Eiha?!? : XML per l'organizzazione di URL con distribuzione multimodale

Jean-Christophe Pazzaglia, Chiara Biancheri, Ivan Marcialis
CRS4 Centro di Ricerca, Sviluppo e Studi Superiori in Sardegna
{jc,chiara,ciano}@crs4.it

Pisa, 9/10 Maggio 2000

Abstract

La crescita esponenziale di risorse presenti in rete ed il largo sviluppo di strumenti per l'accesso multimodale ad internet fanno sorgere nuovi problemi di gestione dell'informazione quali la ricerca di contenuti e la formattazione dei contenitori. In questo contesto ed all'interno del progetto europeo Vision abbiamo costruito un manuale telematico interattivo della cultura e del territorio della Sardegna. Questa raccolta di dati è stata progettata anche come piattaforma di sperimentazione di nuove tecnologie quali l'XML e le sue applicazioni. Il risultato ottenuto è un servizio multimodale, denominato Eiha?!?, fruibile attraverso PC, cellulari e PDA.

Keywords Metadata, DTD, XML, XSL, WML, WAP, DBMS, Search Engine.

Introduzione

L'analisi dello sviluppo del fenomeno Internet , divampato attraverso la costruzione di siti Web e pagine personali per ogni tipo di attività, e della ampia utenza, che si è avvicinata al mezzo della comunicazione globale in questi ultimi anni con nuove e diverse esigenze ed aspettative, ha evidenziato l'importanza fondamentale di due strumenti distinti e complementari quali i motori di ricerca e la multimodalità. In questo contesto il progetto Vision, finanziato dal Fondo Sociale Europeo nell'ambito del programma ADAPT - Bis (Building the Information Society) e realizzato dal CRS4, ha come obiettivo principale la costruzione di un manuale telematico interattivo della cultura e del territorio della Sardegna, consultabile in rete grazie ad un sito Internet dedicato.

La prima fase di questo progetto risiede nella raccolta dei dati già esistenti riguardanti la Sardegna sotto ogni aspetto. I siti raccolti necessitano una formattazione per la classificazione, la consultazione e la ricerca. Questi requisiti possono essere ottenuti tramite l'aggiunta di meta-dati capaci di descrivere in modo completo vari aspetti di ogni risorsa. La tecnologia XML offre quindi un utile strumento per la nostra catalogazione dei siti e per l'elaborazione dei contenuti.

L'articolo è organizzato come segue: nella sezione 1 vengono descritte le diverse funzionalità implementate del sistema; nella sezione 2 diamo una panoramica dell'architettura generale; nella sezione 3 ci occupiamo della descrizione dei tipi di dati attraverso un'analisi delle DTD utilizzate dal sistema; nella sezione 4 viene presentato il meccanismo di query alla base di dati; nella sezione 5 introduciamo brevemente il linguaggio WML ed il suo utilizzo all'interno del nostro sistema, ed infine nella sezione 6 descriviamo come è stata implementata la parte relativa alla navigazione geografica delle risorse.

1 Descrizione funzionalità EIHA?!?

Obiettivo principale di EIHA?!? è lo sviluppo di un'applicazione in ambiente XML per l'indicizzazione, la classificazione e l'integrazione di siti Web sulla Sardegna. È stata quindi effettuata una raccolta manuale e catalogazione dei siti preesistenti sulla Sardegna utilizzando lo schema di documento definito precedentemente: questo compito di ricerca ragionata é stato affidato ad un esperto culturale e storico sardo. Questa raccolta, basata su una DTD e contenente meta-dati sulle risorse archiviate, é fruibile sotto due diversi aspetti complementari: come insieme di dati ordinati ad "indice" nel quale poter navigare e come catalogo di siti per ricerche "intelligenti". Questo lavoro, per il quale sono stati necessari diversi mesi di lavoro, ha permesso la catalogazione di più di 700 siti che descrivono i diversi aspetti della cultura sarda.

A livello di funzionalità dell'interfaccia utente abbiamo deciso di privilegiare diversi tipi di consultazione attraverso molteplici canali. La nostra classificazione permette tre tipi di consultazione:

- navigazione nell'indice gerarchico (modello Yahoo!),
- motore di ricerca contestuale sfruttando la descrizione data del nostro esperto (ad esempio Gavino come Luoghi: "San Gavino" o come Persona: "Gavino Paddeu"),
- navigazione su una carta geografica della Sardegna.

Se queste caratteristiche sono oramai disponibili attraverso la maggior parte dei più comuni portali, pochi di questi offrono la stessa interfaccia per differenti dispositivi di accesso. La consultazione del sito deve quindi essere effettuabile a partire da un insieme di browser diversi sia HTML (vedi figura 9) che XML [2] ed in particolare utilizzando il dialetto dedicato alla nuova generazione di telefonini. Per conseguire tale obiettivo abbiamo scelto di sviluppare una architettura basata sulle ultime tecnologie in ambiente Internet.

2 Architettura del sistema

L'architettura generale del sistema EIHA?!? è presentata nella figura 1. In questa architettura il linguaggio XML, in modo conforme alle DTD descritte nella sezione successiva, è usato come linguaggio di *middleware* per le estrarre le informazioni dalla base di dati. I fogli di stile XSL [3, 7] sono utilizzati per adattare la presentazione delle informazioni al dispositivo finale di destinazione. Durante questa operazione, le capacità di trasformazione del foglio di stile XSL sono principalmente utilizzate per filtrare e riorganizzare queste informazioni secondo le caratteristiche del dispositivo ed i differenti linguaggi ipertestuali. Nell'attuale versione del nostro sistema l'effettiva presentazione è descritta da un foglio di stile CSS compatibile sia per *browser* HTML che per file WML per quanto riguarda la navigazione attraverso un terminale WAP (per la maggior parte dei casi un telefono cellulare).

L'implementazione dell'architettura è basata sullo standard corrente per lo sviluppo di Web dinamici in Java. Una servlet specializzata viene eseguita sul server Tomcat [11]. I ruoli della servlet sono molteplici:

- riconoscere automaticamente il dispositivo dell'utilizzatore ed restituire sempre automaticamente il linguaggio di markup appropriato (generalmente HTML e WML),
- essere responsabile della memorizzazione delle informazioni dipendenti dalla sessione ed amministrare una cache per l'elaborazione di dati che hanno bisogno di molta CPU (per esempio l'elaborazione della mappa in formato SVG),
- amministrare la connessione alla base di dati relazionale grazie alla connessione JDBC ed un form basato su un template XML,

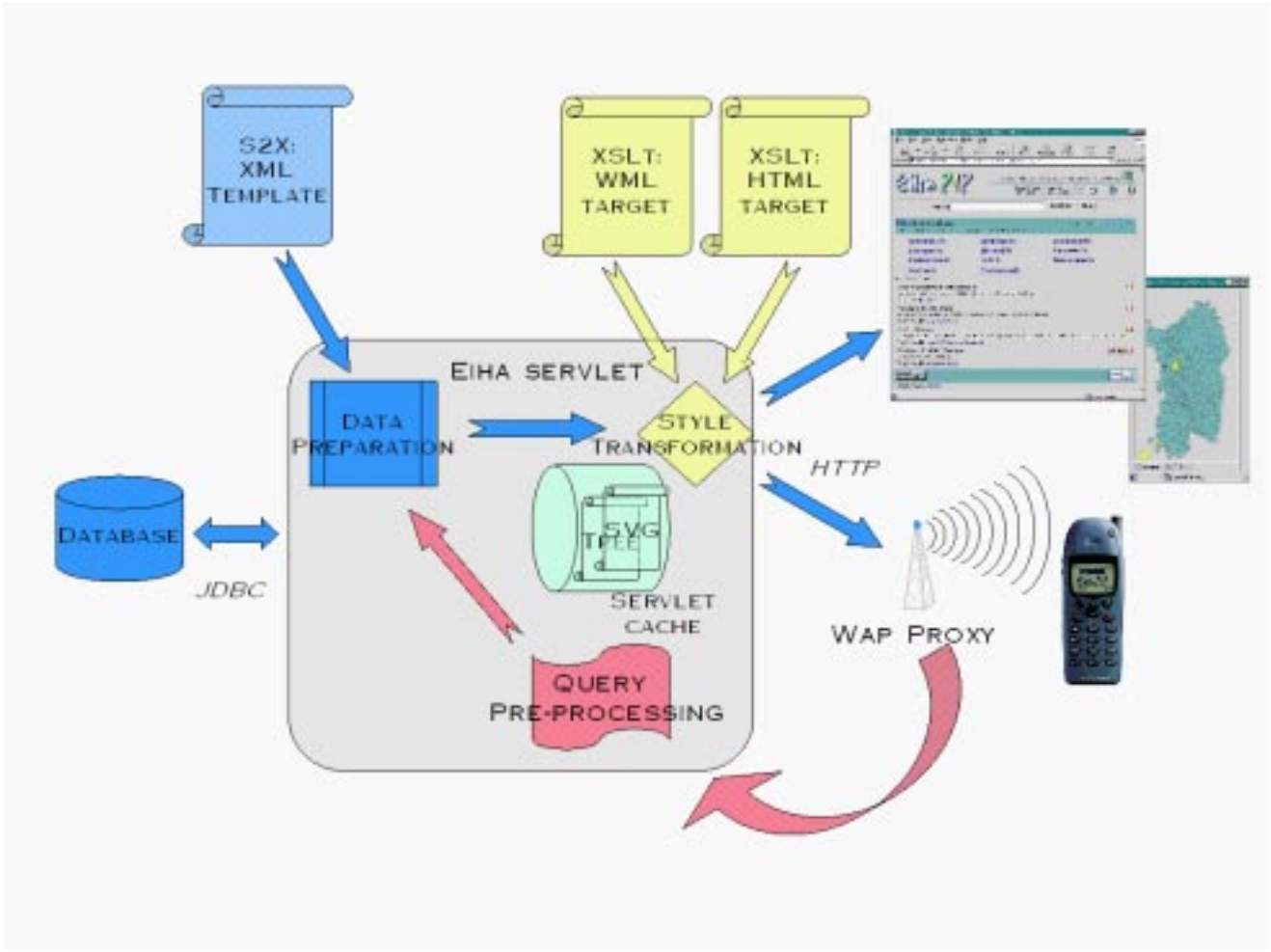

Figure 1: Architettura Eiha?!?

- elaborare la traduzione fra la il form HTML dell'advanced query e la base di dati relazionale.

Vogliamo ora descrivere in modo più approfondito le DTD del sistema, in seguito il meccanismo, basato su *template*, utilizzato per effettuare le *query* alla base di dati ed infine daremo una breve descrizione del linguaggio WML.

3 Modellizzazione dei dati

La prima attività importante per lo sviluppo della nostra applicazione XML è stata la definizione dei Document Type Definition (DTD) relativi ai tipi di documento utilizzati dal nostro sistema. L'analisi e la scrittura di DTD ci hanno permesso di identificare e descrivere i diversi componenti e le loro relazioni.

Possiamo identificare una selezione di DTD associate ai tipi di documento necessari alla descrizione di un indice gerarchico:

- una DTD per la descrizione della gerarchia delle categorie in cui i siti sono catalogati (figura 2: DTD `categorie`),
- una DTD per la descrizione della singola risorsa, contenente, oltre all'URL, una serie di informazioni supplementari quali l'autore, la lingua, le parole chiave ed in particolare un giudizio sul contenuto ed un'informazione geografica (figura 3: DTD `risorsa`),
- una DTD per la descrizione delle località, che viene utilizzata dal nostro sistema georeferenziato (figura 4: DTD `map`).

Queste sopra descritte sono le 3 DTD principali e sono costruite a partire da un insieme di DTD atomiche che vogliono assumere a libreria di tipi di dato astratto come per esempio il tipo `date` e il tipo `url`.

La DTD `categorie` descrive in modo standard la struttura di un albero. Ogni nodo dell'albero contiene un insieme di risorse, l'insieme delle categorie correlate al nodo stesso ed un insieme di sottoalberi. La differenza principale esistente fra il sottoalbero e la categorie correlata risiede nella struttura ricorsiva del primo e nell'essere solo un riferimento ad un nodo la seconda.

La DTD `risorsa` permette la descrizione di siti Web in modo similare al Dublin Core (DC). Come nel DC, per descrivere la risorsa Web si utilizza un insieme di dati (titolo, abstract, parole chiave, ...). Abbiamo esteso questa descrizione in diverse direzioni. Per completare la figura della persona incaricata dei servizi Web, il webmaster, abbiamo introdotto l'elemento autore come responsabile dei contenuti. Per la classificazione di siti in diverse lingue, abbiamo incluso in ogni risorsa un insieme di URL. Per effettuare una valutazione dei siti, oltre al un giudizio soggettivo ed arbitrario, abbiamo inserito una sorta di sommario capace di raccogliere le quantità di file presenti nelle risorse associate ai loro *mime type*. Infine ogni risorsa contiene un riferimento alla sua posizione geografica che ricopre l'area descritta dalla risorsa stessa.

La DTD map descrive una classica carta geografica. Si tratta di una descrizione gerarchica basata sulla suddivisione amministrativa del territorio (stato, regione, provincia, comune). Ogni area contiene un insieme di polilinee costruite con punti georeferenziati (latitudine e longitudine). Un'area può anche contenere un insieme di posti di particolare interesse (montagne, laghi, città, ...). Tutti gli elementi possono contenere una piccola descrizione ed il riferimento ad una URL.

4 Query alla base di dati *template driven*

Le principali ragioni per la scelta di una base di dati relazionale invece dell'XML nativo sono due. In primo luogo le performance di tale sistema ci permettono la memorizzazione di grandi quantità di

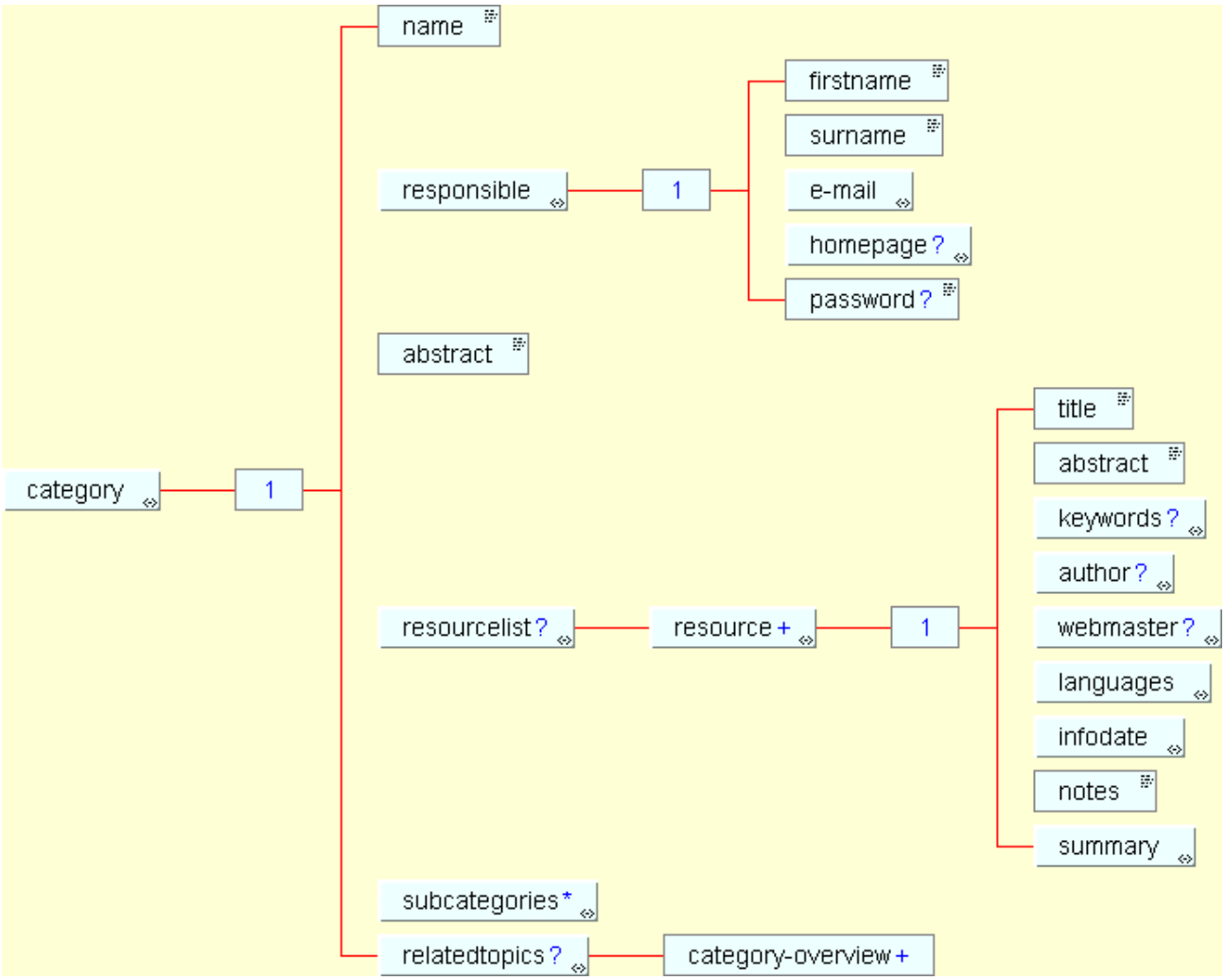

Figure 2: DTD `categorie`

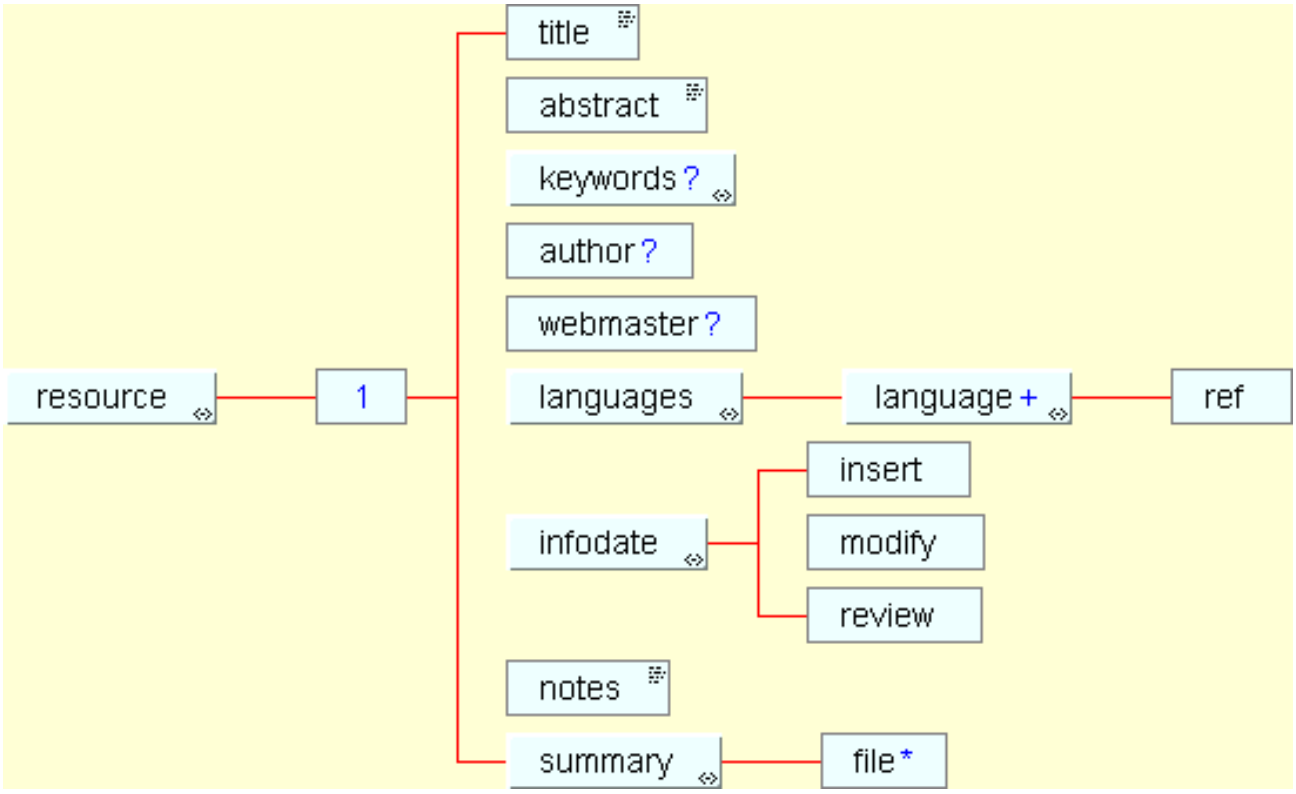

Figure 3: DTD `risorsa`

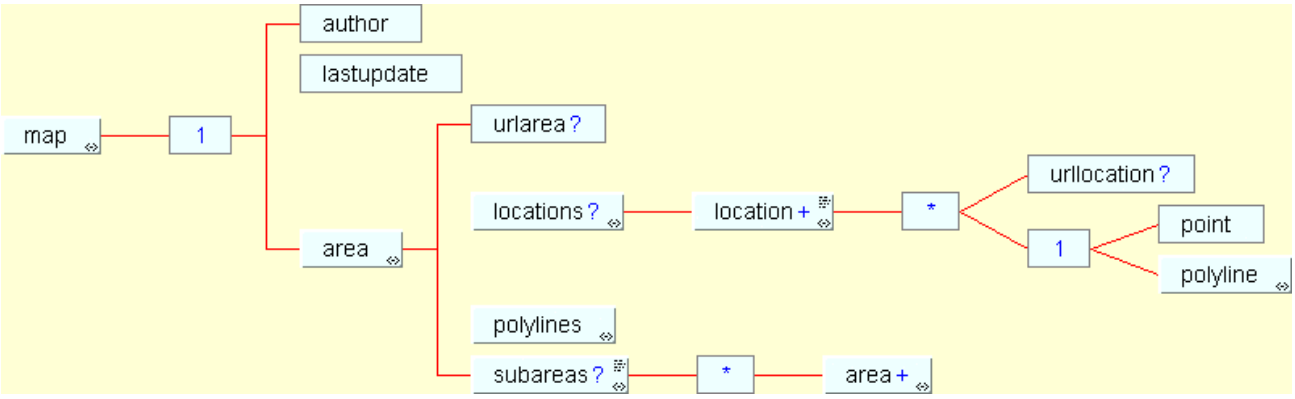

Figure 4: DTD `map`

informazioni. In secondo luogo applicazioni complementari, che possono essere eseguite sia in batch che in *background*, potrebbero agire sullo stesso insieme di dati senza problemi di gestione della consistenza e della coerenza. Nel nostro contesto di fruizione di informazione multimodale, siamo convinti che tecnologia XML porti dei notevoli vantaggi: l'importanza dell'ambiente XML è particolarmente convincente per quanto riguarda la trasformazione e l'ordinamento delle informazioni.

Per effettuare l'estrazione delle informazioni da una base di dati in XML, si possono considerare due tipi di soluzioni [6]:

- un mapping *model driven*: la struttura dell'XML generato è strettamente collegata al soggiacente modello memorizzato (nel nostro caso tabelle relazionali),
- un mapping *template driven*: i dati sono inseriti in maniera integrata nel file XML e la struttura soggiacente è totalmente nascosta.

Entrambe le soluzione possono essere implementate native nella nostra servlet grazie ad un programma Java ad-hoc o grazie ad uno schema più sofisticato che utilizza un file di configurazione. Riteniamo che una soluzione basata su *template* che usi un file di configurazione XML sia quella che meglio si adatta ai nostri bisogni. La PXSLServlet [12] appare un buon candidato avente entrambi i requisiti.

```
- <S2X source="CATEGORY" fields="CATEGORY_ID,CATEGORY_NAME,ABSTRACT,RESPONSIBLE_ID" condition="WHERE CATEGORY_ID=$/s2x.catid">
 - <S2X source="$../" fields="CATEGORY_ID,CATEGORY_NAME,ABSTRACT,RESPONSIBLE_ID">
  - <category>
    - <fathercategory>
     + <S2X source="SUBCATEGORY_NAME_VIEW" fields="CAT_ID,CAT_NAME,SCAT_NAME,SCAT_ID,NBRES" condition="WHERE SCAT_ID=$/s2x.catid">
      </fathercategory>
      <name>$CATEGORY_NAME</name>
     + <S2X source="PERSON" fields="PERSON_ID,FIRSTNAME,SURNAME,E_MAIL_USERNAME,E_MAIL_DOMAIN" condition="WHERE
       PERSON_ID=$../RESPONSIBLE_ID">
      <abstract>$ABSTRACT</abstract>
     + <resourcelist>
     - <subcategories>
       - <S2X source="SUBCATEGORY_NAME_VIEW" fields="SCAT_NAME,SCAT_ID,NBRES" condition="WHERE CAT_ID=$/s2x.catid ORDER BY
         SCAT_NAME">
         - <S2X source="$../" fields="SCAT_NAME,SCAT_ID,NBRES">
           - <category-overview>
              <id-category>$SCAT_ID</id-category>
              <name>$SCAT_NAME</name>
              <number>$NBRES</number>
             </category-overview>
           </S2X>
         </S2X>
      </subcategories>
     + <relatedtopics>
      </category>
   </S2X>
</S2X>
```

Figure 5: Un file S2X

Il maggior vantaggio della PXSLServlet è l'utilizzo di un *mapping* in XML basato su template. Uno tipo speciale di nodi, i tag `S2X`, permette la specifica di *query* alla base di dati insieme alla definizione di variabili con *lexical scope*, accessibile con una sintassi semplice che consente l'utilizzo del *dataset result* nel *template* XML. Questo formato, come mostrato in figura 5 permette l'inserzione dei dati nel *template* in maniera dichiarativa ed abbastanza intuitiva senza dover conoscere le specificità della libreria JDBC. Infine, per poter caratterizzare le *query*, abbiamo aggiunto la possibilità di includere variabili derivanti dalle URL (questa possibilità è stata aggiunta nella release 0.3).

5 Wap e WML

Il *wireless*, dai cellulari alle Pda, a causa delle limitazioni legate alle batterie ed alle dimensioni, tende ad avere CPU poco potenti, poca memoria, display piccoli e dispositivi di input semplificati. Allo stesso tempo, le reti mobili tendono ad avere molte limitazioni rispetto alla reti tradizionali: meno

banda, maggiori tempi di attesa, minore stabilità e disponibilità.

Per rispondere a tali problematiche, un consorzio industriale, che raggruppa i maggiori attori della comunicazione *wireless*, ha proposto la piattaforma Wap [9]. Il Wap, in parte utilizzando le tecnologie esistenti, ed in parte introducendone di nuove, fornisce un ambiente in grado di superare le limitazioni delle reti mobili e dei terminali consentendo la distribuzione di servizi utilizzando un telefonino opportunamente equipaggiato od un Pda.

L'architettura Wap, come mostrato in figura 6, può essere vista come un'estensione dell'ambiente Web dedicata ai dispositivi mobili.

Per permettere l'accesso ai servizi attraverso dispositivi mobili, un server è dedicato alla traduzione dei servizi HTTP verso clienti Wap. Questo server si chiama *wap gateway* ed è responsabile della traduzione di un linguaggio di *markup* (quale HTML) in una versione codificata del documento. C'è un duplice guadagno nell'utilizzo della versione codificata: si riduce in modo considerabile la quantità di byte necessari al trasferimento dei dati e si ha la certezza di avere documenti senza errori. Questi due vantaggi permettono lo sviluppo di *browser* leggeri, i così detti *micro-browser*, basati su un protocollo con banda ristretta quale GSM.

Ovviamente i servizi standard costruiti per il Web saranno penalizzati da uno schema di traduzione automatica ed è per questo motivo che i servizi vengono sviluppati tenendo conto delle limitazioni del dispositivo attraverso un linguaggio di *markup* nativo del Wap: il *wireless markup language* (WML).

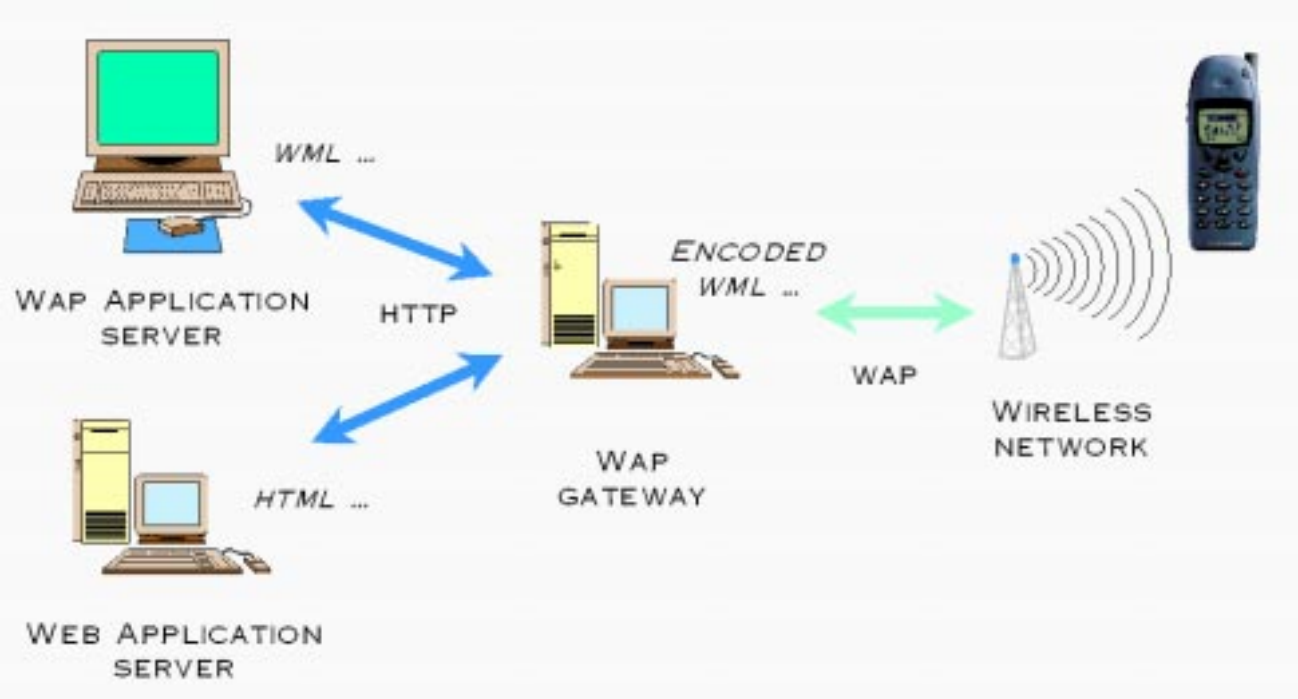

Figure 6: Architettura Wap

5.1 I linguaggi per il WAP

Vogliamo qui brevemente introdurre i linguaggi Wap e per fare ciò utilizzeremo un parallelo con i linguaggi per il Web. L'alter ego per il Wap rispetto all'HTML è chiamato Wireless Markup Language [10]. WML é un'applicazione XML sviluppata per distribuire contenuti e servizi su telefoni cellulari e più in generale su apparecchi che hanno le seguenti caratteristiche: display di dimensioni limitate

e con bassa risoluzione, limitate capacità di input, limitate capacità di elaborazione ed una banda ristretta.

Poiché WML è stato progettato per essere un linguaggio di ipertesto, molti concetti base sono comuni all'HTML (meccanismo delle URL, *Anchor* ...). Comunque è importante sottolineare che sebbene le potenzialità del WML siano un sottoinsieme di quelle dell'HTML, WML non è un sottoinsieme dell'HTML.

Una delle differenze più importanti esistenti fra questi linguaggi è l'unità di trasferimento di base. Si può considerare un'equivalenza fra il contenuto dell'HTML (il *body*) e la pagina visualizzata, WML presenta la nozione di *deck* come unità di trasferimento mentre l'elemento base per la visualizzazione è la *card*. Un *deck* contiene un insieme di *card* identificate dal rispettivo identificatore (id) ed un *template* condiviso. Il *template* permette la condivisione di una struttura comune per l'insieme delle *card* disponibili in quel *deck*. Infine la presenza di diverse *card* in un unico *deck* consente all'utilizzatore la navigazione fra le *card*senza dover effettuare una nuova connessione alla rete.

E' anche disponibile un linguaggio di script, chiamato WMLScript: utilizza la stessa sintassi di JavaScript ma offre soltanto un sottoinsieme degli operatori e delle funzionalità. WML offre anche una funzione di timer che permette di lanciare azioni (*switching, scrolling...* basate sul tempo.

I vantaggi derivanti dall'utilizzo dell'ambiente XML si sono manifestati principalmente nello sviluppo della versione Wap di EIHA?!? . Due sono le caratteristiche da considerare nell'implementazione di un servizio Wap:

- Contrariamente ad un *browser* per il Web, il *gateway* Wap accetta solo file validati tramite la DTD WML per poterli compilare.
- Come mostrato nella figura 7, la versione Wap necessita una elaborazione abbastanza complessa dei dati XML (*card* multiple, menu contestuali,...)

Queste possibili difficoltà sono state superate grazie all'utilizzo, nel nostro sistema, di un motore XSLT. Per costruzione, XSLT produce un XML già ben formato, e con qualche accorgimento anche il WML prodotto è conforme alla DTD e questo già dalla prima versione di EIHA?!? . Anche se diverse caratteristiche di XSLT possono considerarsi poco funzionali (specialmente la mancanza di supporto per le trasformazioni dinamiche), è veramente semplice modificare l'albero del risultato sfruttando la struttura di quello di input. Consigliamo di seguire un approccio simile per lo sviluppo di servizi Wap.

6 Modulo Geografico

Uno dei primi prototipi del nostro sistema utilizzava un *applet* Java per la visualizzazione delle risorse nella carta geografica. L'ultima versione di EIHA?!? utilizza il *plugin* SVG sviluppato da Adobe [5]. Il formato SVG [4] è attualmente un working draft del W3C. SVG è un dialetto XML che si basa su una DTD specifica e che permette interazioni con il Document Object Model [1]. Questo formato è un'ottimo strumento per distribuire e visualizzare disegni vettoriali. Le interazioni con il *browser* utilizzano il modello a eventi del DOM e possono essere controllate attraverso JavaScript.

Il perimetro delle aree è stato estratto dal sistema GIS proveniente dall'ESRI [8], ed è memorizzato nella nostra base di dati. Come mostrato in figura 1, il file SVG è generato automaticamente e mantenuto nella cache dalla servlet.

Abbiamo creato un link bidirezionale fra la mappa e la descrizione delle risorse poiché tutte le risorse Web sono georeferenziate durante il passo di classificazione dei siti. Questo link permette di mostrare o nascondere risorse sulle pagine Web secondo le selezioni geografiche effettuate, e simmetricamente di illuminare le risorse sulla mappa (vedi figura 8).

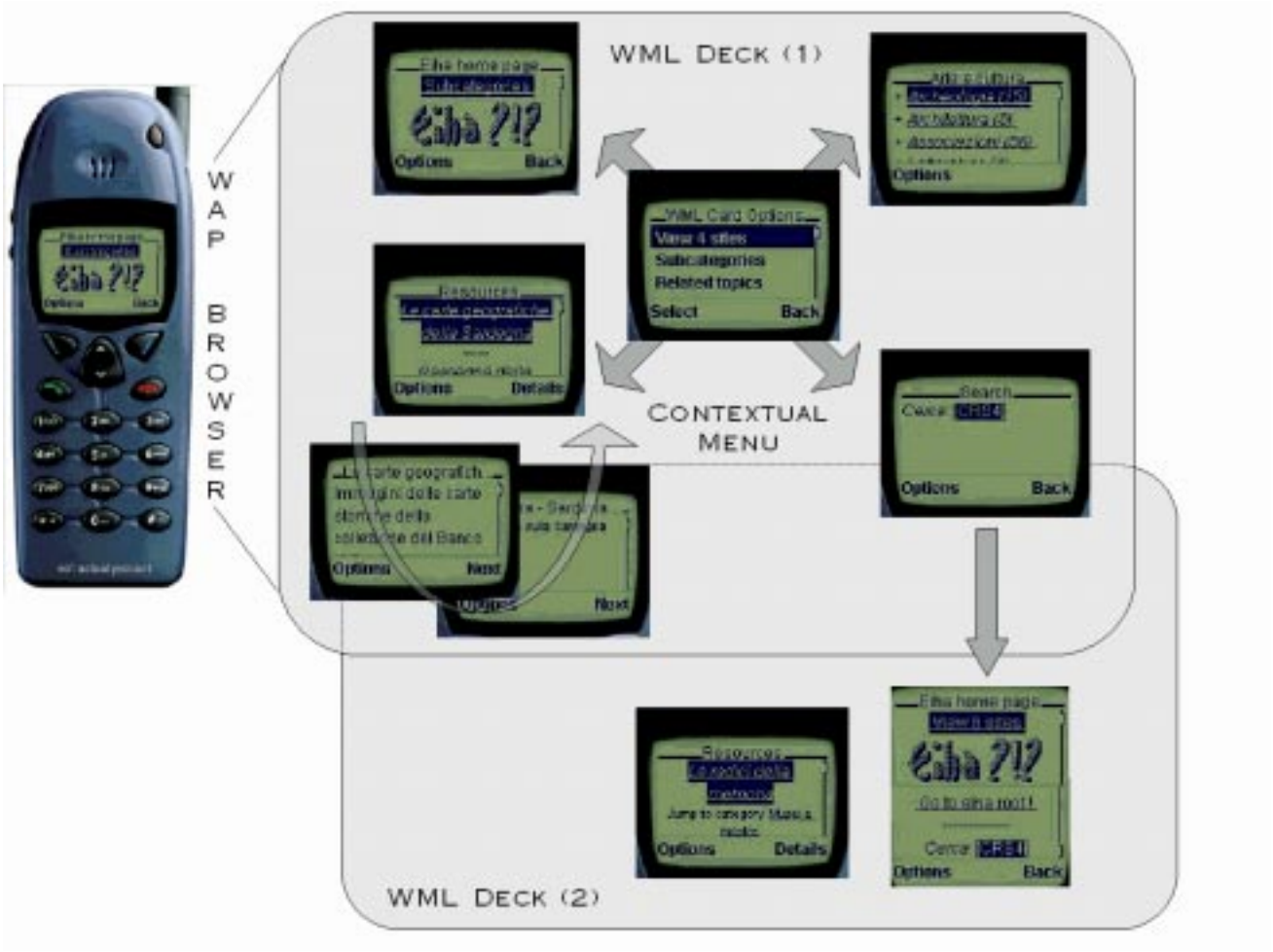

Figure 7: Una visione di Eiha?!? WML

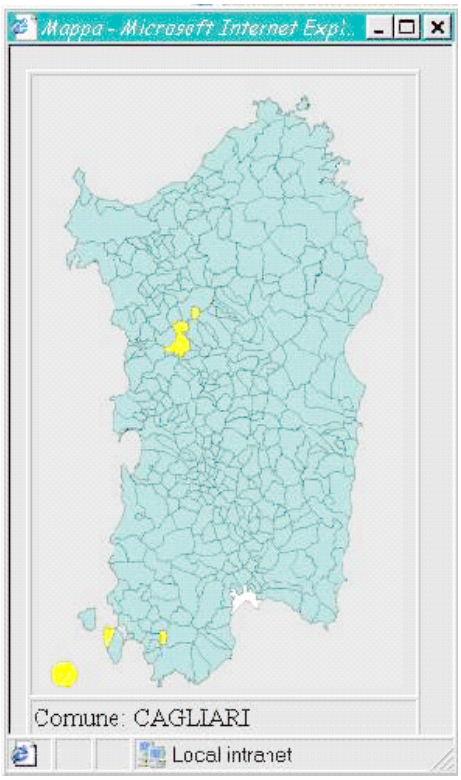

Figure 8: Un insieme di comuni illuminati sulla mappa geografica di EIHA?!?

Riflessioni e Conclusione

La realizzazione di EIHA?!? , nel contesto descritto, ci ha dato una serie di puntatori interessanti riguardo i diversi aspetti della tecnologia XML. In questa sezione vogliamo fare un sommario degli insegnamenti appresi.

All'inizio del progetto abbiamo deciso di usare XML come linguaggio di *middleware*. Per poter sviluppare la nostra architettura, abbiamo dedicato una parte importante del nostro tempo alla definizione del nostro insieme di DTD. Questo lavoro ci ha permesso di definire chiaramente la sintassi fra i vari moduli. Comunque ci siamo resi conto che il meccanismo di inclusione delle DTD è uno strumento debole per il riutilizzo.

E' abbastanza semplice introdurre una base di dati relazionale in un sistema basato su XML. I sistemi di *query* guidate da *template* sono potenti e versatili per l'estrazione dell'informazione. In questa circostanza i due modelli dei dati, DTD e schema della base di dati, devono essere mantenuti. Siccome stiamo utilizzando XML per rappresentare conoscenza e non documenti, crediamo che lo schema della base di dati potrebbe essere estratto in maniera semi-automatica dell'insieme delle DTD. Inoltre, per automatizzare completamente il meccanismo è necessario aggiungere dell'informazione alle DTD: questa informazione permetterebbe lo sfruttamento delle peculiarità della base di dati quali i *datatype* evoluti o l'utilizzo di controlli di integrità. Per lo sviluppo di progetti ampi occorre senza dubbio seguire questa strada al fine di minimizzare gli sforzi per l'aggiornamento delle strutture dati.

Uno dei benefici apportati dalla tecnologia XML è la presentazione dell'informazione insieme alla meta-informazione. Questa peculiarità non è direttamente disponibile nella nostra architettura attuale, che fornisce informazioni pre-processate. Un *backdoor* permette una vista in formato XML dell'informazione. Vogliamo sottolineare che questa possibilità è disponibile non solo per i dati standard, ma anche per l'informazione geografica: nella nostra idea il formato SVG sta ai dati geografici

come il formatting object (fo:) sta ai dati testuali.

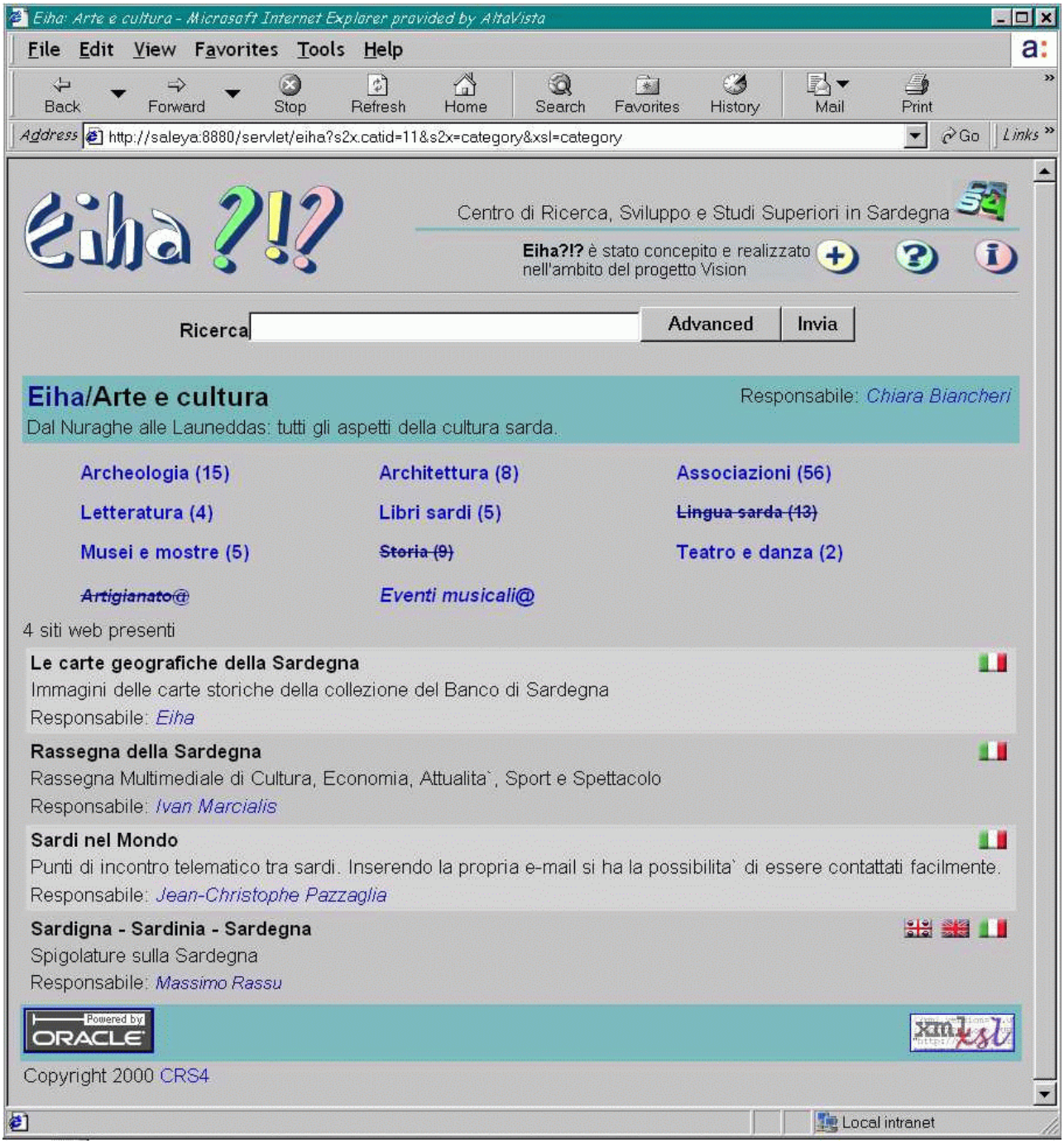

Figure 9: La versione HTML di Eiha?!?

Essendo Eiha?!? la nostra prima esperienza di sviluppo di sito multimodale abbiamo optato per una piattaforma di sperimentazione delle tecnologie XML. Queste apportano dei reali vantaggi per quanto riguarda lo sviluppo di siti Web non standard. Speriamo di aver condiviso con i lettori ciò che abbiamo acquisito attraverso questa esperienza.

Come ogni sito Web, Eiha?!? è in continua evoluzione. Attualmente ci siamo orientati allo sviluppo delle applicazioni complementari che permettono la gestione automatica delle inserzioni e dell'aggiornamento dei dati.

References

[1] Document object model (dom) level 1.0. http://www.w3.org/TR/REC-DOM-Level-1/, October 1998. W3C Recomandation.

[2] Extensible markup language (xml) 1.0. http://www.w3.org/TR/REC-xml, February 1998.

[3] Extensible stylesheet language (xsl) 1.0. http://www.w3.org/TR/xsl/, February 2000.

[4] Scalable vector graphics 1.0 specification. http://www.w3.org/TR/SVG/, March 2000. W3C Working Draft.

[5] Adobe. Svg technology preview web site. http://beta1.adobe.com/svgpreview_alpha/SVG/main.html, January 2000.

[6] Boutter, R. Xml and databases. Tech. rep., Technical University of Darmstadt, 2000.

[7] Clark, J. Xsl transformation (xslt) 1.0. http://www.w3.org/TR/xslt/, November 1999.

[8] ESRI. Gis and mapping software. http://www.esri.com, January 2000.

[9] Martin, B., and Hjelm, J. Wap forum - w3c cooperation white paper. http://www.w3.org/TR/NOTE-WAP, October 1998.

[10] Nokia. *WML Reference, version 1.1.* P.O. Box 226, FIN-00045 NOKIA GROUP, September 1999.

[11] PROJECT, J. Tomcat, is a world-class implementation of the java servlet 2.2 and javaserver pages 1.1 specifications. http://jakarta.apache.org, March 2000.

[12] Tchistopolskii, P. A. Pxslservlet. http://www.pault.com/Pxsl/, March 2000.