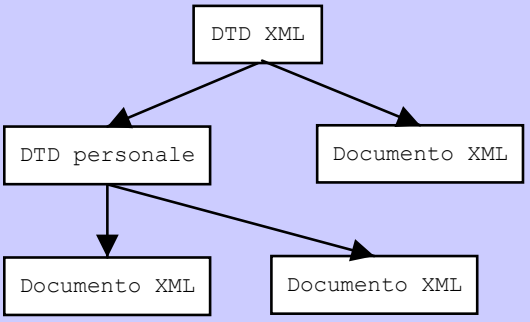
Seminario
Basi di dati XML

A cura di Sergio Iacobelli

Sommario

Parte I: XML e Database

DTD



DBMS XML

- Sono vantaggiosi per gestire documenti XML come moduli, documenti
- Con informazioni con troppe referenze i dati si denormalizzano
- L'integrità referenziale nei DBMS XML (esterna e interna) è debole, cioè si possono introdurre meccanismi non standard di integrità referenziale (utilizzo di ID/IDREF o key/keyref)
- Esistono diverse tipologie di DBMS XML

XML e Database: il problema

- Problema:
  - è possibile/necessario memorizzare documenti XML in un DBMS?
  - Quale tecnologia è necessaria a questo scopo?
- Risposta:
  - è certamente possibile memorizzare e gestire documenti XML in un DBMS
  - la tecnologia necessaria a questo scopo dipende dal perché vogliamo gestire documenti XML in un DBMS

Tipologie di documenti XML

- Due possibili usi per documenti XML:
  - ***Data Centric***: i documenti possono rappresentare lo strumento con il quale dati tradizionali (es. relazionali) vengono trasferiti su Web
    - XML come veicolo per trasporto di dati
    - Esempio: ordini di vendita, scheduling di voli, menù
  - ***Document Centric***: l'informazione è rappresentata dal documento in sé
    - XML come modello per la rappresentazione dei dati
    - Esempio: libri, documenti in genere

Documenti Data Centric

- Struttura regolare
- livello di dettaglio piuttosto fine
- contenuto omogeneo
- l'ordine con cui gli elementi allo stesso livello appaiono è ininfluente
- Utilizzati per "machine consumption"
- Esempi: ordini di vendita, scheduling di voli, menù,…

Esempio: ordini di vendita

```
<Orders>
  <SalesOrder SONumber="12345">
    <Customer CustNumber="543">
      <CustName>ABC Industries</CustName>
      ...
    </Customer>
    <OrderDate>981215</OrderDate>
    <Line LineNumber="1">
      <Part PartNumber="123">
        <Description>
            Turkey wrench: Stainless steel, one
  piece...
        </Description>
        <Price>9.95</Price>
      </Part>
      <Quantity>10</Quantity>
    </Line>
    <Line LineNumber="2">
      ...
    </Line>
  </SaleOrder>
</Orders>
```


Documenti Document Centric

- Struttura irregolare
- Livello di dettaglio meno fine
- contenuto eterogeneo
- l'ordine degli elementi allo stesso livello è significativo
- in genere progettati per "human consumption"
- Esempi: libri, email, …

Product Description

```
<Product>
<Name>Turkey Wrench</Name>
<Developer>Full Fabrication Labs, Inc.</Developer>
<Summary>Like a monkey wrench, but not as big.</Summary>
<Description>
<Para>The Turkey wrench, which comes in both right- and left-
  handed versions ....</Para>
<Para>You can:</Para>
<List>
  <Item><Link URL="Order.htm">Order your turkey
  wrench</Link></Item>
  <Item><Link URL="Wrench.html">Read about
  wrenches</Link></Item>
  <Item><Link URL="catalog.zip">Download the
  catalog</Link></Item>
</List>
  ....
</Description>
</Product>
```


XML e DBMS

- Ciascuna tipologia di documenti richiede una particolare tecnologia per la sua gestione

data ⟷ Relational/object-oriented DB

document ⟷ DB basato su XML (XML è il modello dei dati)

XML e DBMS

- Due categorie di DBMS:
  - **XML-Native DBMS**:
    - comprendono un insieme di nuovi sistemi la cui architettura è stata progettata per supportare totalmente le funzionalità necessarie alla gestione di documenti XML
    - tecnologia non ancora matura
    - utili per Document Centric
    - Esempio: Tamino
  - **XML-Enabled DBMS**:
    - comprendono tutti i DBMS che mantengono integra la propria architettura estendendola con funzionalità necessarie alla gestione di documenti XML
    - sono tipicamente Object-Relational (Oracle 9i,…)
    - utili per Data Centric e parzialmente per Document Centric
  - L'approccio dipende dal tipo di documento (data o document-centric) e dal tipo di operazioni che si vuole effettuare:
  - Memorizzazione di un XML document in un campo di tipo CLOB (or BLOB).
  - Memorizzazione del documento XML in una struttura relazionale.

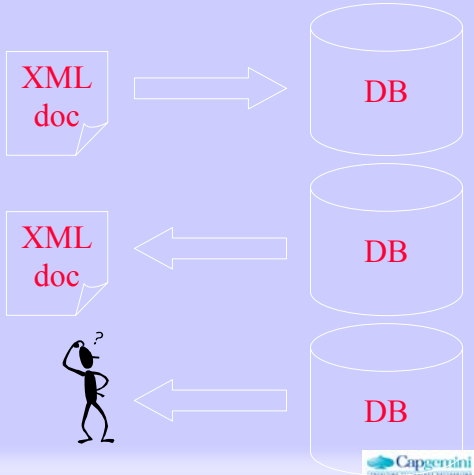
XML e DBMS

- Nel seguito.
  - Problematiche relative alla gestione di documenti Data Centric e Document Centric in XML-Enabled DBMS

XML-Enabled DBMS e documenti data centric

Problematiche per Data Centric

- Tre problematiche di base:
  - come rappresentare i dati contenuti nei documenti XML nel DBMS
  - come generare documenti XML partendo dai dati contenuti nel DBMS
  - come interrogare i dati estratti da documenti XML



Rappresentazione dati

- È necessario definire un mapping tra la struttura dei documenti XML e lo schema del DB
  - ➔Per memorizzare i dati contenuti in un documenti XML in un DB, deve esistere una o più tabelle con lo schema richiesto dal mapping
- rappresentazione strutturata
- Vantaggi:
  - approccio piuttosto semplice
  - i dati sono facilmente interrogabili
- Svantaggi:
  - Scarsa flessibilità: la tabella deve essere conforme al documento
  - il documento di partenza non è più recuperabile

1. DBMS relazionale

- Un documento XMl viene rappresentato come una singola tabella o un insieme di tabelle
- la struttura del documento XML è simile alla seguente:

```
<database>
  <table>
    <row>
      <column1>...</column1>
      <column1>...</column1>
            ...
    </row>
    ...
  </table>
</database>
```

- approccio tipico per DBMS relazionali, object-relational

Esempio

Documento XML

```
<clienti>
  <row>
    <numero> 7369 </numero>
    <nome> PAUL </nome>
    <cognome> SMITH </cognome>
  </row>
  <row>
    <numero> 7000 </numero>
    <nome> STEVE </nome>
    <cognome> ADAM </cognome>
  </row>
</clienti>
```

Tabella Clienti

Numero	Nome	Cognome
2000	MIKE	SCOTT
7369	PAUL	SMITH
7000	STEVE	ADAM

Capgemini

Esempio

Documento XML

```
<clienti>
  <row>
    <numero> 7369 </numero>
    <lista_clienti>
      <cliente>
        <nome> PAUL </nome>
        <cognome> SMITH </cognome>
      </cliente>
      <cliente>
        <nome> STEVE </nome>
        <cognome> ADAM </cognome>
      </cliente>
    </lista_clienti>
  </row>
</clienti>
```

Tabella Lista_Clienti

Numero
2000
7369

Tabella Clienti

Numero	Num_cliente	Nome	Cognome
2000	1	MIKE	SCOTT
7369	2	PAUL	SMITH
7369	3	STEVE	ADAM

Capgemini

2. DBMS object relational

- Il documento può sempre essere mappato in una singola tabella, utilizzando campi strutturati

Esempio

Documento XML

```
<clienti>
  <row>
    <numero> 7369 </numero>
      <cliente>
        <nome> PAUL </nome>
        <cognome> SMITH </cognome>
      </cliente>
  </row>
  <row>
    <numero> 7000 </numero>
      <cliente>
        <nome> STEVE </nome>
        <cognome> ADAM </cognome>
      </cliente>
  </row>
</clienti>
```

Tabella Clienti

Numero	Cliente
2000	nome: MIKE, cognome: SCOTT
7369	nome: PAUL, cognome: SMITH
7000	nome: STEVE, cognome: ADAM

Esempio

Documento XML

```
<clienti>
  <row>
    <numero> 7369 </numero>
    <lista_clienti>
      <cliente>
        <nome> PAUL </nome>
        <cognome> SMITH </cognome>
      </cliente>
      <cliente>
        <nome> STEVE </nome>
        <cognome> ADAM </cognome>
      </cliente>
    </lista_clienti>
  </row>
</clienti>
```

Tabella Clienti

Numero	Cliente
2000	nome: MIKE, cognome: SCOTT
7369	CLIENTE: nome: PAUL, cognome: SMITH CLIENTE: nome: STEVE, cognome: ADAM

Interrogazione dati

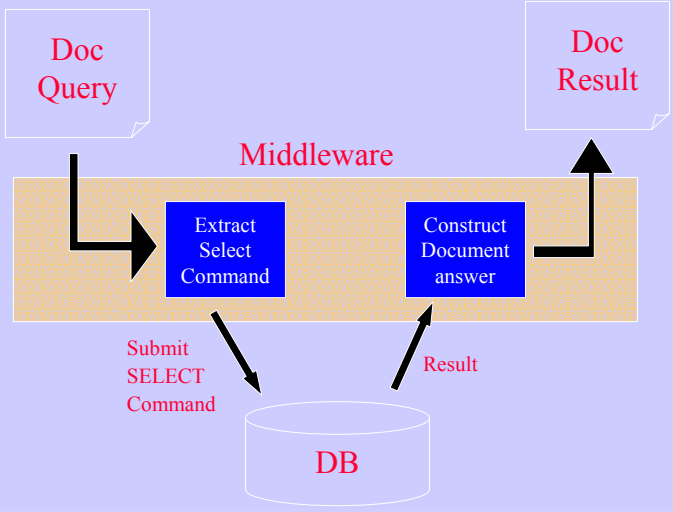
- Poiché i dati vengono rappresentati secondo il modello supportato dal DBMS (es. relazionale), è possibile utilizzare i linguaggi supportati dal DBMS per l'interrogazione dei dati memorizzati
- approccio template-based:
  - la query viene rappresentata nel documento XML
  - necessità di middleware

Flight Information

```
<?xml version="1.0">
<FlightInfo>
   <Intro>The following flights have available seats:</Intro>
   <SelectStmt>
     SELECT Airline, FltNumber, Depart, Arrive FROM Flights
   </SelectStmt>
   <Conclude>We hope one of these meets your needs</Conclude>
</FlightInfo>
```

```
<?xml version="1.0">
<FlightInfo>
   <Intro>The following flights have available seats:</Intro>
   <Flight>
     <Row>
        <Airline>ACME</Airline><FltNumber>123</FltNumber>
        <Depart>Dec 12, 1998 13:43</Depart><Arrive>...<Arrive>
     <Row>
   </Flight>
   <Conclude>We hope one of these meets your needs</Conclude>
</FlightInfo>
```


Interrogazione dati



Generazione documenti XML

- **Problema**: fornire una rappresentazione XML ai dati recuperati tramite query dal DBMS
- si utilizza il mapping inverso rispetto a quello utilizzato per la memorizzazione
- operazione importante per attribuire un formato standard ai dati ritrovati, prima di inviarli sulla rete

Esempio

```
SELECT nome, cognome
FROM Clienti
WHERE Numero = "7369
```

Tabella Clienti

Numero	Nome	Cognome
2000	MIKE	SCOTT
7369	PAUL	SMITH
7000	STEVE	ADAM

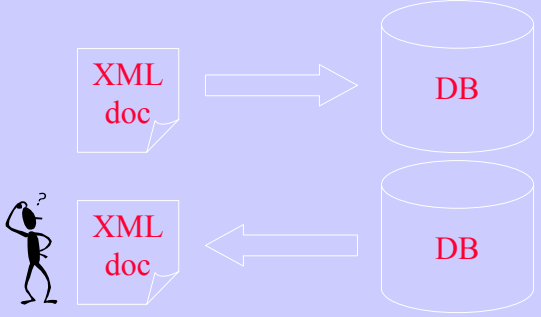
Documento XML

```
<clienti>
  <row>
    <nome> PAUL </nome>
    <cognome> SMITH </cognome>
  </row>
</clienti>
```

XML-Enabled DBMS e documenti Document Centric

Problematiche per Document Centric

- Due problematiche di base:
  - come rappresentare i documenti XML nel DBMS
  - come interrogare i documenti XML



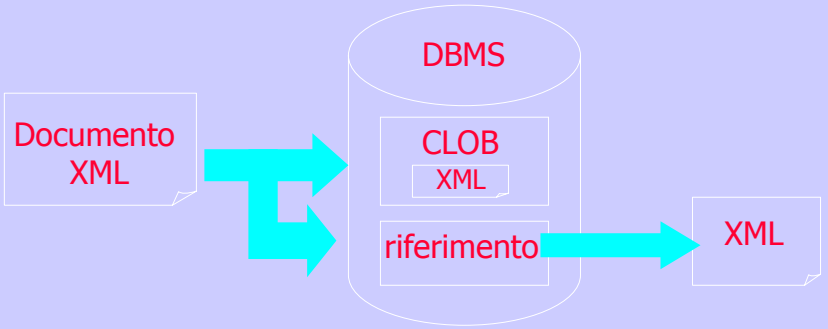
Rappresentazione

- Permette di mantenere integro il documento XML
- Due approcci:
  - rappresentazione non strutturata
    - documento come unico oggetto
  - rappresentazione ibrida
    - documento parzialmente rappresentato secondo la rappresentazione strutturata e parzialmente secondo la rappresentazione non strutturata

Rappresentazione non strutturata

- Il documento viene tipicamente mappato in un singolo campo di una tabella di tipo:
  - CLOB (Character Large Object): il documento è fisicamente contenuto nel campo della tabella
    - alcuni DBMS (IBM DB2) supportato tipi ad hoc: XMLVARCHAR
  - riferimento: il campo contiene il riferimento al documento, memorizzato altrove, sul file system
- Vantaggi:
  - flessibile
- Svantaggi:
  - i dati sono non strutturati
  - interrogazione più complessa
  - la tabella può contenere documenti eterogenei (diversi DTD)

Rappresentazione non strutturata



Esempio

Documento XML

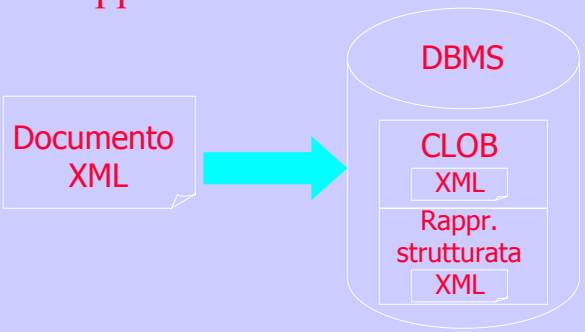
```
<clienti>
  <row>
    <numero> 7369 </numero>
    <nome> PAUL </nome>
    <cognome> SMITH
  </cognome>
  </row>
  <row>
    <numero> 7000 </numero>
    <nome> STEVE </nome>
    <cognome> ADAM
  </cognome>
  </row>
</clienti>
```

Tabella Clienti

Id	Documento_XML
10	`<clienti> <row> <numero> 7369 </numero> <nome> PAUL </nome> <cognome> SMITH </cognome> </row> <row> <numero> 7000 </numero> <nome> STEVE </nome> <cognome> ADAM </cognome> </row> </clienti>`

Rappresentazione ibrida

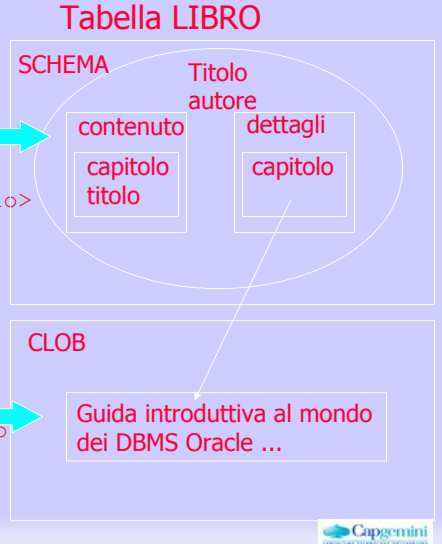
- Rappresentazione che combina rappresentazione strutturata e non strutturata



Esempio

Documento XML

```
<libro>
  <titolo> Oracle Guide </titolo>
  <autore> M. Abbey  </autore>
  <contenuto>
    <capitolo n='1'>
      <titolo> Introduzione </titolo>
      …
    </capitolo>
    …
  </contenuto>
  <dettagli>
    <capitolo n='1'>
      <sezione n='1'>
        Guida introduttiva al mondo
            dei DBMS Oracle …
      </sezione>
    </capitolo>
  </dettagli>
```



Content Management System

- Sono un altro tipo specializzato di base di dati XML-Native.
- Serve per il controllo dei documenti “human-written”.
- Permette di spezzare i documenti nei frammenti discreti, quali: gli esempi, le procedure, i capitoli, i metadati, i nomi dell'autore, le date di revisione ed i numeri di un documento.

Interrogazione documenti

- Dal punto di vista del DBMS, un documento memorizzato in modo non strutturato non è che un documento di testo
- in genere i DBMS supportano strumenti per ritrovare i documenti in base al contenuto
- nel caso di documenti XML, mettono a disposizione operatori avanzati da utilizzare in statement SQL per recuperare documenti XML in base al contenuto

Interrogazione in Oracle

- È possibile utilizzare un particolare motore di ricerca per testi
  - Intermedia Text (ne parleremo nel contesto Multimedia)
- utilizzando questo strumento è possibile abilitare ricerche sui vari elementi ed attributi di un documento XML, tramite un meccanismo di indicizzazione
- SQL viene esteso in modo da supportare predicati ad hoc per la ricerca in documenti XML

Interrogazione in Oracle

- Nuova funzione:
  - `CONTAINS(XML_COLUMN, QUERY_TAG)`
    - `XML_COLUMN`: colonna (attributo) in cui sono contenuti i documenti XML
    - `QUERY_TAG`: predicato che permette di specificare condizioni sui documenti XML
  - `QUERY_TAG::= <tag_value> WITHIN <tag_name> | <attribute_value> WITHIN <tag_name@attribute_name>|`
    …
  - `CONTAINS` restituisce un valore maggiore di 0 se la condizione è verificata

Esempio

- LISTA_CLIENTI(NUMERO, DOCUMENTO_XML)
- SELECT * FROM LISTA_CLIENTI
  WHERE COND >0;
- COND = CONTAINS (DOCUMENT_XML, 'PAUL WITHIN NOME')
  - determina tutti i documenti contenuti nel campo DOCUMENT_XML che contengono un tag NOME con valore PAUL
- COND = CONTAINS (DOCUMENT_XML, '1 WITHIN NUM@Cliente')
  - determina tutti i documenti contenuti nel campo DOCUMENT_XML che contengono un elemento Cliente con un attributo NUM di valore 1

XML e Oracle 9i

- XML-enabled
- supporta rappresentazione strutturata, non strutturata in campi CLOB e BFILE, e ibrida.
- Interrogazione, rappresentazione strutturata e non, tramite apposito linguaggio di interrogazione.
- Creazione di strutture nativa XML con indici nativi bitmap.
- generazione documenti XML a partire dal contenuto DB

XML e IBM DB2

- XML enabled
- supporta rappresentazione strutturata, non strutturata in campi ad hoc, e ibrida
- Nuovi tipi di dato:
  - XMLVARCHAR: documenti XML memorizzati come VARCHAR
  - XMLCLOB: documenti XML memorizzati come CLOB
  - XMLFILE: riferimento ad un documento XML, memorizzato su file system
- interrogazione rappresentazione non strutturata tramite:
  - operatori specifici, che permettono di navigare la struttura del documento
  - text extender, che supporta funzionalità aggiuntive di analisi del contenuto (ne parleremo nel contesto Multimedia)
- generazione documenti XML a partire dal contenuto DB

Parte II: XML in azienda

I dati

- Dati divisionali
- Dati progettuali (base dati)
- Dati gestiti dal singolo dipendente (documenti)
- Moduli (sia divisionali che non)
- Dati riepilogativi (datawarehouses e dataMart)
- Dati economico/finanziario (SAP)
- Dati statistici (base dati, SAS, e non)
- Dati decisionali (direttive aziendali)
- Altre tipologie

Organizzare i dati

- **Idea di base**: Conoscere le proprie entrate/uscite di dati (es. entrate/uscite per NAS).
- Elencare le proprie tipologie di dati divise in:
  - Dati strutturati provenienti dalle base dati
  - Tipologie di documenti
- Elencare a chi normalmente forniamo dati, definiamo le strutture.
- Elencare chi normalmente ci fornisce dati, definiamo le strutture.
- Definizione e creazione dei nostri DTD per tipologia di dato.
- Definizione dei fogli XSL, uno per fornitore di dati.
- Utilizzo di un parser DOM o SAX spedire o ricevere i dati.

Tools che forniscono dati XML

- Prodotti Microsoft in particolare Office (Word, Excel …dalla versione 2000).
- Prodotti Oracle (Desiner, Warehouse Builder, Rdbms, ecc.).
- SAS, SAP ecc.

- **In generale: Tutti i vendors ormai tendono ad esportare i propri dati in XML.**

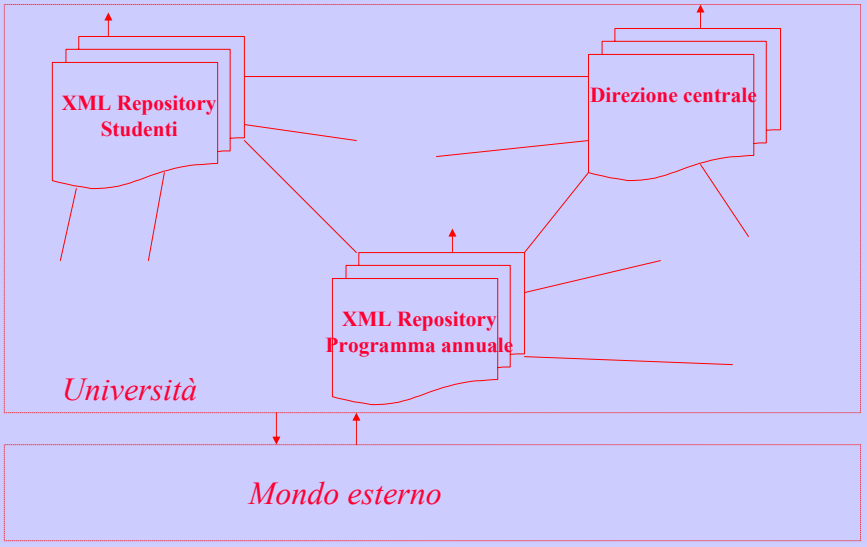
Il repository

- Idea di base: Creare un repository locale XML in modo da creare uno **Star Enterprise XML Repository**.
- Il repository contiene:
  - Modelli DTD dei propri dati
  - I Metadati (Descrizione dei propri dati)
  - I file di interscambio XML

Repository XML

- Basato su un DB in grado di memorizzare dati. XML in forma nativa, cioè utilizzo di un database Pure-XML.
- I più accreditati: Oracle 9i, DB2, Tamino.
- Creato su un DTD che descrive la base dati XML derivato dal DTD del CWM (common warehouse model).

Star Enterprise XML Repository

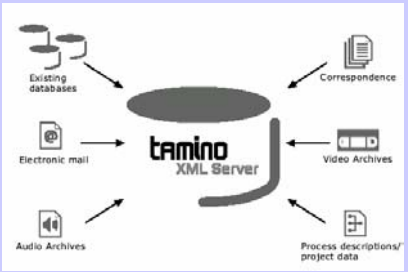


Ipotesi di progetto realizzativo del repository

- Realizzazione di un repository sperimentale.
- Creazione dei propri DTD e dei DTD dei più consueti creditori/debitori di dati.
- Costruzione di alcuni fogli XSL.
- Creazione di un applicativo fornisca i dati in modo semi-automatico (parser).
- *Presentazione del progetto.*

Parte III: XML Native - Tamino

DBMS XML Native: Tamino



Vantaggi di Tamino

- **Integrazione e scambio di dati:**
  Natura gerarchica di XML, come possibile descrizione degli oggetti che consistono in parecchi oggetti annidati, distribuiti su basi di dati multiple e eterogenee.
- **Amministrazione dei dati:**
  Tamino è un server che trae beneficio dalle tecnologie delle basi di dati come l'inidicizzazione, compressione, caching
- **Scalabilità:**
  Tamino realizza un alto grado di scalabilità in virtù della capacità di XML nel comprendere i tipi supplementari delle informazioni, senza invalidare le informazioni esistenti
- **Gestione:**
  La struttura arborescente e flessibile collegata di XML rende Tamino molto dinamico, nel senso che è facile aggiungere delle informazioni addizionali senza cambiare la struttura di base adattando i dati attuali
- **Sicurezza:**
  Oltre a fornire un meccanismo di autorizzazione al controllo di accesso agli oggetti XML memorizzati, Tamino integra i concetti di sicurezza ai livelli differenti (per esempio, a livello di trasporto e applicazione).

XML Native contro XML Enabled

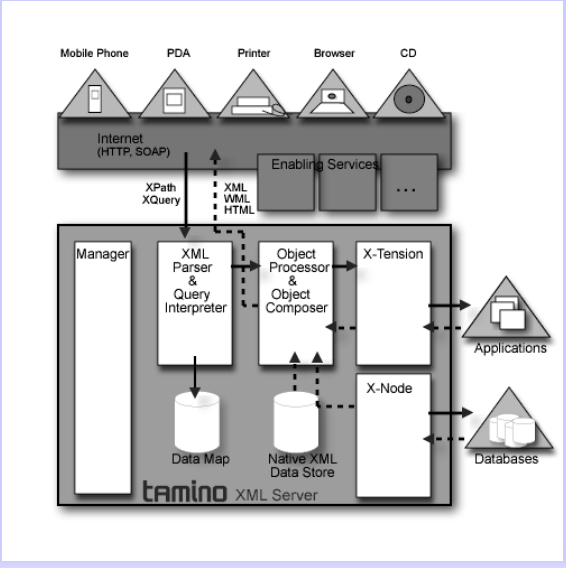
XML

- Dati in singole strutture gerarchiche.
- I nodi hanno elementi e/o attributi.
- Gli elementi possono essere annidati.
- Gli elementi sono ordinati.
- Gli elementi possono essere ricorsivi.
- Lo Schema può essere opzionale.
- Memorizzazione diretta e richiamo di documenti XML.
- Query con XML standard.

RDBMS

- Dati in tabelle multiple.
- Celle con un singolo valore.
- L' ordine delle righe e delle colonne nelle tabelle non sono definite.
- Lo Schema è richiesto.
- Poco supporto per elementi ricorsivi.
- I join sono necessari per richiamare i documenti XML.
- Query con SQL sono adattate in XML.

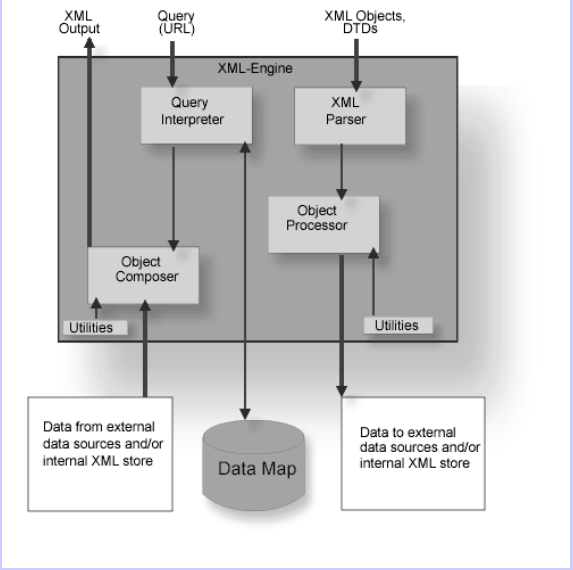
Struttura di Tamino



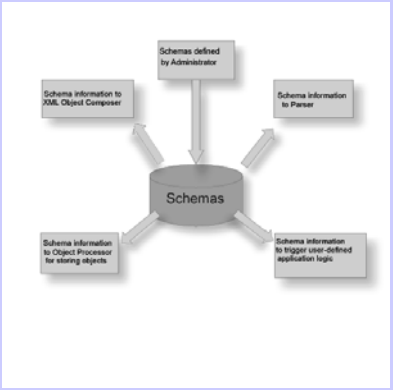
Struttura di Tamino (2)

- **Il Data Store in XML-Nativo**
  - comprendente il motore XML (Parser XML + Object Processor)
  - includono le funzionalità di ricerca per Tamino con linguaggi X-Query, XQuery e full-text
  - Il motore permette a Tamino di memorizzare gli oggetti nativi di XML, e di richiamarli da un deposito nativo di dati
  - ***Object Processor:***
- **L'Object Processor** è utilizzato quando si memorizzano oggetti in XML nativo. Il supporto per sorgenti di dati esterne è fornito dalla Tamino X-Node e Tamino
- ***Interprete Query:***
  - Tamino supporta due linguaggi per le query: Tamino X-Query, basato sullo standard XPath ed il linguaggio XQuery raccomandato dalla W3C.
- **Il Data Map** è la base di "conoscenze" del nucleo del Tamino XML Server
- **L'X-Node** è il componente d'integrazione di Tamino con sistemi esterni di memorizzazione di dati
- **L'X-Tension** permette la chiamata di funzioni definite dall'utente, le cosiddette Server Extensions scritte in C o Java

Interrogazioni con Tamino

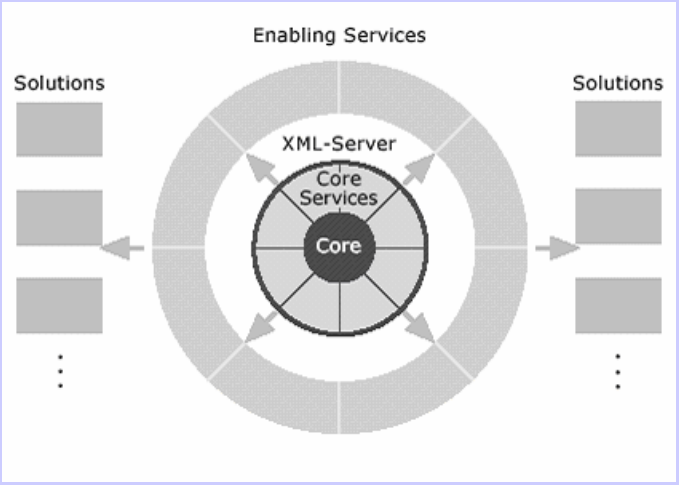


Tamino: Data Map

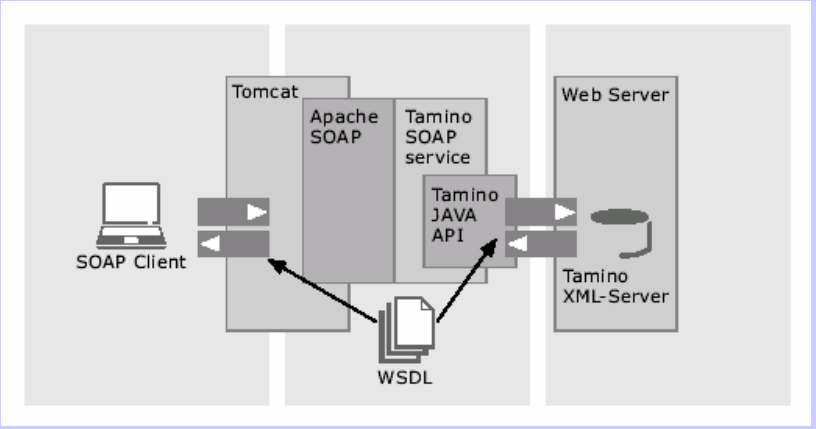


- Creazione DTD:
  - Importazione dello schema e composizione
  - Namespace XML
  - Mapping a risorse di dati esterne
  - Tipi di dati, dati definiti dall'utente.
  - Ridefinizione dell'indice.
  - L'importazione dello schema e la composizione.
  - Mapping ai tipi di dati esterni.
- Amministrazione

Concezione di Tamino



Tamino: Il servizio SOAP



Tamino: Servizi

- **Il servizio di Storage** memorizza XML nel formato originario
- **Il servizio di Query** Questo servizio consiste di tre componenti principali: una provvede a richiamare l'intero testo un'altra é basata su XPath (Tamino X-Query)
- **– Il servizio Full-text retrieval** Servizio per gli ambienti document centric che richiedono mezzi di ricerca basati di più sul contenuto
- **Il servizio XML Schema**
- Il servizio XML Schema supporta la specificazione di XML Schema del W3C
- **Il servizio Security** Per ogni dato memorizzato specificatamente in Tamino XML Server, il Security Manager permette di definire o modificare i diritti di accesso
- **– Il servizio External DB** Questo servizio External DB (esterno alla base di dati, Tamino X-Node) fornisce un accesso adeguato a fonti esterne di dati
- **Il servizio API**
- **– Il Tamino WebDAV Server** Il WebDAV (Web-based Distributed Authoring and Versioning) è uno standard per agevolare la ricerca di fonti di informazioni sul Web

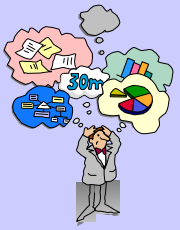
Parte IV: DBMS XML per il Data Mining

Che cosa è il Data Mining

Del termine Data Mining sono state date diverse ed utili definizioni

Il Data Mining (noto anche come Knowledge Discovery in Databases – KDD) è l'insieme di tecniche innovative, sviluppate nel campo della statistica e del "machine learning", utilizzante per analizzare i dati presenti in azienda, impiegando strumenti di esplorazione e modellazione per cercare informazioni utili, nascoste e non evidenti, all'interno di grandi volumi di dati, con un processo iterativo e interattivo e metterle in una forma facilmente comprensibile all'uomo.

Il Data Mining è l' "automatica" estrazione di pattern di informazioni da dati storici, che permettono alle compagnie di focalizzare i più importanti aspetti del loro business. Tali informazioni sono rivelatrici di cose che non si conoscono o ancora più impensabili.



Il termine "Data Mining" è basato sull'analogia delle operazioni dei minatori che "scavano" all'interno delle miniere grandi quantità di materiale di poco valore per trovare l'oro. Nel Data Mining, l'*"oro"* è l'informazione, precedentemente sconosciuta o indiscernibile, il materiale di poco valore sono i dati e le operazioni di scavo sono le tecniche di esplorazione dei dati.

Logica del Data Mining

Nel Data Mining si mettono insieme sia tecniche esplorative sia confermative in una logica ciclica:

- si sceglie cosa studiare,
- si costruisce un modello matematico che tenti di spiegare gli impatti del variare del campione di input sui risultati,
- si verifica la sua robustezza e la sua correttezza, se non è soddisfacente ad una prima analisi si raffina il modello e si procede nuovamente al suo test, e così via fino a quando si ottengono dei risultati soddisfacenti.
- alla fine, quando il modello è sufficientemente accurato, si rende disponibile a tutti gli utenti interessati.

LE FASI

- Nella fase di ***Problem*** (problema di business) si passano in rivista le informazioni di business o gli indicatori chiave che identificano il problema che si vuole conoscere meglio. E' anche la fase di assessment della metodologia

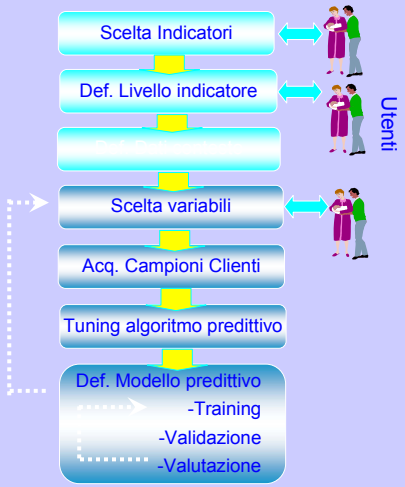
- La fase di ***Model*** assolve al processo di comprensione delle relazioni tra i diversi fattori che influenzano il problema in esame per ottenere delle conoscenze approfondite.

- Infine, la fase di ***Plan*** comprende il processo di crescita della conoscenza ottenuta dall'analisi di fatti storici e dall'utilizzo del modello costruito nella fase di *Model* per formulare le linee guida che deve adottare l'azienda in riferimento al problema studiato.

Capgemini

metodologia di mining

- Scelta degli indicatori
- Definizione del livello dell'indicatore
- Definizione dei dati di contesto rilevanti
- Scelta delle variabili
- Acquisizione campioni casuali di Clienti per training e per validazione del modello (dimensione significativa)
- Scelta dell'algoritmo predittivo (regressione logistica, RBF, rete neurale 'backward propagation')
- Definizione del modello predittivo (processo iterativo con selezione variabili e loro trasformazioni):
  - Training del modello
  - Validazione sui dati DWH
  - Valutazione dei risultati



Capgemini

Tecniche di analisi per la costruzione del modello

Identificato il problema di business e preparati i dati da esplorare bisogna scegliere quale tecnica si vuole applicare per analizzare i dati. In molte situazioni un analista può usare una varietà di tecniche, ma ogni tecnica guida l'analisi in una particolare direzione e rappresenta i risultati in modo differente.

Le varie tecniche di analisi, chiamate anche approcci, modelli o funzioni, in accordo alle categorie di applicazioni in cui possono essere usate, sono raggruppate in sei classi principali come segue:

- Classificazioni,
- Regressioni,
- Serie storiche
- Clustering,
- Associazioni,
- Reti neurali

Data Mining e XML: PMML

- Il PMML (Predicative Model Markup Language) è il linguaggio che permette di descrivere una query di mining.
- Il W3C ha rilasciato il DTD del PMML.
- Strutture di mining riportate dal PMML:

- PMML General Structure
  PMML Conformance
  Header
  Data Dictionary
  Mining Schema
  Statistics
  Normalization
  Tree Classification
  Polynomial Regression
  General Regression
  Association Rules
  Neural Network
  Center-based and Distribution-based Clustering

Oracle e il data mining

- Oracle Mining, che permette:
  - Query su tutti i modelli
  - Possibilità di click-stream
  - Esportazione in PMML
- Oracle 9i
  - 2 modelli per il mining integrati
  - Possibilità di analisi click-streaming per creare modelli comportamentali su WEB
- Disponibili le classi java/C++ per il parsing PMML

Parte V: DBMS XML per i Metadati

Categorie di Metadato

- Categorie che individuano le aree applicative dei metadati:
  - -Quelle che descrivono il dato;
  - -Quelle che descrivono il modo in cui immagazzinare ed usare il dato stesso.
- Categorie di tipo di metadato:
- **Tecnici**: schemi relazionali, trasformazioni, applicativi.
- **Business**: Processi, organizzazione aziendale, KPI.

Metadati: qualità desiderabili

• Essere persistenti;
• Essere dinamicamente estendibili in modo tale da permettere che informazioni addizionali vengano aggiunte durante operazioni real time;
• Essere accessibili e manipolati a diversi livelli;
• Essere trasferibili attraverso diverse applicazioni mantenendo le interfacce esterne;
• Presentare diverse viste a seconda del livello in cui si trovano;
• Contenere metodi estendibili per manipolare e trasformare i dati;

Raccolta metadati: il problema

- I metadati sono raccolti tramite:
- Documenti
- Esportazione dai tools (es. Desiner, Discover, ecc.)
- Siti intranet
- Ecc.

Metadati: la soluzione

•Metadata bridges:

Oggi giorno alcune delle compagnie esistenti decidono di implementare delle proprie interfacce fra i diversi prodotti software utilizzati. Queste interfacce software utilizzate prendono il nome di bridges (ponti). Questi traducono i metadati appartenenti alle varie fonti dati nel formato unico di metadati richiesto dal target database.

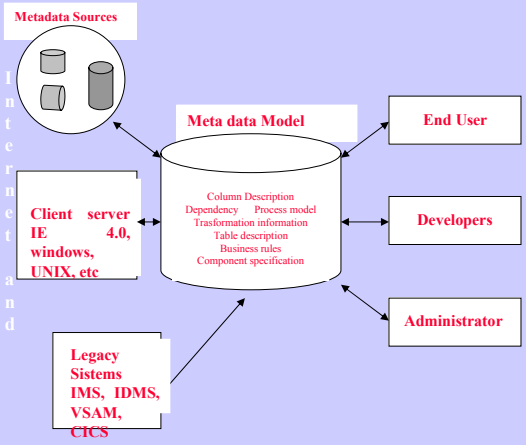
•Metadata repository:

La soluzione basata sul metadata repository che permette di avere un ambiente centralizzato per i metadati, sembra essere la soluzione percorsa dalla maggior parte delle aziende.

•Intelligent software:

Questi Intelligent Software usualmente consistono di un motore e di diverse componenti software in grado di gestire l'accesso ai metadati presenti sui diversi sistemi legacy.

Metadati: Repository



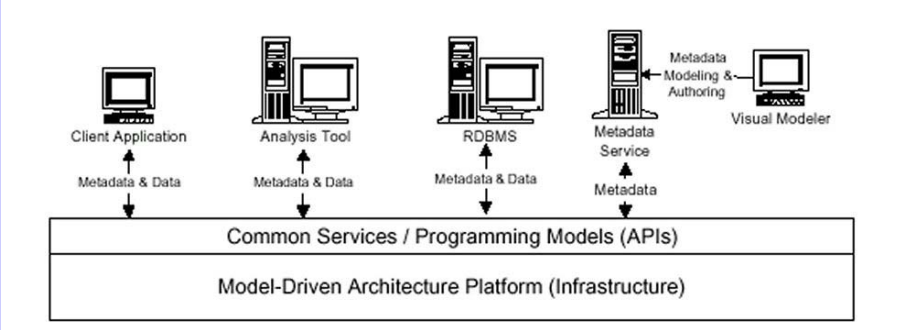
Modello di metadato

- Esigenza di un metamodello per poter rappresentare tutti i metadati.
- Nasce il CWM (Common Warehouse Metamodel) proposto dall'OMG.
- Permette l'interscambio dei metadati.
- Offre un **DTD** generalizzato a cui tutte le aziende possono aderire.

Model Driver Architecture

- Per cercare di integrare i vari modelli di metadati aziendali nasce il **Model Driver Architecture**.
- MDA pone al centro UML, MOF (*Meta Object Facility*) e CWM (*Common Warehouse Meta-model*).
- L'idea consiste nel definire un (meta)modello (*core model* denominato nella terminologia *UML*) indipendente da ogni piattaforma, una collezione di (meta)modelli specifici per ciascuna divisione.

MDA: Architettura



MDA: Architettura (2)

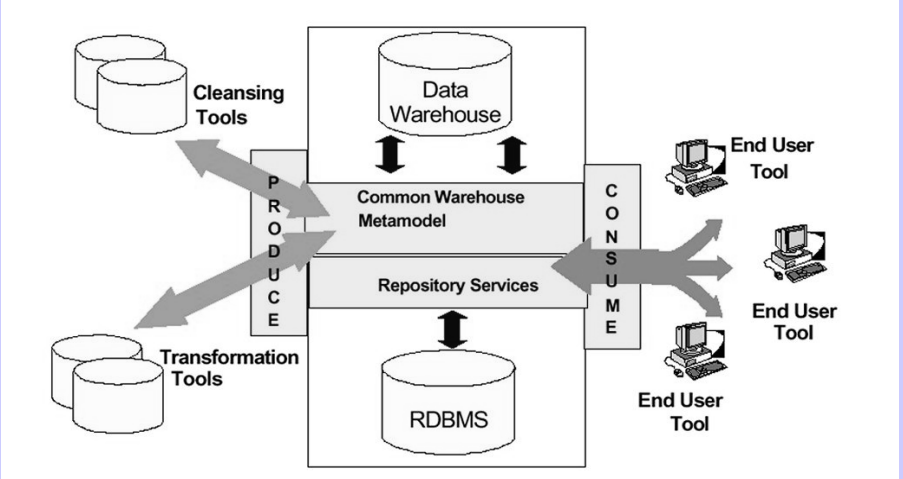
***MOF(Meta Object Facility)*:** Linguaggio formale (sintassi e semantica) per la rappresentazione dei metadati.

***XMI(XML Metadata Interchange)*:** Formato d'interscambio per lo scambio e pubblicazione dei metadati.

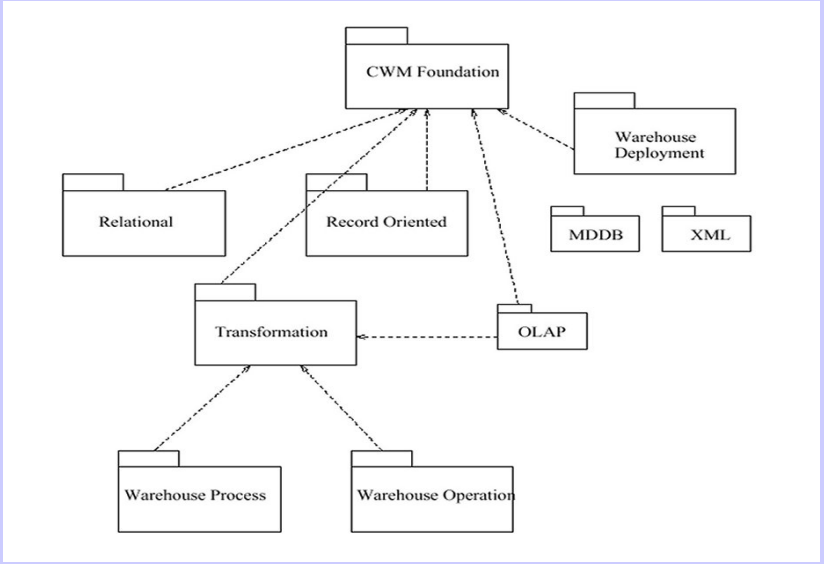
***CWM(Common Warehouse Metamodel)*:** Modello che permette l'accesso e la localizzazione dei metadati.

***UML(Unified Modeling Language)*:** Linguaggio standard per la definizione di metamodelli.

La soluzione CWM



Struttura del CWM



CWM Foundaction

CWM Foundation : è una collezione di package di metamodelli contenenti elementi che rappresentano concetti e strutture condivisi dagli altri package CWM. Il CWM Foundation copre le seguenti aree:

- **Business Information** definisce le informazioni orientate al business;
- **Data Types e CWM Types** contiene la definizione dei costrutti dei metamodelli utilizzati;
- **Expressions** definiscono metodi per la registrazione di espressioni condivise dagli altri package CWM in una forma comune che può essere usata nello scambio d'informazioni;
- **Keys e Indexes** definiscono chiavi ed indici;

CWM Package

•***Warehouse Deployment*** contiene elementi per registrare le modalità con le quali le componenti software ed hardware sono utilizzate all'interno del data warehouse;
•***Relational*** descrive l'accesso ai dati attraverso interfacce relazionali. Questo package segue lo standard SQL:1999;
•***MDDB (Multidimensional Database)*** è una generica rappresentazione dei database multidimensionali;
•***Record Oriented*** descrive i concetti base di un record e della sua struttura;
•***XML*** contiene tipi ed associazioni che descrivono le fonti dati xml;
•***Trasformation*** contiene le trasformazioni tra diversi tipi di fonti dati: object oriented, relazionali, record oriented, multidimensionali, XML e OLAP;
•***OLAP*** definisce un metamodello che descrive i costrutti OLAP essenziali che vengono usati tra diverse applicazioni e tool OLAP
•***Warehouse Process*** documenta il processo di flusso usato per eseguire le trasformazioni;
•***Warehouse Operation*** contiene informazioni riguardanti le operazioni giornaliere effettuate sul data warehouse;

Riassumendo: Lo Standard per l'interscambio di metamodelli è XMI

- Rispetta il DTD del CWM
- Rappresenta modelli UML
- Utile per intercambio di progetti documentati con UML

Parte VI: Parser XML Tecnologia Oracle

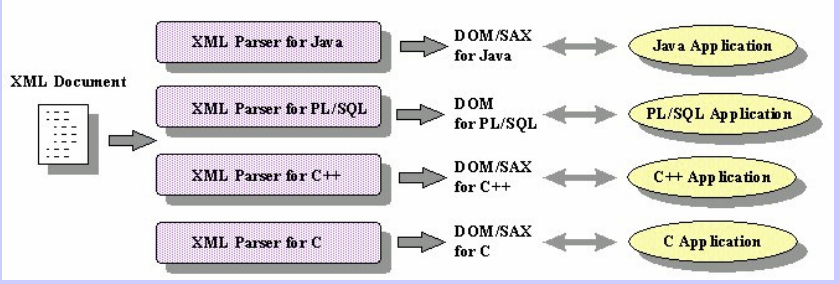
Oracle XML Developer's Kit

- Oracle XDK e' disponibile per Java, C, C++, e PL/SQL
- Disponibile su OTN (con supporto Oracle)
- I componenti del XDK sono:
  - XML Parsers
  - XSL Processor
  - XML Schema Processor
  - XML Class Generator
  - XML Transviewer Java Beans
  - XSQL Servlet

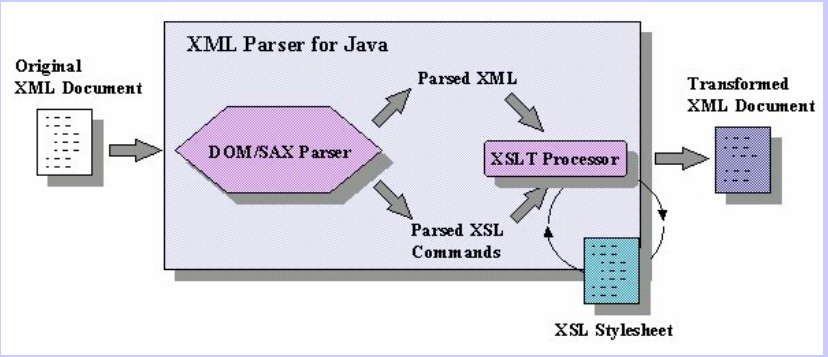
XML Parsers

- XML Parsers in Java, C, C++ e PL/SQL
- Interfacce DOM 2.0 e SAX 2.0
- Supporto integrato per XSLT
- Parser Validante e Non Validante
- Supporto per Namespace
- New high performance architecture
- Supporto per le codifiche: UTF-8, UTF-16, ISO-10646-UCS-2, ISO-10646-UCS-4, US-ASCII, EBCDIC, ISO-8859-*, Shift_JIS

XML Parser



XSL Trasformation Support



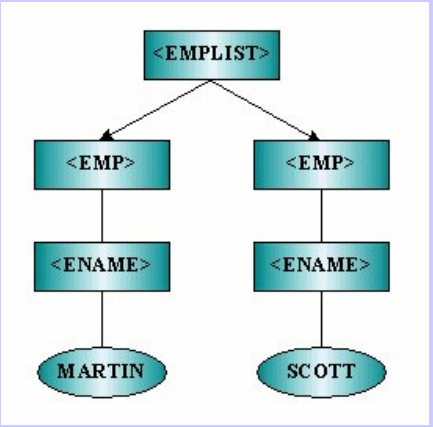
Un occhiata al "SAX"

```
<?xml version="1.0"?>
 <EMPLIST>
  <EMP>

<ENAME>MARTIN</ENAME>
  </EMP>
  <EMP>
   <ENAME>SCOTT</ENAME>
  </EMP>
 </EMPLIST>
```

diventa

```
start document
start element: EMPLIST
start element: EMP
start element: ENAME
characters: MARTIN
end element: EMP
start element: EMP
start element: ENAME
characters: SCOTT
end element: EMP
end element: EMPLIST
end document
```

Un occhiata al DOM



XML e Java 1/2

```
import org.w3c.dom.*;
import oracle.xml.parser.v2.*;

…

// Parse XSL and XML documents
        parser = new DOMParser();
        parser.setPreserveWhitespace(true);

        xsldoc = (parser.parse(xslURL)).getDocument();
        xml = (parser.parse(xmlURL)).getDocument();

        // Instantiate the stylesheet
        XSLStylesheet xsl = new XSLStylesheet(xsldoc, xslURL);
….
```

XML e Java 2/2

```
XSLProcessor processor = new XSLProcessor();

// Process XSL
DocumentFragment result = processor.processXSL(xsl, xml);

// Create an output document to hold the result
out = new XMLDocument();

// Create a dummy document element for the output document
Element root = out.createElement("root");
out.appendChild(root);

// Append the transformed tree to the dummy document element
root.appendChild(result);

// Print the transformed document
out.print(System.out);
```

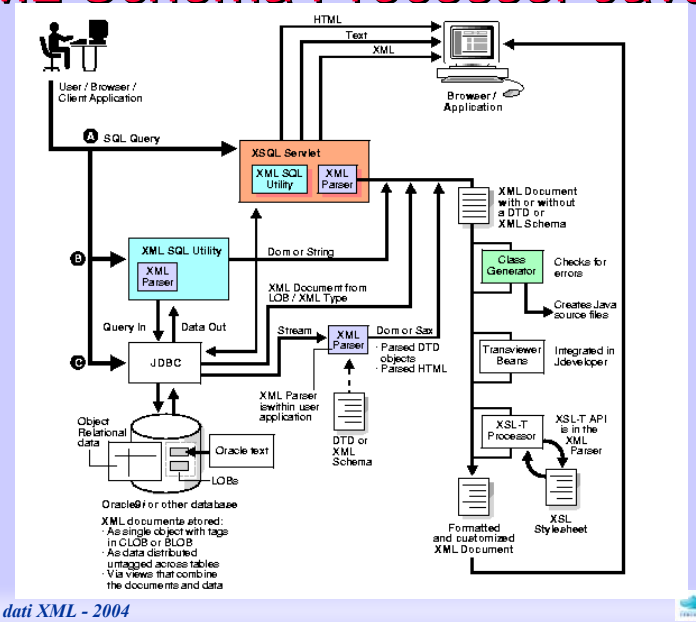

XML e PL/SQL

```
begin
-- new parser
  p := xmlparser.newParser;

-- set some characteristics
  xmlparser.setValidationMode(p, FALSE);
  xmlparser.setErrorLog(p, dir || '/' || errfile);
  xmlparser.setBaseDir(p, dir);

-- parse input file
  xmlparser.parse(p, dir || '/' || inpfile);

-- get document
  doc := xmlparser.getDocument(p);
```

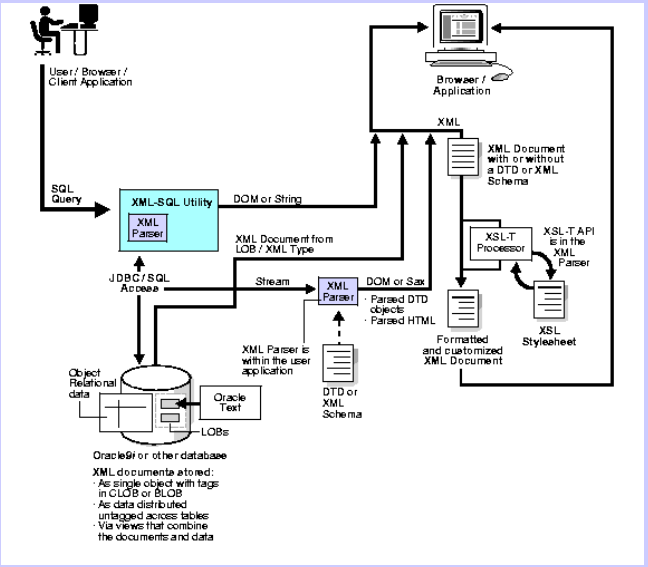
XML Schema Processor

- XML Schema Processor disponibile per Java
- Supporta le attuali specifiche W3C
- Supporta sia I tipi semplici che I tipi complessi
- C, C++ e PL/SQL

XML Schema Processor Java

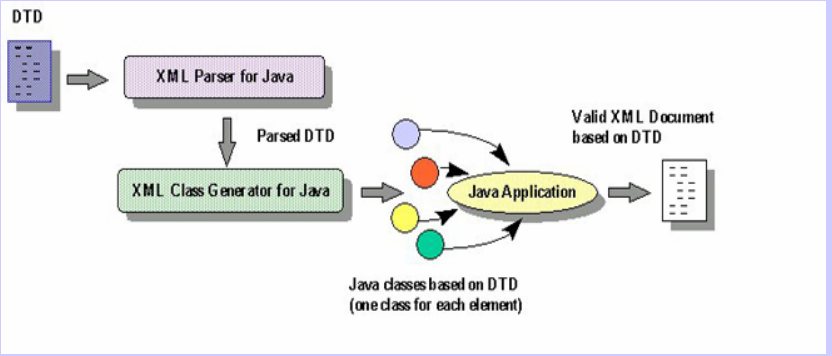


XML Schema Processor PL/SQL



XML Class Generator for Java

Generates Java Classes from XML Documents

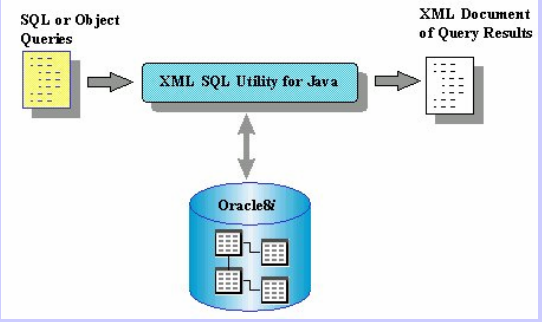


Server-Side XML Components

- XML SQL Utility
  - Genera documenti XML da query SQL
  - Restituisce text, DOM, o ResultSet Object
  - Genera il DTDs dallo schema
- XSQL Servlet
  - Produce dinamicamente documenti XML utilizzando una o piu' querySQL.
  - Trasforma il documento XML utilissando XSLT sul server

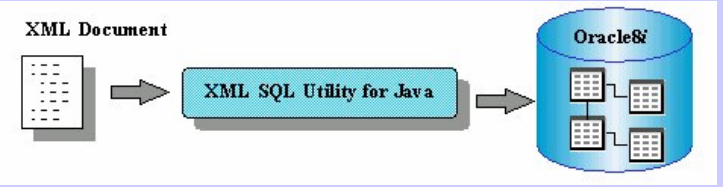
XML SQL Utility

- Passa una query al Database generando un documento XML
- Scrive dati XML in una tabella



XML SQL Utility

- Utilizzo XML per scrivere in una tabella



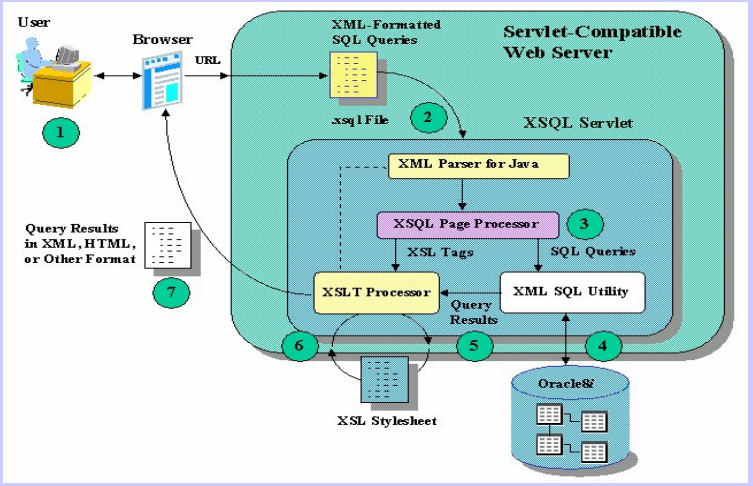
XML SQL Utility: esempio

- SELECT EMPNO, ENAME FROM EMP WHERE EMPNO = 7654;

  genera

```
<?xml version="1.0"?>
<ROWSET>
    <ROW id="1">
        <EMPNO>7654</EMPNO>
        <ENAME>MARTIN</ENAME>
    </ROW>
</ROWSET>
```

XSQL Servlet



XSQL Servlet

- Genera documenti XML da query XML

```
<?xml version="1.0"?>
 <query connection="demo">
 SELECT 'Hello World' AS "GREETING" FROM DUAL
 </query>
```

```
<?xml  version = '1.0'?>
 <ROWSET>
  <ROW id="1">
    <GREETING>Hello World</GREETING>
  </ROW>
 </ROWSET>
```

- Si basa su XML SQL Utility

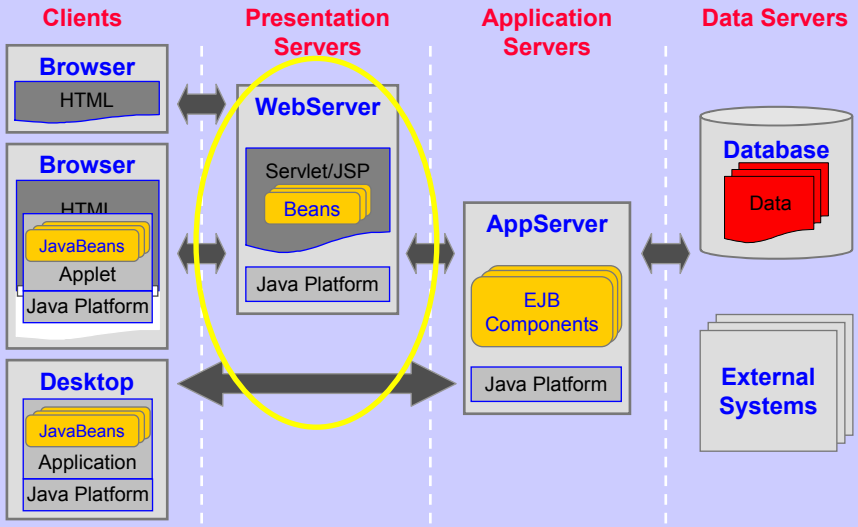
Java + XML = Oracle JDeveloper

- Build Transactional B2B Applications Quickly
  - XML, HTML, Servlet/JSP, Business Components
  - Write Once, Deploy Everywhere
    - Browser, WAP phone, PDAs (Palm Pilot), etc.

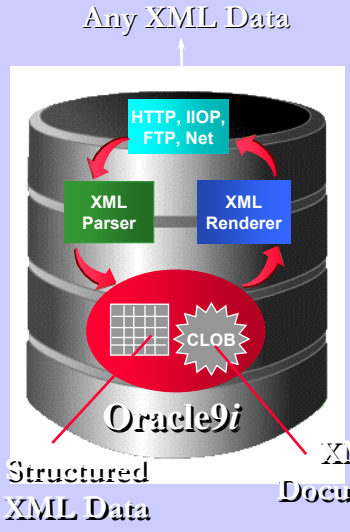
XML in Oracle JDeveloper

- BC4J Framework utilizza XML
- Supporta XSQL Servlet
- XDK Transviewer Beans
- Supporta XML
- XML Web Bean

Architettura Internet Standard



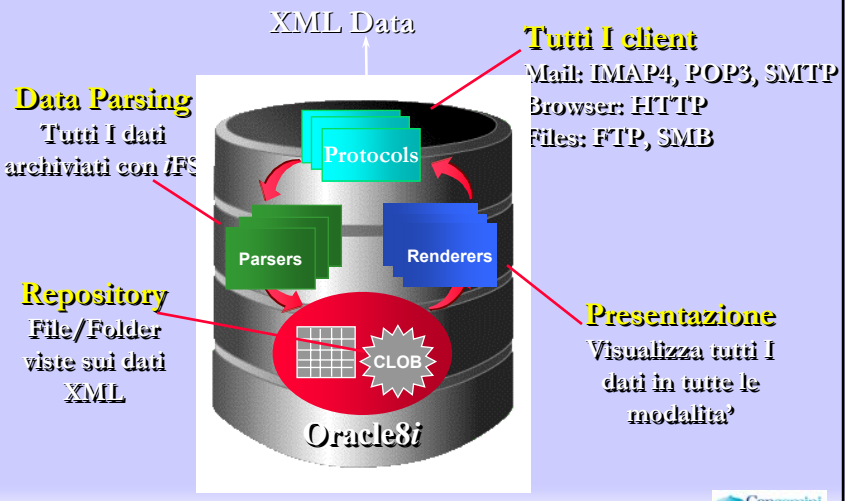
Oracle9*i* - The XML-Enabled Database...



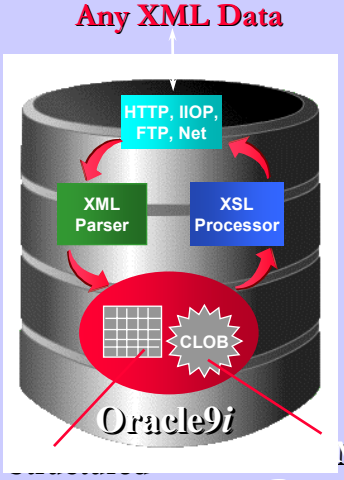
- XDK Integrati
- Archivia documenti XML come dati o come documenti
- Query XML
- Ricerca di tags XML con *inter*Media Text
- *i*FS semplifica la gestione degli schemi

Oracle Internet File System (*i*FS)

Integrato Database "File" System



*i*FS: Store and Manage XML



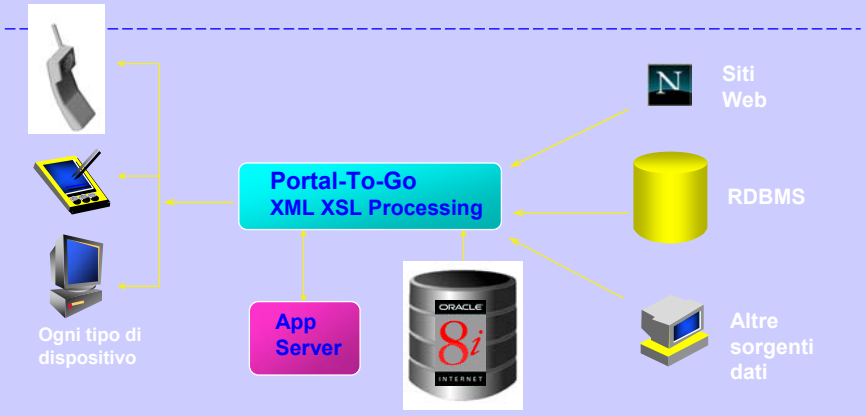
Servizi Oracle *i*FS

- Parsing e archiviazione di documenti XML
- Mappa dati XML in colonne di tabelle
- Permette query SQL e manipolazione di dati XML
- Archivia documenti XML come dati o come documenti

*Ricerche con inter*Media Text

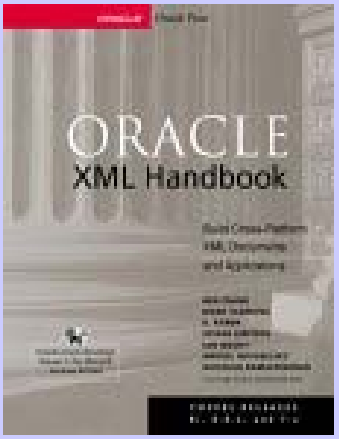
- Supporto nativo XML
- Ricerche gerarchiche
- Ricerche sugli attributi e sugli elementi

Oracle 9i AS - Portal-To-Go



Risorse XML Oracle

Oracle Technet: http://otn.oracle.com/tech/xml



Riferimenti e proposte

- E-mail: sergio.iacobelli@capgemini.com
- Approfondimenti su “Basi di dati XML in ambito Datawarehouse” con utilizzo sia di un motore dimensionale che di un motore XML gerarchico.

CONCLUSIONI