CALPAS4 -- Four Years Later

Charles S. Barnaby
BSG Software
2560 Ninth Street, Suite 212M
Berkeley, CA 94710

ABSTRACT

In a paper presented at the 1985 predecessor of this conference, I maintained that current PC microcomputer technologies provided the opportunity to develop a new generation of graphically oriented, interactive building modeling programs. Our efforts to implement one such program, called TAKEOFF, have not been an unqualified success but do provide lessons about this type of program. Generally, the technical proficiency required by interactive graphic programming is in another realm when compared to that required for "simple" building modeling. The paper reviews some of the problems encountered during the development of TAKEOFF.

INTRODUCTION

At the Building Energy Simulation Conference (Seattle, 1985), I presented a paper entitled *CALPAS4 and Beyond -- Microcomputers, Graphics, and Building Energy Simulation.* In that paper I asserted that microprocessor graphics workstations provided the basis for a new generation of building simulation tools which would overcome the limitations of traditional batch driven programs. Further, I noted that standardized building data interchange formats were urgently needed to promote efficient development of the computerized design industry.

One implementation that was to take advantage of the new microcomputer capabilities was CALPAS4, the successor to Berkeley Solar Group's widely used CALPAS3 hourly thermal simulation model. The 1985 paper included some examples of the graphic input features envisioned for CALPAS4.

Over the nearly four years since the 1985 conference, BSG has invested large amounts of time and money in the development of an integrated, graphically oriented building analysis program. During this project, it became clear that the portions of the program devoted to geometric data management were much more complex than those concerned with analysis. This central graphic input processor was renamed TAKEOFF, since it performs the functions normally done by the user doing manual takeoffs.

The results of the TAKEOFF implementation effort are mixed and provide numerous lessons about developing building analysis software. The purpose of this paper is to describe and assess our experience.

PC DEVELOPMENTS SINCE 1985

The 1985 paper maintained that PC-style microcomputers offered adequate numerical and graphical power for building modeling applications. In the ensuing four years, microcomputer hardware has continued to develop at an astonishing rate. In 1985, 4.77 mHz 8088 based systems were the norm. Today, such machines are approaching obsolete; systems built around the 80286 and 80386 with high clock speeds are taking their place.

The 80386 is particularly significant for two reasons. First, it offers memory management capabilities equal to those used in the DEC VAX and other minicomputers. Second, the 80387 numerical coprocessor is extremely fast. An 80386/80387 PC is significantly faster than minicomputers which were the standard a few years ago.

In 1985, the overwhelming dominance of PC style microcomputers was not a foregone conclusion. Predictions were made at the 1985 conference that workstation implementations of VAX or similar architectures (with unsegmented virtual memory and good numerical capabilities) would become the primary platforms for building simulation. Now, however, the 80386 generation has eliminated any doubt about the dominance and adequacy of PCs for building modeling.

As has been widely observed, software lags ever farther behind the rapidly improving hardware. Both operating system and development software are currently major impediments to efficient implementation of building applications. Despite the advanced capabilities of the 80386 processor, most systems use the DOS operating system, which limits memory to 640 k and allows extremely undisciplined behavior on the part of application programs. These attributes make software development more difficult and costly. New operating systems such as OS/2

promise to improve this situation, but they are slow coming into wide use.

Graphics applications require both numerical power and a large amount of memory. The industry is at a point where the most popular type of personal computer clearly has enough of both for building related applications. System and application software must now catch up.

TAKEOFF

The objective of the TAKEOFF project was to develop an interactive, integrated, graphically oriented program for building energy analysis, equipment sizing, and other applications. Drawing the building promised to be faster and more accurate than describing it with a text based input language, as was discussed in the 1985 paper.

The transformation of this idea into a viable computer program has proven to be prodigiously difficult. After about 25,000 person hours of development, we have completed what can be regarded as a prototype version of TAKEOFF. It has been marketed on a limited basis, but user experience with it indicates that it has too many design and implementation errors to be generally useful. On the other hand, the program has clearly established that graphic input is both fast and accurate; users who have found ways around TAKEOFF's problems refuse to return to programs requiring manual input.

The essence of TAKEOFF is a simple sketching process closely integrated with a database of building components. To input a building, the floor plans are drawn using cursor key commands, as shown in Figure 1. As the building is drawn, records are immediately created in the associated database, which can be accessed with a single keystroke (see Figure 2). Additional drawing in elevation, section, and roof modes allows assembly of a full, three-dimensional representation of the building, as shown in Figure 3.

Once a building has been drawn, keyboard commands can be used to print lists of building components, make drawings on a printer, or invoke calculations such as energy simulations.

Prior to embarking on the TAKEOFF project, our previous programming experience was primarily in the realm of batch driven, non-graphic programs such as CALPAS3. The complexity introduced when moving to a program like TAKEOFF is enormous, because it is both interactive and graphic.

INTERACTIVE EDITING

Batch programs benefit from the "once through" nature of their input. Building applications generally require some type of flexible memory management, since the number of any type of component contained in a given building can vary so much. In batch programs, this variation is easily accommodated with a "fill it up from the bottom" approach, in which data associated with each component is added at the beginning of free memory. No components will ever be deleted, so more complex memory management schemes are not required. At worst, some forward references (from one component to another which has not yet been input) will have to be resolved after the input file is read.

In an interactive environment, where user editing is allowed, two things are different. First, any building component can be deleted at any time. This means that a full dynamic memory management system is required. Second, because user input can arrive in any order, the coherence of the data structure cannot be relied upon at any particular moment. (For instance, a wall might refer to a construction which is not currently defined.)

Memory. The solution to the first problem is harder than it looks. TAKEOFF is implemented in C, so it would appear that using the standard malloc/free library functions would provide the required memory management. However, malloc and free do not provide any compaction mechanism and our experience (learned the hard way) is that serious fragmentation occurs relatively quickly in interactive applications. TAKEOFF's memory management scheme does not provide for compaction and the program often fails due to lack of contiguous memory.

Data consistency. During TAKEOFF development, we experimented with several approaches to the problem of inconsistent data caused by the sequence of user editing. It is straightforward to handle references *from* an object being edited, but difficult to deal with references *to* the object. For example, if a wall being altered refers to a construction type, the program can rapidly verify that the construction type being referenced is in fact defined. On the other hand, if the construction type is being altered, locating all walls referring to it requires a search or maintaining a more complex data structure. This could easily have serious performance implications. Ultimately, in TAKEOFF we abandoned any effort to check references to data being edited. Errors caused by editing mistakes of this type are detected only when a calculation is initiated, making the program more batch style in nature.

GRAPHICS

The graphic interface introduces another group of technical problems.

Numerical precision. The implementation of the geometric calculations involved in graphics requires rigorous and consistent attention to issues related to numerical accuracy. There is a tendency for inexperienced programmers to use theoretically correct techniques and to test their code using simple cases, which generally work correctly. Subtle bugs can lurk in programming of this type, since only specific situations will produce numerical inaccuracy. Note also that dealing with numerical inaccuracy requires additional computation -- when in theory a test for equality is adequate, the actual code must test for the absolute value of the difference being small.

Data management. The graphics interface involves even more demanding dynamic data management requirements than does the interactive nature of the program. A full three dimensional representation of a building can involve hundreds of surfaces, some with "subareas" such as windows or doors. Each surface has three or more points; thousands of points are required. All of this data must be dynamically managed.

Bit mapped display. Another memory difficulty is introduced by modern bit mapped displays. High resolution displays, such as the EGA and VGA, require over 100 k of video memory. When a context change occurs in the program (for instance, if the user invokes Help), the display must be saved for later restoration. (Alternatively, the program can maintain sufficient state information so the display can be regenerated; this approach is both difficult to implement and slower.) Obtaining large contiguous blocks of memory is impossible, so in TAKEOFF we limit screen resolution to 640 x 200 pending a better solution.

Hidden surface elimination. A program like TAKEOFF must be capable of displaying a building with hidden surfaces eliminated. While hidden surface elimination is a standard computer graphics technique, implementing it so it operates quickly and reliably is not straightforward.

Printers. No standards exist for the format of graphic data being transmitted to dot matrix and laser printers. It is a large task to support even a popular subset of them.

PROGRAM DESIGN

While the above sections deal with some of the technical issues which made the implementation of TAKEOFF difficult, the overwhelming problems were those of program design. The general organization of TAKEOFF is derived from standard architectural drawings -- data is manipulated in plan, elevation, and section modes. This organization is accessible to users and seems natural, but has some important consequences.

Plan-by-plan input. TAKEOFF is organized around the idea of the "plan", which is a contiguous area of a building at a single level. This allows building drawings to be entered simply from drawings. However, the plan-by-plan scheme introduces what we term the adjacency problem. In a split level design, for instance, separate plans must be used to enter areas of the building which have different floor levels. The program must then analyze to separate plans and deduce which portions are actually adjacent (and would be modeled with no heat transfer) and which portions actually face ambient conditions. These adjacency calculations turn out to be extremely complex in many situations, involving searching for and intersecting hundreds of polygons.

Three-dimensional editing. The aspects of TAKEOFF which are the most successful are those which involve two-dimensional data, such as plan and elevation editing. Three-dimensional schemes, such as used in roof and section manipulation, are less well worked out. This is an obvious consequence of the two-dimensional nature of the screen.

CONCLUSION

Our experience with TAKEOFF has clearly demonstrated that formidable design and technical problems must be solved in order to successfully implement an interactive graphic building analysis program. However, it is also clear that the effort is worth while. Working with TAKEOFF is much faster, more accurate and generally more fun than working with traditional programs. Whether it is ever completed to the point of commercial viability or not, the direction shown by TAKEOFF is productive and will be the standard in coming years.

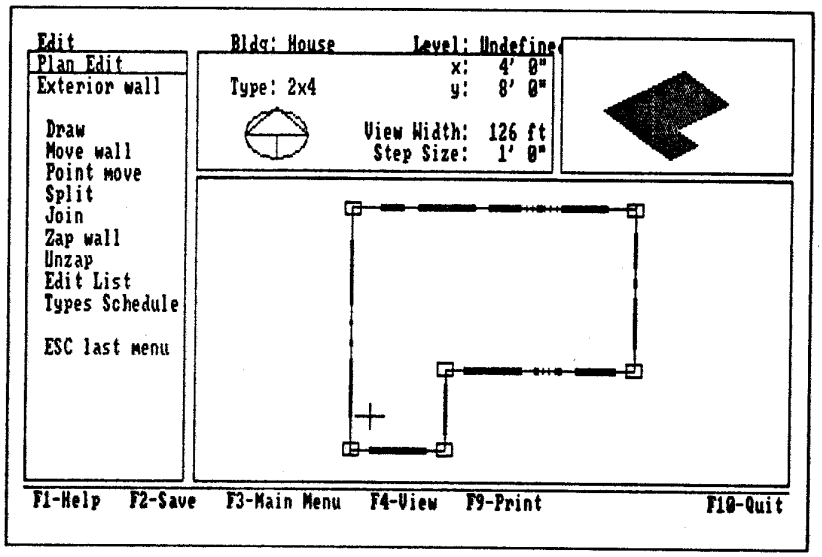

Figure 1. TAKEOFF Plan Edit

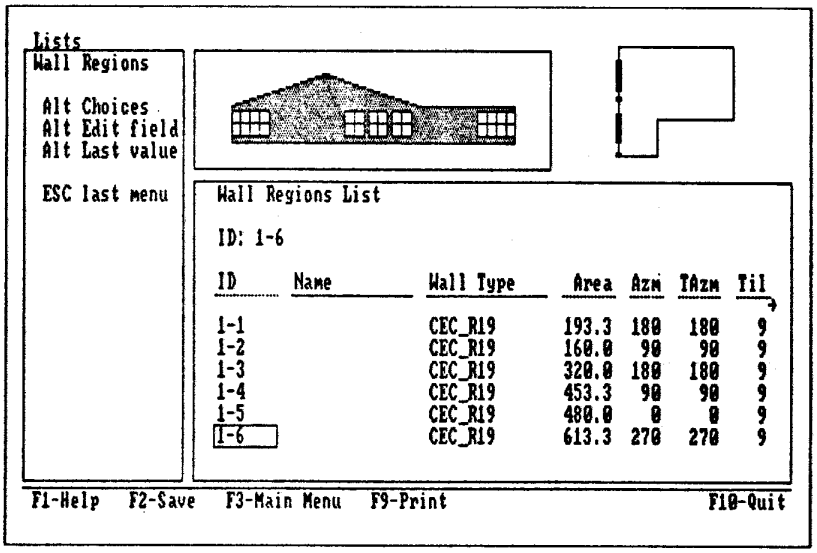

Figure 2. Integration of graphic data with database of building components

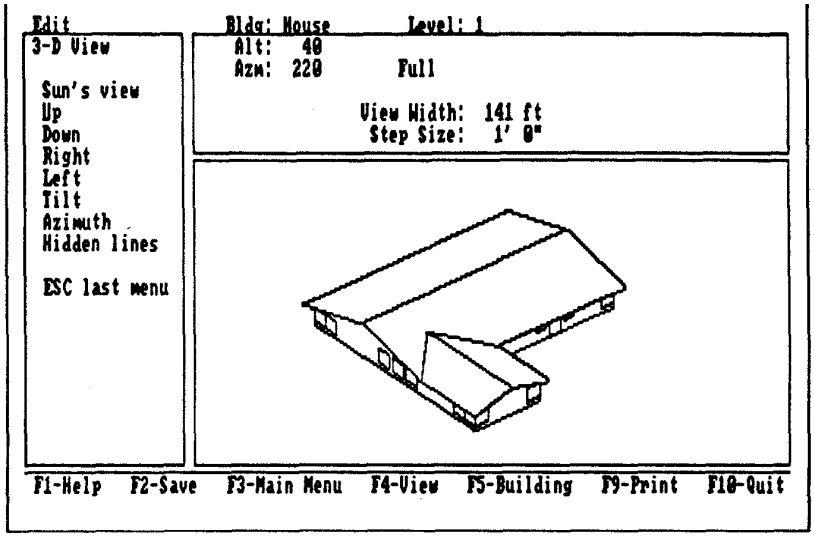

Figure 3. Full three-dimensional view of building with hidden lines removed