DYNAMIC DATABASE MANAGEMENT IN COMPUTER AIDED RESIDENTIAL DISTRICT DESIGN SYSTEM

JINGWEN GU
College of Architecture and Urban Planning
Tongji University, Shanghai, CHINA
Email: jwgu@tju.ihep.ac.cn

AND

GUANGHUI XIE
Jinshan Urban Planning Bureau, Shanghai, CHINA

Abstract. Compare with business database, the engineering database is much more difficult to be designed and implemented. In this paper the issues on the dynamic engineering component database and its management involved in implementation of an AutoCAD based experimental CAD system, Computer Aided Residential District Planning and Design System (CARPDS), are discussed. The discussions focus on the organization, access control of the database and the supporting environment.

1. Introduction

Organization and management of a collection of data files relating to an ongoing project are important (W. J. Mitchell and M. McCullough 1991); construction and maintenance of a well-defined special database applicable to a definite task are also crucial in some fields. At present many popular commercial DBMS, such as DBASE, INFORMIX, SYBASE and ORACLE etc, can be used to develop ordinary business databases effectively. However, no such DBMS is currently available for providing all of the facilities required in all engineering applications.

CAD applications often impose additional requirements that are not adequately provided for by conventional DBMS. This is mainly due to the special characteristics of CAD data and the nature of the desired processing functions, and has led to the development of purpose-written DBMS for CAD (C. J. Anumba 1996). Compare with business database, the engineering database in a CAD environment is much more difficult to be designed, implemented, manipulated and managed due to the following facts. First, CAD

CAADRIA '98 : Proceedings of The Third Conference on Computer Aided Architectural Design Research in Asia.
eds. T. Sasada, S. Yamaguchi, M. Morozumi, A. Kaga, and R. Homma
April 22-24, 1998. Osaka University, Osaka, Japan. Pp. 243-251

applications handle a wider range of data types including geometric and non-geometric data. The records in an engineering database are typically variable in length, which make the storage of and access to the data complicated. Secondly, the interactive graphics or CAD environment is absolutely necessary for such normal operations as append, update of geometric data items. Thirdly, integration of CAD data structure and the user interface is now recognized as an important aspect of engineering systems (C. J. Anumba 1996). The graphical exhibition of geometric data items is undoubtedly helpful to user reference to or selection of elements during the operation on database. With the rapid advances of computer applications in engineering fields, issues associated with data structures and DBMS for engineering CAD systems become more and more challenged ones in development. In this paper the issues on the dynamic engineering database and its management involved in implementation of an experimental CAD system, Computer Aided Residential District Planning and Design System (CARPDS), are discussed. The emphases are put on the organization, access control of the database and the supporting environment.

2. Organization of the Database

2.1. SYSTEM OVERVIEW

The planners in China now widely use the CAD technology in their work on urban planning and design. However, most of them use CAD only as a drafting rather than design tool because of the lack of appropriate software that can provide the facilities they need in the planning and design. Planning and design of a great lot of residential districts (quarters) in China is one of the heavy tasks that the Chinese planners and architects are facing to. During the schema planning and design of residential district, the planners have to consume much of their working time in tedious works on graphics-oriented site planning and analyses.

The CARPDS will be designed to relieve the planers' burden in residential district planning and design and act as not only drawing aids but also design tools. It will provide planners the supports in the planning cycle from site input, road planning, and land block division to building layout and design, environmental constituent assignment, index statistics, report making and result output, and from qualitative display to quantitative analysis. Figure 1 shows the functional structure of CARPDS. It can be seen from the figure that the database and its management play a key role and are in fact the core of the system.

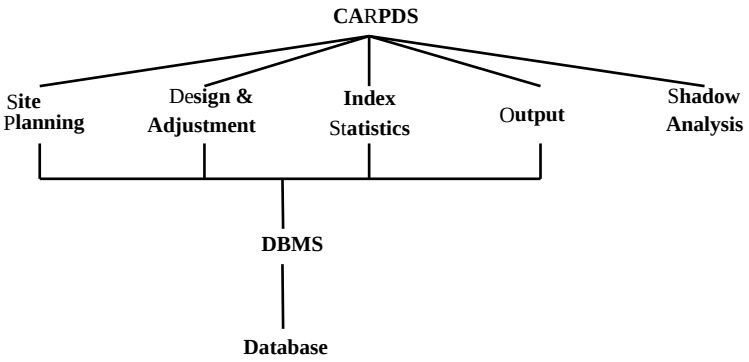

Figure 1. System Structure

The planner may need to refer to the regulations held in the database, so that he can make the site planning subjected to design specifications. In design or adjustment the designer can search the database according to some conditions to find a case of single building, housing block or other planning primitives as an "assembly part". Or he can select a constructing mode to synthesize separate units into new house building or translate a floor plan to generate a multi-story building and then add the derived building as a new case into the database. The necessary attribute data, such as architectural or base area, number of stories and type of buildings etc, can be linked to the referenced entities in the design, fetched from the database and used to make a report or index statistics for the planning schema.

2.2. DATABASE ORGANIZATION

As we can see from above section, the database is functionally composed of three types of libraries, specifications library, case library and "generative mode" library. Figure 2 shows the logical hierarchical structure of the database organization. The information in the specification library is organized based on text, and the generative mode library contains pre-defined functions that act upon the simple architectural primitives with parameters to produce a complex. The two libraries will never be altered once they are created. Therefore, they are substantially static and will not be considered here.

The case library contains different levels of planning parts (cases) created in advance or in previous planning or design project. The parts can be individually used as working cases in current project or can be assembled to form a new level of case. Some housing buildings, for instance, can assemble the housing block, and housing blocks, public buildings, gardens plus other site parts can assemble a residential quarter. Horizontally and structurally, the case library can be consisted of several sub case libraries according to the classes of

architectural and environment components, of which some contain more than one groups (clusters) of elements (see figure 2). Vertically and functionally, each element in the case library should contain two parts of data, one case model including 3-D or 2-D geometric block and necessary attributes, and up to four reference views (icon buttons). The model case will be directly used in the design and the icons will be used in user interface for friendly and easy reference. For example, the model case of a housing building will include a 3-D building model and its related attributes, such as the base and architectural area, the number of floors, the number of different types of housing units, etc. And the reference icons may be such views as front and side elevations and perspective views. In practice, the number and the contents of the elements available in the system case library are typically different from city to city and region to region, and will grow up with the system being put into real applications. That is, the case library is dynamically variable, and the main concerns in this paper are right the issues on management of such a "dynamic extendable" case library. Physically, the case library should be organized in such a way that it not only meets the special requirements in the design environment and but also allows the normal operations for a database. Firstly and basically, each of its main groups of elements can be inquired quickly with or without conditions or referenced during the design. Secondly, its elements can be partially or entirely updated, be deleted and appended.

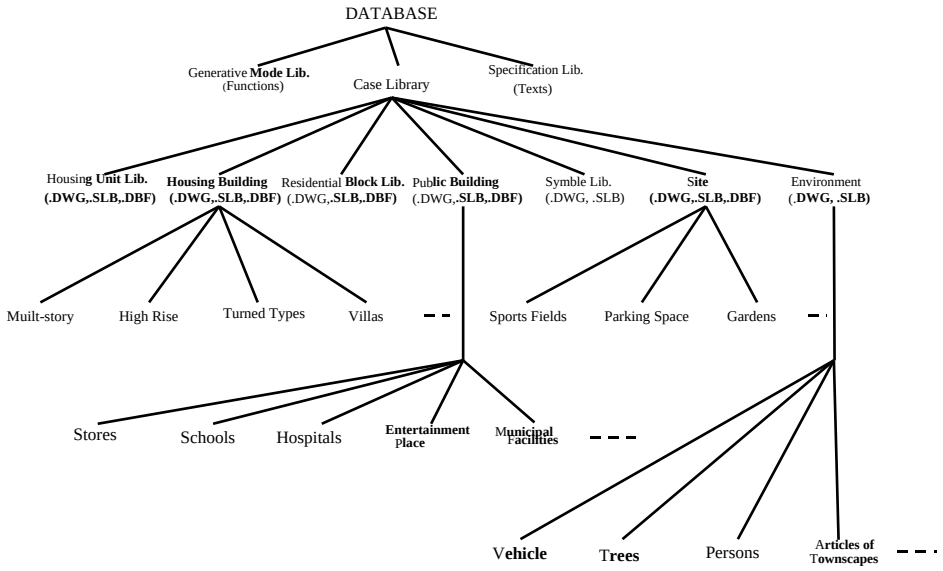

Figure 2. Logical Hierarchical Structure of Database

In AutoCAD, the slides are right good solutions to the reference views of the element and the SLD files are easily obtained by the command, MSLIDE. A

single SLB file, at the same time, can be used to put all slides corresponding to those elements within the same class together. That is, one and only one SLB file is required for a sub case library.

The standard AutoCAD DWG file may, at the other hand, contain both geometric and attributes entities and can be used as a case model. However, such a file integrated with geometric and attributes data is not ideal and difficult for conditional inquiry to the engineering database. In CARPDS all elements in the same class of architectural components are subjected to have the same number of attribute fields. A DBF file is then created for each class and the attributes of every element within that class is appended into the corresponding DBF file instead of being resided in its DWG model file.

By using the same structure as SLB file, all DWG models within a class can be combined in a single file (DLB). Each sub case library thus contains at most one SLB, DBF and DLB file respectively. For each element, of course, a unique name, group or class name plus serial number, is assigned to its SLD, DWG file and attribute field (Key Attribute) so that the key linkage among three files can be built.

3. Access Control of the Database

3.1. CONDITIONAL INQUIRY

As well known, inquiry is one of the most common operations on the database. For example, the planner wants to find a case, say a housing building, in the case library that fits the particular needs at hand. While retrieving of objects based on geometric shape or image characteristics is impossible now, using some conditions on pre-defined attributes can do the search. Figure 3 shows the dialog box for inquiry on housing buildings, in which the designer can give some conditions as listed. With AutoCAD SQL Interface, ADS ASI functions, the search into the DBF file can be executed. The following ADS C statements are used to construct and execute such a SQL clause as above.

```
strcpy(cluse, "SELECT * FROM house WHERE H_ID>:id1 AND H_ID<:id2");
    if (item.data.COND_ID)
      { if (item.data.AREA_flag)
          { strcat(cluse, " AND g_area>=:aa1 AND g_area<=:aa2");
            strcat(cluse, " AND base>=:ba1 AND base<=:ba2");   }
        if (item.data.FLOOR_flag)
          strcat(cluse, " AND floors>=:fl1 AND floors<=:fl2");    }
    if (asi_com(&comhandle, cluse)!=ASI_GOOD)
      { ads_alert("compile error");
        asi_post(&drvhandle, &dbhandle, &comhandle);
```

```
        return(RTERROR);   }
……
  if (asi_exe(&comhandle)!=ASI_GOOD) {
     ads_alert("searth error");
     asi_post(&drvhandle, &dbhandle, &comhandle);
     return(RTERROR);    }
```

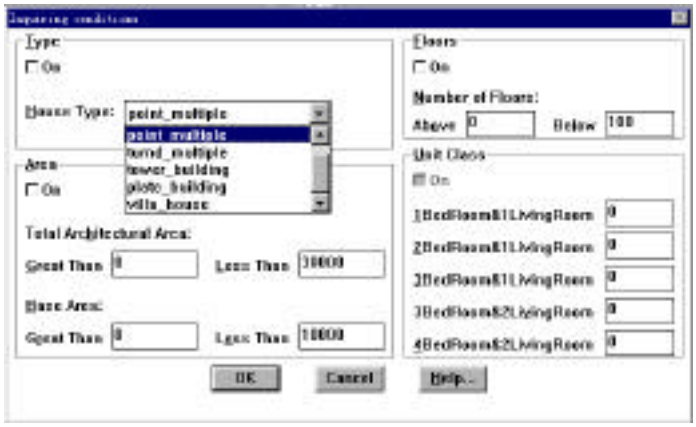

Figure 3. Conditional Query Dialog Box

All records that match the conditions are fetched and a dynamic reference list (DRL) is created based on their key attributes. With the DRL, their images in the SLB file are shown as icon buttons in the subsequent dialog box (see figure 4). The user can then double click any one of them to either insert its corresponding DWG block in the DLB file into his current design, or view its related attributes in detail. The attributes can be linked to inserted blocks and then be used at the analysis phase. The user also can pick some one of radio view buttons to switch them among different views to help him make further decision.

3.2. APPEND, DELETE AND UPDATE

The user interface for append, delete and update of the case library is similar to that shown in figure 4 with the mode radio buttons, Insert and Inquiry, replaced by Delete and Update, and a button, Append, added. A DRL is also built with these operations. Deleting an element is simplest: remove the record from DBF, DWG and SLD blocks from DLB and SLB, erase its item in DRL and refresh the display of dialog box. In fact removing of record from DBF will draw a case away the system. In this case, of course, redundant DWG and SLDs remain in the database and will be discarded by the maintainer. With AutoCAD SQL Extension (ASE) commands, the following ADS C statements can be used to delete a record.

```
cmd=ads_buildlist(RTSTR,"C:asesetrow",RTSTR,"K",RTSTR,type,RTSTR,"",0);
ads_invoke(cmd,&result);
ads_relrb(cmd);
cmd=ads_buildlist(RTSTR,"C:asedelrow",0);
return_code=ads_invoke(cmd,&result);
```

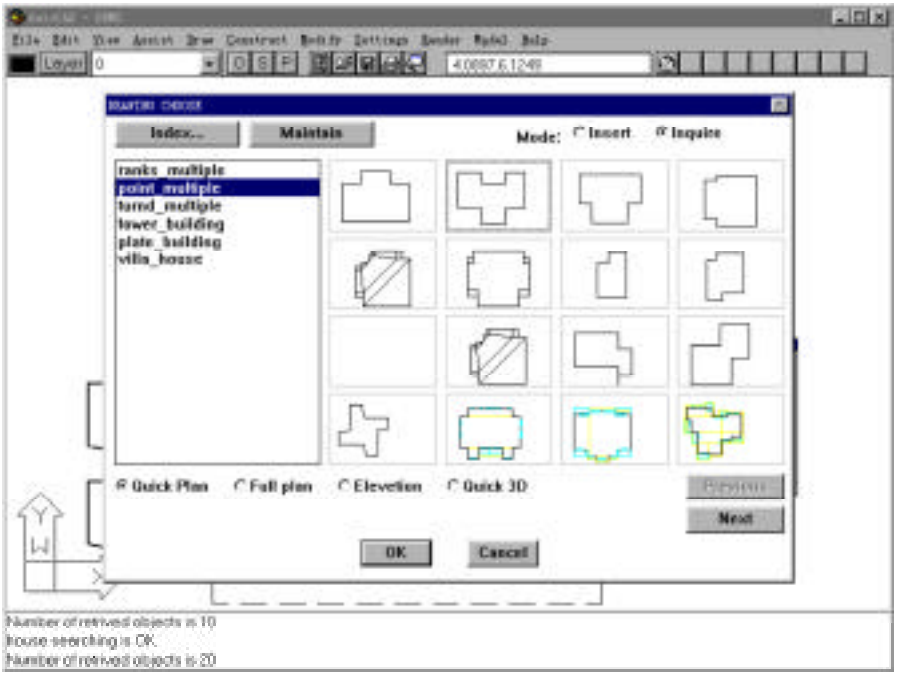

Figure 4. Dialog Box for Access Control

Procedurally, appending an element is slightly more complex because it will involve some interactive graphic actions. The main steps are as follows:

1. Create a DRL corresponding to all records in the group or class into which the element will be appended.
2. Ask the user to select objects to be grouped as desired geometric model. The objects can be from the current working drawing or from a DWG file. If the user chooses a DWG file and the selection mode is interactive, the file is first loaded into current drawing area and selection is then activated. The selection mode may be either interactive or automatic. All entities in current drawing or the DWG file will be grouped into the target DLB and the center of current drawing extents is taken as the base point if user choose automatic mode, or only those selected objects are included and the base point is also required. The appended file name is assigned based on the DRL.

3. Ask the user to input the related attributes and a record is appended into the DBF. Even if the user does not make any response to this instruction, a blank record only with the key attribute will be appended.
4. Ask the user to define up to four 3-D viewpoints so that the slides can be made and then be appended into SLB. Similar to that in step 2, both automatic and interactive modes are provided for user choosing. If user skips this step by using the cancel button, the slides will not be made and appending of element is not affected (see fig. 4).
5. Add a new item into the DRL and refresh the display of dialog box.

Note that those strings listed in the list box (see fig. 4) are group names under certain class and encoded in a separate group index file that is dynamically monitored during the operation. A new group can thus also be introduced easily.

In principle, updating an element is equivalent to the operations of deleting followed by appending with the exception of its key attribute being held. In addition, an optional mode is provided to allow the user to update only part of the element data.

3.3. MAINTAINER

As stated above, it is probably that deleting or appending an element may result in redundant or losing of data in the database. A separate maintainer is required to check and maintain the consistency and completeness of the database. It will remove the redundant data if they are found, and prompt the user to supply a deficient part if it find any. Unfortunately, the maintainer is not completed yet.

4. The supporting environment

The DBMS discussed in this paper is implemented in AutoCAD ADS, DCL, ASE and ASI. All dialog boxes used as GUI are written in DCL and controlled by ADS, ASE commands and ASI functions are used to support operation for attribute data, and ADS is also used to support the integration with the design module or stand-lone working mode. The programming language and compiling environment for the system is MS Visual C++ 1.0. The system control chart is shown in figure 5.

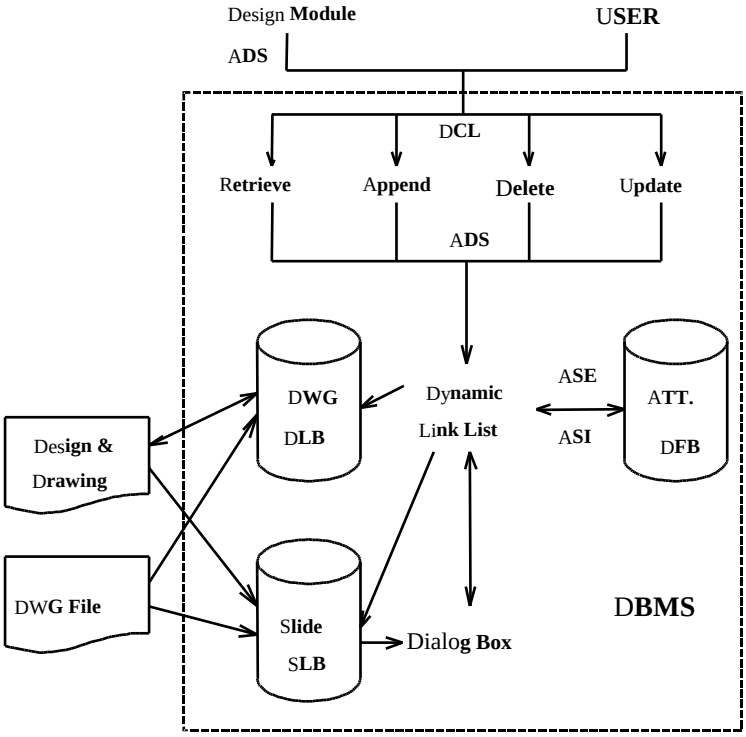

Figure 5. System Control Chart

References

Mitchell, William J. and M. McCullough: 1991, *Digital Design Media*, New York: Van Nostrand Reinhold

Anumba, C. J.: 1996, Functional integration in CAD systems, *Advances in Engineering Software* **25**(1996), 103-109

Anumba, C. J.: 1996, Data structures and DBMS for computer-aided design systems, *Advances in Engineering Software* **25**(1996), 123-129