition] con possibilità di importare da un campione la configurazione iniziale. Oltre all'impostazione della composizione, cardinalità, alternatività, ecc., esso consente di impostare proprietà non previste in una DTD quali:

Tipo di valore (numero, data, ecc.)

Proprietà di indicizzazione: stringa intera, parola per parola, entrambe, trasparente, salta, ecc.
Indice derivato: ad esempio indice del campo anno di una data
Indice complesso: ad esempio per concatenazione del dato con altri elementi e attributi
Indice basato su una funzione del dato
Indice esteso, creato da una libreria esterna
Regole di unicità di record
Regole di allocazione del record su file e *directory* del *file system*
Valori di *cache* del record per un accesso immediato in memoria
Thesaurus di validazione del dato.

Il risultato è una struttura d'archivio rappresentata da una DTD e da un file di configurazione che descrive *in toto* le attività che Extraway svolgerà sui singoli record.

Su questa coppia di file poggia ogni successiva realizzazione, dalla più generica alla più specifica.

Il modulo di disegno delle maschere di interfaccia

Extraway Form Editor è lo strumento per la creazione di semplici ed efficaci interfacce senza scrivere una riga di codice. Dato un modello XML esso genera automaticamente le maschere di ricerca, visualizzazione singola e multipla, inserimento, modifica, visualizzazione e scelta indici di campo.

L'*editor* consente di personalizzare l'aspetto ed i tipi di controllo passando da una vista strutturale della maschera alla vista della struttura dei dati. È disponibile una libreria di base di controlli che prevede definizione di blocchi ripetibili, con annidamento, con campi di tipo linea, testo, numerico, data con calendario associato, liste di opzioni statiche, liste di opzioni da thesaurus dinamico, liste di consultazione uffici e persone interne, liste di consultazione anagrafica soggetti e persone esterne, controlli di selezione o scansione di immagini, ecc.

La *console* di amministrazione

La *console* di amministrazione è la *home page* per l'amministratore. Essa consente di inizializzare un archivio, mappare ed indicizzare file XML preesistenti, effettuare ricerche, navigazioni, inserimenti e modifiche con una interfaccia standard che non

richiede alcuna predisposizione. Essa consente operazioni straordinarie come compattamenti, riallocazione dei record secondo nuove politiche di distribuzione sul *file system*, test di connessione, ecc.

Interfacce di programmazione

Il programmatore può interagire con il motore di Extraway utilizzando:

- Le classi Java del modulo Extraway Broker
- La libreria Extraway Web Services
- Il linguaggio di *template* XHTML di Extraway
- Il modulo Extraway JDBC
- I.R. Model. Un insieme di classi Java che ricalca i metodi di programmazione dei più diffusi motori *open source* di *information retrieval*.

Extraway Broker

Extraway Broker è un *package* Java che espone quasi tutti i comandi del motore, le cui principali classi riguardano:

- Connessione
- Archivio
- Record
- Insieme di record
- Indici
- Thesaurus
- e molto altro ancora...

Esso rappresenta il modo più efficiente di interazione con Extraway.

Web Services

Gran parte dei comandi di Extraway sono disponibili come servizi web, corredati da esempi, tra i quali:

- Ricerca secondo tantissime opzioni
- Recupero di record e file di contenuto
- Bloccaggio/sbloccaggio di un record
- Salvataggio di un nuovo record o modifica di uno esistente
- Consultazione ed aggiornamento file associati ai record
- Gestione sottoinsiemi di risultati
- Consultazione indici e tesauri
- Gestione di gerarchie di record
- e altri.

Extraway Template Engine

Extraway contiene un motore di modelli (*template*) di presentazione. Basato su un insieme di fogli XSL, esso ha lo scopo di valorizzare maschere tipo, disegnate da un grafico web, con i dati XML provenienti in tempo reale dal motore di Extraway. Il flusso dei dati è il seguente:

la risposta del motore di Extraway ad un qualsiasi comando è in formato XML (sia nel caso di un record che dell'identificativo di un insieme di record o di termini);
ogni tipo di risposta XML, in relazione ad ogni singolo tipo di comando, referenzia un foglio di stile XSL specifico;
il foglio di stile referenzia un modello (*template*), tipicamente in formato XHTML;
il *template* viene processato eseguendo le azioni in esso contenute (ad esempio, inserendo il valore di *default* su un campo di *input* della maschera prendendolo dai dati attuali, o ripetendo l'elemento attuale tante volte quanto è contenuto in un dato elemento XML, ecc.);
alla fine il *template* viene restituito al chiamante (tipicamente un web server java).

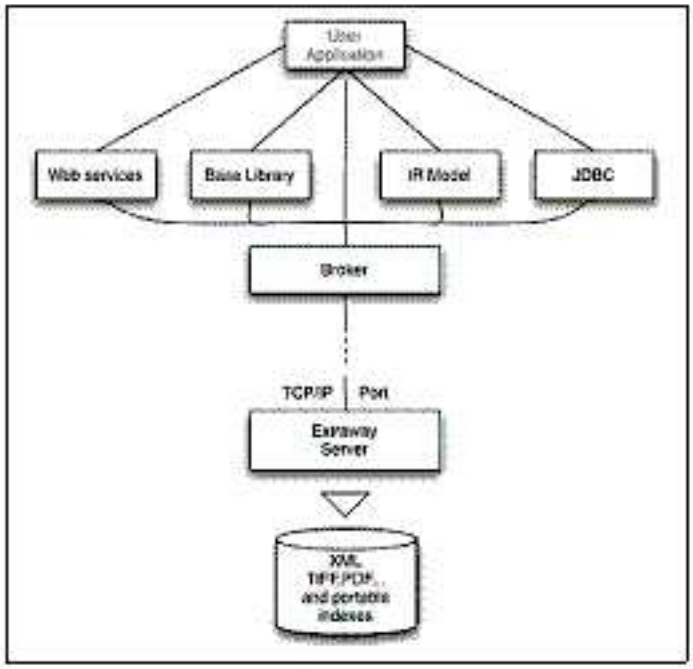
In modo speculare, l'invio di un comando al server è generato da una maschera opportunamente compilata da parte dell'utente. La sottomissione al server dei contenuti della maschera produce l'assemblaggio di tutti i valori in un pacchetto XML che viene spedito al motore per l'inserimento o l'aggiornamento ovvero l'esecuzione di qualsiasi altro comando.

Il motore viene fornito con *template* per gli stati interattivi più frequenti:

Inserimento di un record
Aggiornamento di un record
Navigazione degli indici
Navigazione di thesaurus
Navigazione tra gerarchie di record
Consultazione di valori di elementi e attributi sulla base di un *pattern*
Maschera di ricerca
Vista di un singolo record
Vista di una aggregazione di record.

Gli scenari applicativi

La realizzazione di un'applicazione secondo uno degli schemi descritti porta ad uno scenario che può essere riassunto tramite la seguente figura.



Nell'osservare questo schema va posta particolare attenzione al fatto che esso è principalmente composto da moduli indipendenti dalla piattaforma. Solo il motore Extraway dipende dalla piattaforma (Windows, Unix, Linux, Mac).

Per concludere, dal momento che Extraway trova una naturale applicazione nella realizzazione di piattaforme documentali, lo schema a p. 117 mostra come lo scenario descritto può essere parte di un più vasto ambiente corredato da moduli, quali un *file conversion server*, *mail and fax server*, librerie di sicurezza per l'accesso e l'utilizzo di *access controll lists* e così via.

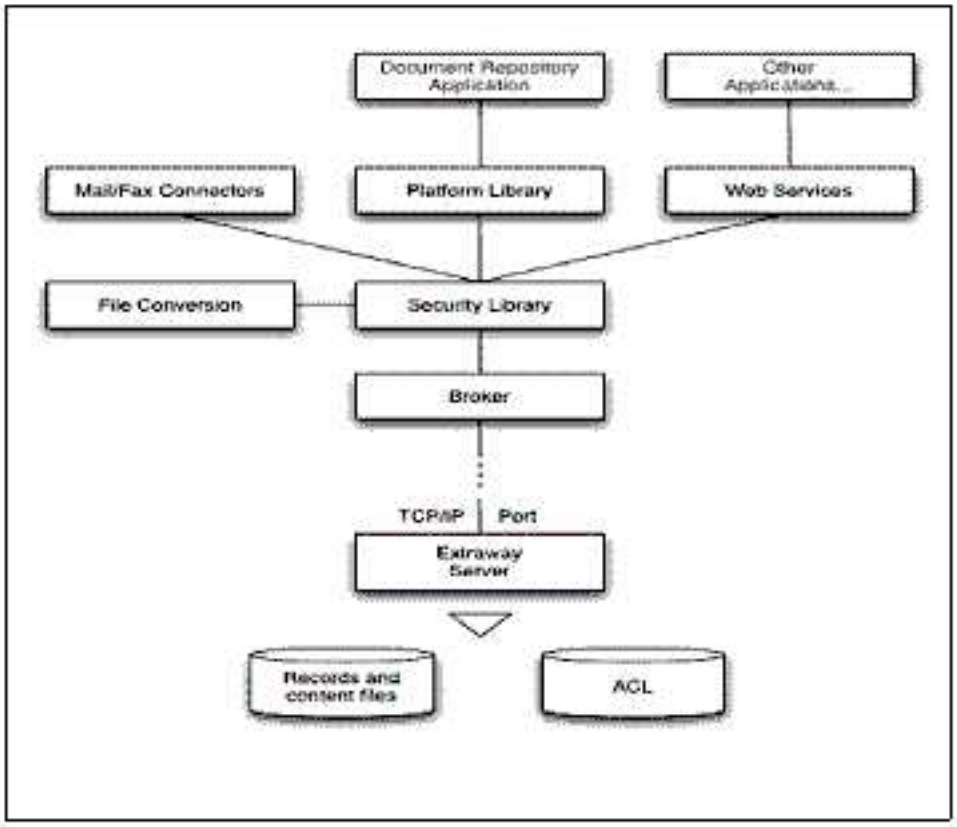
Scalabilità degli scenari

Tutti gli scenari visti nel corso dei precedenti paragrafi prevedono una struttura "3 Tiers" ovvero la presenza di tre strati *software* che si occupano di:

presentazione delle informazioni raccolte e controllo delle informazioni inserite dall'utente (*web server*);

elaborazione delle informazioni inserite dall'utente e gestione delle comunicazioni con la banca dati (*application server*);

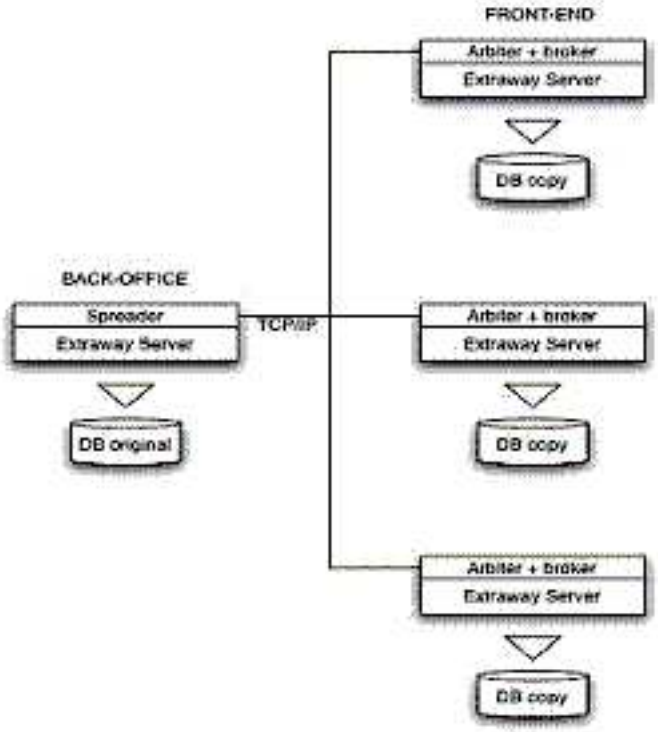
gestione dei dati vera e propria (*database server*).



Una simile struttura si presta naturalmente a consentire una scalabilità verticale e, se le condizioni lo consentono, anche orizzontale. Il primo ed il secondo strato *software* possono essere scalati orizzontalmente senza limitazioni mentre il terzo, vale a dire il *database server* ovvero Extraway, può essere scalato orizzontalmente solo in alcuni casi.

Quella rappresentata nella figura di p. 118 è la configurazione più ricca e "scalata" che è attualmente concessa da Extraway. Essa può essere applicata quando esista una netta separazione tra una rete di redazione (*back office*) ed una rete di sola consultazione (*front office*).

Sulla sinistra è raffigurato il *back office* contenente l'archivio primario. Sulla destra stanno i server secondari, ognuno con una copia completa e autonoma dell'archivio mantenuta aggiornata dal server di *back office* nei momenti di sua validazione.



I server del *front office* sono governati da un modulo *Arbitro*. Esso può essere presente in più postazioni del *front office* ma in un dato istante nel tempo solo uno di essi è attivo. L'*Arbitro* opera in comunicazione con i *broker* di tutte le postazioni *front*.

Quando un archivio è considerato valido per la trasmissione verso il *front office*, un modulo di propagazione (*spreader*) si connette con il corrente arbitro primario e concorda la trasmissione dell'aggiornamento di un archivio ad un server secondario il quale viene preventivamente disattivato, cioè posto in uno stato tale per cui le prossime richieste degli utenti vengono girate dal corrispondente *broker* verso gli altri server.

La propagazione riguarda solo i file di dati XML inseriti o modificati nonché il file contenente gli indici e le mappe. Dal momento invece che gli allegati informatici (PDF, TIFF, DOC, ecc.) non vengono mai modificati, vengono eventualmente spedite solo le nuove versioni.

La propagazione prosegue con altri server secondari fino al momento in cui quelli contenenti l'archivio aggiornato rappresentano la maggioranza. A questo punto vengono invertite le parti: i server aggiornati con la versione corrente dell'archivio vengono posti in modalità attiva mentre gli altri, precedentemente consultabili, in modalità disattiva. La procedura di aggiornamento dello *spreader* proseguirà aggiornando i server residui che verranno via via riabilitati.

Questo processo può risultare trasparente per l'utente, il quale può continuare le sue operazioni perché anche i file di risultati temporanei residenti su un sistema disattivato sono resi disponibili agli altri server. Ciò è possibile grazie al fatto che gran parte dei comandi è senza stati e lo stato dell'applicazione viene restituito all'utente in modo da poter essere ripresentato per il prossimo comando anche ad un altro server.

In caso di rottura di un server di *front-end*, il modulo arbitro primario è in grado di reagire ridirezionando i prossimi comandi agli altri server. Parimenti, in caso di guasto della postazione dell'arbitro primario, una qualsiasi altra postazione *front* può provvedere a sostituirsi ad esso garantendo la continuità del servizio.