March 1994

Application of SGML and OODB Techniques in a Textual Database

Jian Zhang
University of Pennsylvania

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-94-14.

Application of SGML and OODB Techniques in a Textual Database

Abstract

An electronic dictionary system (EDS) is developed using object-oriented database techniques based on ObjectStore. The EDS is basically composed of two parts: the Database Building Program (DBP), and the Database Querying Program (DQP). The DBP reads in a dictionary encoded in SGML tags, and builds a database composed of a collection of trees which holds dictionary entries and several lists which contain values of various lexical categories. With text exchangeability introduced by the SGML, the DBP is able to accommodate dictionaries of different structures, after modifying its configuration file. The DQP adopts SQL-like syntax and handles queries by exploiting the category lists through a optimization procedure. Initial tests show that compared with relation database, the DQP enjoys much faster speed and offers much higher flexibility in setting both lexical criterion and context requirements.

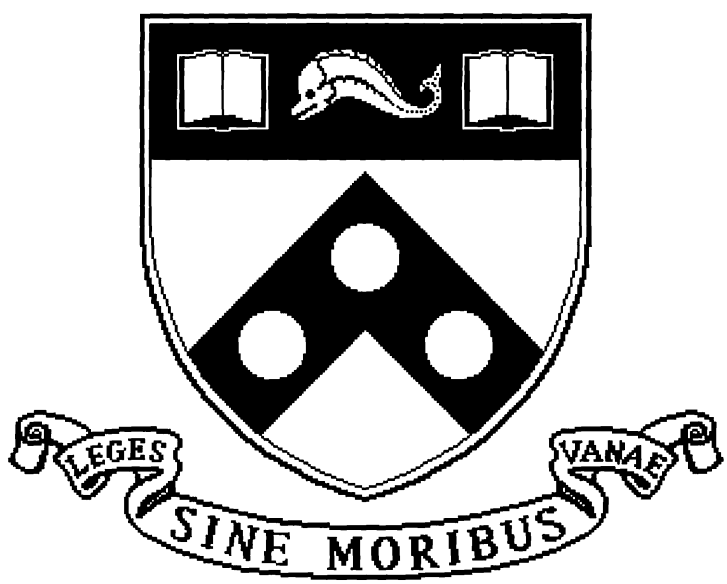
Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-94-14.

Application of SGML and OODB Techniques In a Textual Database

MS-CIS-94-14

Jian Zhang

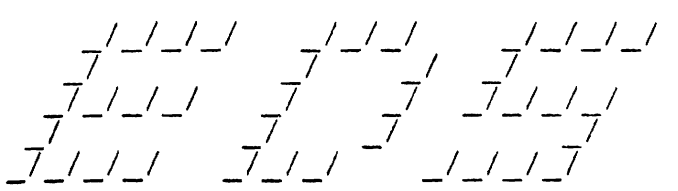


University of Pennsylvania
School of Engineering and Applied Science
Computer and Information Science Department
Philadelphia, PA 19104-6389

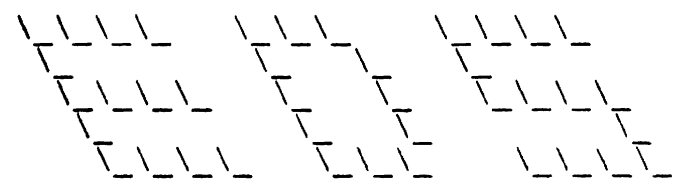
March 1994

Appendix A Example of EDS Runs

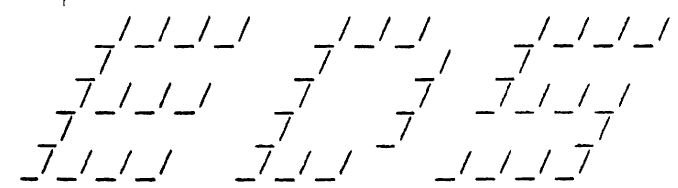
```
saul.cis.upenn.edu% edsload in card0 /jian/db1
```



```
Database /jian/db1 alread exists
Overwrite   Modify   Exit   Help > h

 Overwrite    --- overwrite the existing database
 Modify       --- add new entries into the existing database
 Exit         --- exit the OODD program
 Help         --- to get this message

Overwrite   Modify   Exit   Help > o
   --- Spanish Dictionary ---
Reading dictionary ...
done

==========================
 Entries          70
 Words            1183
 Characters       6259
==========================

Database /jian/db1 has been established. Bye!
```



```
saul.cis.upenn.edu% edsq
```



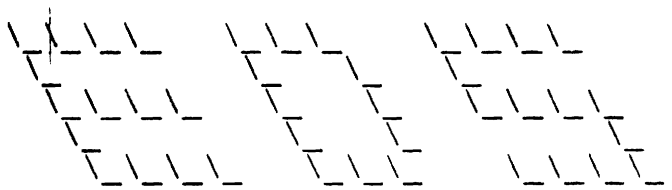
```
   --- Spanish Dictionary ---

query > a
;
ERROR: invalid word a

query > help;
COMMAND:   About   sTatistics  List   Select   Quit > a
about --- describ EDS
SYNTAX: about;

COMMAND:   About   sTatistics  List   Select   Quit > t
statistics --- report statistics of the dictionary
SYNTAX: statistics;

COMMAND:   About   sTatistics  List   Select   Quit > l
list --- list element values
SYNTAX: list <element> = <value>
   e.g.   list pos = v, i.;
```

```
COMMAND:   About   sTatistics  List   Select   Quit > s
select --- display selected entries or paths
SYNTAX: select <element list> [formatted]
        where <leaf element> = <value>[, <value>]
              [in <ancestor element>]
              [and ...]
   e.g. 1) select form sense  where geo = Arg.;
        2) select all formatted
           where style = fig. in form and
                 text = a*, b* in sense;

COMMAND:   About   sTatistics  List   Select   Quit > q

query > about;

     //___   //__\   //___
    //__    //__/   ___//

=======================================
     Electronic Dictionary System
       (object-oriented tech.)

     Author:          Jian Zhang
     Organization:    CIS. UPENN.
     Date:           April, 1993
=======================================


query > statistics;

       DATABASE STATISTICS
==================================

Dictionary Size:
---------------
Entries                          1462
Words                           27720
Characters                     150715
Avg. words/entry                   19
Avg. word length                    5

Element List Sizes:
------------------
text                             7191
pos                                19
grammar                            25
domain                             34
style                               8
geography                          21
language                           14
foreign                           283
==================================


query > list pos = *;
19 items found.  Display? [y/(n)] > y
adj.
adv.
amb.
com.
conj.
f.
```

```
interj.
m.
n. pr.
prefijo
prep.
s.
s. f.
s. m.
v. auxiliar.
v. i.
v. r.
v. substantivo
v. t.

query > select all where pos = prep.;
1 entry found.
Display   Constraint   Alternative   Select   Query   Exit   Help > d

  SEG'UN prep. ( lat. secundum ) Con arreglo a: seg{'u}n eso no
  vendr{'a}. ||-- Adv. Como, con arreglo a: se le pagar{'a} seg{'u}n
  lo que haga. || Con arreglo a lo que dice otro: seg{'u}n San Mateo.
  || Seg{'u}n y como, m. adv. de igual manera. || Seg{'u}n y conforme,
  m. adv. seg{'u}n y como. Tambi{'e}n se usa en sentido de duda: {?}Lo
  har{'a}s ma{n}ana? -Seg{'u}n y conforme. Abr{'e}viase a veces en
  seg{'u}n.

Display   Constraint   Alternative   Select   Query   Exit   Help > h

 Display     --- display the matched paths / entries
 Constraint  --- add more search constraints
 Alternative --- add alternative search conditions
 Select      --- select elements for display
 Query       --- start a new query
 Exit        --- exit the OODD program
 Help        --- to get this message

Display   Constraint   Alternative   Select   Query   Exit   Help > s

select > help;
Select elements to display
SYNTAX: <element list> [formatted]
  e.g.  1) form sense;
        2) all formatted;

select > all formatted;
1 entry found.
Display   Constraint   Alternative   Select   Query   Exit   Help > d


[SEG'UN]

(prep.)
  lat. secundum
    Con arreglo a: seg{'u}n eso no vendr{'a}.

(Adv.)

    1. Como, con arreglo a: se le pagar{'a} seg{'u}n lo que haga.

    2. Con arreglo a lo que dice otro: seg{'u}n San Mateo.

    3. Seg{'u}n y como, m. adv. de igual manera.
```

4. Seg{'u}n y conforme, m. adv. seg{'u}n y como. Tambi{'e}n se usa en sentido de duda: {?}Lo har{'a}s ma{n}ana? -Seg{'u}n y conforme. Abr{'e}viase a veces en seg{'u}n.

```
Display   Constraint   Alternative   Select   Query   Exit   Help > q

query > select path where text = *cion in sense;
2 paths found.
Display   Constraint   Alternative   Select   Query   Exit   Help > d

  SALA f. Habitacion donde se constituye un tribunal:

  $ALVACI'ON f. Acci{'o}n y efecto de salvar o salvarse: {'a}ncora de
  $alvacion. SIN'ON. V. Rescate.

Display   Constraint   Alternative   Select   Query   Exit   Help > q

query > list text = *c*i*s*;
245 items found.  Display? [y/(n)] > n

query > list text = *cis*;
8 items found.  Display? [y/(n)] > y
decisi{'o}n
francisco
galicismo
precisa
precisar
preciso
solecismo
trapecista

query > quit;

Bye, use me again!
```



Appendix B. SGML DCD for the Sample Dictionary

```
<!-- SGML Document Type Definition -->
<!-- Pequeño Larousse Ilustrado Spanish Dictionary 1991/
     (ISBN 970-607-006-0) -->
<!-- Language Analysis Center, 9/15/92 -->
<!-- Authors: Jian Zhang, Heather Davenport -->
<!-- NOTE: changes in tags (deleting space in tags,
                            due to XGML naming convention)
        <cross reference>  becomes    <crossReference>
        <related entry>    becomes    <relatedEntry> -->

<!DOCTYPE LarousseSpanDict [

<!ELEMENT LarousseSpanDict o o (entry|crossReference)+ >

<!ELEMENT entry - - (#PCDATA, F, #PCDATA, S, #PCDATA,
                     ((S|relatedEntry|note), #PCDATA)* ) >

<!ELEMENT crossReference - - (#PCDATA, F, #PCDATA,
                              (unote, #PCDATA)*,
                              (note, #PCDATA)+ ) >

<!ELEMENT F  - - (#PCDATA, pform, #PCDATA,
                  ((pform|pron|gnote|unote|lg|note|etym),#PCDATA)*)>

<!ELEMENT relatedEntry - - (#PCDATA,
                            (((gnote,#PCDATA)+,
                              ((S|relatedEntry),#PCDATA)+,
                              (note,#PCDATA)* ) |
                             (note,#PCDATA,
                              ((syn|ant|par|note),#PCDATA)+)) ) >
<!ELEMENT S - - (#PCDATA,
                 ((gnote|unote|lg|note|descrip|relatedEntry),#PCDATA)+) >

<!ELEMENT descrip - - (#PCDATA,
                       ((gnote|unote|note|eg),#PCDATA)* ) >

<!ELEMENT etym - - (#PCDATA, lg, #PCDATA, orth, #PCDATA,
                    ((lg|orth|gloss|pron|note), #PCDATA)* ) >

<!ELEMENT gnote - - (#PCDATA, (pos|gram), #PCDATA,
                     ((pos|gram|note|orth), #PCDATA)* ) >

<!ELEMENT unote - - (#PCDATA, (dom|styl|geog), #PCDATA,
                     ((dom|styl|geog), #PCDATA)* ) >

<!ELEMENT (pform|eg) - - (#PCDATA, (note, #PCDATA)*) >

<!ELEMENT note - - (gnote|lg|note|orth|pron|S|eg|xref|#PCDATA)+ >

<!ELEMENT   (orth|pron|xref|syn|ant|par|gloss|
pos|gram|dom|styl|geog|lg) - - (#PCDATA) >   ]>
```

Appendix C. Document Instance of the Sample Dictionary

```
<entry>
  <F>
    <pform> ACTIVIDAD </pform>
    <gnote>
      <pos> f. </pos> </gnote> </F>
  <S>
    <descrip> Facultad de obrar:
      <eg> la actividad del fuego. </eg> </descrip> </S> ||
  <S>
    <descrip> Diligencia, eficacia. </descrip> </S> ||
  <S>
    <descrip> Prontitud en el obrar:
      <eg> actividad en el esp{'i}ritu. </eg> </descrip>
    <related entry> (
      <note> SIN'ON. </note> V.
      <syn> Acci{'o}n. </syn> ) </related entry> </S> ||
  <related entry> || --
    <note> CONTR. </note>
    <ant> Pereza, </ant>
    <ant> desidia. </ant> </related entry> </entry>

<entry>
  <F>
    <pform> ACERCAR </pform>
    <gnote>
      <pos> v. t. </pos> </gnote> </F>
  <S>
    <descrip> Poner cerca lo que estaba lejos, aproximar:
      <eg> acercar la silla. </eg> </descrip> </S>
  <related entry> || --
    <gnote>
      <pos> V. r. </pos> </gnote>
    <S>
      <descrip>
        <eg> acercarse a uno. </eg> </descrip> </S>
    <related entry> ||--
      <note> CONTR. </note>
      <ant> Alejar. </ant> </related entry> </related entry> </entry>

<cross reference>
  <F>
    <pform> ACAB'OSE </pform>
    <gnote>
      <pos> m. </pos> </gnote> </F>
  <note> V.
    <xref> ACABAR. </xref> </note> </cross reference>
```

Appendix D. Header File of the EDS

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <malloc.h>
#include <ctype.h>
#include <iostream.h>
#include <ostore/ostore.hh>
#include <ostore/coll.hh>
#include <ostore/relat.hh>

#define W               80
#define M               20
#define ANY            999
#define OFFSET          20
#define COMMON_REF_N 9999
#define EXIT            0
#define NEXT            1

extern database* db1;

/*====================== CLASSES ======================*/

class ITEM;
class NODE;
class PROF;
class delim;
class disp;
class hide;
class paren;
class cond;

class ITEM {
    char *itext indexable;
    os_Set<NODE*> nodes;
  public:
    char *Text () { return itext; }
    os_Set<NODE*> Nodes () { return nodes; }
    int  has_ref (NODE *cp) { return nodes.contains(cp); }
    void add_ref (NODE *cp) { nodes.insert(cp); }
    void show () { cout << itext << endl << flush; }
    ITEM (char*);

};

class NODE {
    char _symb;
    char *ntext;
    os_List<ITEM*> items;
    os_List<NODE*> children inverse_member parent;
    NODE *parent inverse_member children;
  public:
    char Symb () { return _symb; }
    char *Text () { return ntext; }
    NODE *Parent () { return parent; }
    void insert_item (ITEM *ip) { items.insert(ip); }
    void set_parent (NODE *par) { parent = par; }
    void set_text (char*);
    void leaf_text (char*, char*);
    void restore_text (char*);
    int  has_anc (NODE*);
    int  has_anc2 (char);
    int  has_sib ();
    void find_route (char*);
    os_List<NODE*> get_path (NODE*,os_List<NODE*>);
```

```
    NODE *get_anc (char);
    NODE *get_rt ();
    int  get_paren (char*, char*);
    int  match (cond*,os_Set<NODE*>*);
    int  matched_node (cond*);
    void show_selected (char*,int);
    void show_tree (int);
    void show_leaf (char*,int);
    void show_index ();
    void show_text (int);
    int  hiding (NODE*);
    NODE (char);
};

class PROF {
    char *_tag;
    char *_next;
    char _symb;
    int _lid;
    char *_name;
  public:
    persistent<db1> os_Set<PROF*> extent;
    char *Tag () { return _tag; }
    char *Next () { return _next; }
    char Symb () { return _symb; }
    int  Lid () { return _lid; }
    char *Name () { return _name; }
    PROF (char*,char*,char,int,char*);
};

class delim {
    char _symb1;
    char _symb2;
    char *_deli;
  public:
    char Symb1 () { return _symb1; }
    char Symb2 () { return _symb2; }
    char *Deli () { return _deli; }
    delim (char,char,char*);
};

class disp {
    char *_tsymbs;       // symbols of target components
    char _psymb;         // symbol of an intermediate parent component
    char *_disps;        // symbols of showable children of the intermediate
  public:                  // component given the target components
    char *Tsymbs () { return _tsymbs; }
    char Psymb () { return _psymb; }
    char *Disps () { return _disps; }
    disp (char*,char,char*);
};

class hide {
    char _tsymb;
    char _asymb1;
    char _hsymb;
    char _asymb2;
  public:
    char Tsymb () { return _tsymb; }
    char Hsymb () { return _hsymb; }
    char Asymb1 () { return _asymb1; }
    char Asymb2 () { return _asymb2; }
    hide (char,char,char,char);
};

class paren {
    char _open;          // opening parenthesis of text
```

```
    char _close;          // closing parenthesis of text
    char *_route;         // route from root to the text component
  public:
    char Open () { return _open; }
    char Close () { return _close; }
    char *Route () { return _route; }
    paren (char,char,char*);
};

class cond {
    char _leaf_symb;
    os_Set<char*> _values;
    char _anc_symb;
    os_Set<ITEM*> items;
    int  ref_n;
  public:
    char Leaf_symb () { return _leaf_symb; }
    os_Set<char*> Values () { return _values; }
    char Anc_symb () { return _anc_symb; }
    os_Set<ITEM*> Items () { return items; }
    int  Ref_n () { return ref_n; }
    void insert_items (os_Set<ITEM*> itms) { items |= itms; }
    void incre_ref_n (int n) { ref_n += n; }
    void get_value (char*);
    cond (char,os_Set<char*>,char);
};

/*====================== Global Variables  ======================*/
persistent<db1> os_Set<ITEM*> word_list ;
persistent<db1> os_Set<ITEM*> pos_list ;
persistent<db1> os_Set<ITEM*> gram_list ;
persistent<db1> os_Set<ITEM*> dom_list ;
persistent<db1> os_Set<ITEM*> styl_list;
persistent<db1> os_Set<ITEM*> geog_list;
persistent<db1> os_Set<ITEM*> lg_list  ;
persistent<db1> os_Set<ITEM*> orth_list;
persistent<db1> os_Set<char*> common_words;

persistent<db1> int ent_n, wd_n, ch_n;

extern os_Set<delim*> delim_extent;
extern os_Set<disp*> disp_extent;
extern os_Set<hide*> hide_extent;
extern os_Set<paren*> paren_extent;

extern FILE *dict, *card;
extern char valid_punct[], dummy[], chstr[],
            rt_symbs[], re_symb, sen_symb, des_symb,
              leafs[], abbvs[], gunotes[], hdwd_route[],
            listnames[], dict_title[];
extern os_Set<ITEM*> *lists[];
extern int  list_n, max_ref, line, column, indent, sn, dn;

/*====================== Function Prototypes ======================*/

void open_files(int, char**);
void init_db(char*);
void init_lists();
ITEM *new_item(char*);
NODE *new_node(char);

// ----- member method/extension module -----
int  stol(char);
int  ntol(char*);
int  ttos(char*,char*,char*);
int  ttol(char*,char*,int*);
char ltos(int);
```

```
char ntos(char*);
char *ston (char);
char *get_deli(NODE*,NODE*);

// ----- input reading module -----
void read_card();
void read_dictionary();
void read_tagged(NODE*);
void read_text(NODE*,char,int);
void link_list(char*,int,NODE*,int);
void read_next(char*);
int  shrink(char*,char*);
int  cap_pattern(char*);

// ----- query module -----
void querying();
int  select(char**);
int  list(char**);

void get_query(char*,char*);
int  get_sele_symbs(char**,char*);
int  get_format(char**);
int  get_condition(char**,os_List<cond*>*);
int  get_values(char**,os_Set<char*>*);
void insert_condition(cond*,os_List<cond*>*);

int  get_matches(char**,int*,os_Set<NODE*>*);
int  search_list(int,char*,os_Set<ITEM*>*);
void get_items(cond*);
void get_nodes(cond*,os_Set<NODE*>*,int);
void check_nodes(NODE*,cond*,int,os_Set<NODE*>*);
void search_global(cond*,os_Set<NODE*>*);

void report_matches(char,int,os_Set<NODE*>);
void show_matches(char*,int,int,os_Set<NODE*>);
void show_path(os_List<NODE*>,int);

void convert_to_root(os_Set<NODE*>*);
void sort_nodes(os_List<NODE*>*);
void sort_words(os_List<char*>*);

// ----- help module -----
void about();
void statistics();
void prompt_help(char*);
void query_help();
void select_help();
void and_help();
void or_help();
void where_help();
void eg_help();
void title1();
void title2();
void title3();

// ----- utility module -----
char menu(char*,char*,int);
void menu_help(int);
int  skip();
char *skip2(char*);
void scan_quote(char**,char*,int*);
void scan_abbrv(char**,char*,int*);
void sscanf3(char**,char*,int*);
char *sscanf2(char*,char*);

void print(char*,int);
void indenting();
```

```
void error(int, char*);
int  err_msg(char*,char*);

void del_punct(char*,int);
void restore_cap(char*,char*,char);
void tolowers(char*,char*);
char *ctos(char);
char *tail(char*);
int  is_punct(char*);

int  in_text(char*,NODE*);
int  next_is(char*,char*);
int  check_next(char**,char*);
int  strcmps(char*,char*);

char *alloc(char*);
char *alloc2(char*);
```