BLAST INPUT PREPROCESSORS

Linda K. Lawrie

U.S. Army Corps of Engineers
Construction Engineering Research Laboratory
Champaign, Illinois

ABSTRACT - Since the release of version 3.0 in 1981, development efforts for the Building Loads Analysis and System Thermodynamics (BLAST) system of computer programs have focused on enhancing user-friendliness. Two input preprocessors have been developed at and are supported by the US Army Construction Engineering Research Laboratory (USA-CERL). Both of these input preprocessors were designed to interact with the design engineer and, as an end result, produce a BLAST input model.

Each of the input preprocessors is described in detail. Experience of using these methods as input devices to the BLAST energy analysis computer program is related by illustrative examples from actual design applications.

Since each of the preprocessors allows the user to interact with an actual building model, this method of using application programs could serve as a general prototype for future input preprocessor development.

BACKGROUND

The Building Loads Analysis and System Thermodynamics (BLAST) Energy Analysis Simulation Program was initially released for use in February 1977. Besides the extensive research and development of the algorithms implementing the thermodynamic modeling required for building energy analysis, the user interface to the simulation program used a technique new to engineering application programs. This technique, long familiar to computer scientists, employed an "English-like", block structured language created by using formal grammar specifications. The BLAST user, as do other energy analysis program users, specifies the model of the building to the simulation program. The program interprets this model, adds any additional information that the user has omitted or that is referenced in run time libraries, and calculates the energy consumption based on the entire building model. This process, describing the entire input to a program and then having the program execute according to that input, is known as a batch oriented process typical of programs requiring either a large amount of computer system resources or needing complete information about what the program is to do before actual program execution. The batch process suffices for most energy analysis modeling and allows the user to be doing other work during actual program execution.

User attitudes to computer program interfaces have more to do with user motivation and necessity to use the program than with the actual program interface. Therefore, software developers must try to understand this aspect of the user in order to develop usable programs. This understanding has often been ignored by researchers in innovative efforts to advance a particular field. However, software developers may be more attuned to the idea of listening and understanding users because software, in order to be successful, must be used.

Another element in user attitudes is the perceived maintenance and support to the software. Software usage is dynamic. If the users' needs for the software

change, enhancements / changes must occur. If errors are found in the software, these errors must be corrected. If such support is not provided, the software will not be used. In this light, the US Army Corps of Engineers has sponsored a BLAST Support Office to provide software and user support of the BLAST system to its users.

PREPROCESSOR DESCRIPTIONS

Several efforts helped early BLAST users adapt to the new inputting technique. An input booklet, simultaneous with BLAST version 2.0 release in mid 1979 (1,2), provided the user with templates designed around the BLAST input vocabulary. Other input preprocessors were developed as interactive computer programs under the auspices of the Air Force and Navy as well as private concerns. In each case, the motivation for creating the preprocessing technique was to enhance (lessen) the learning curve time for the BLAST user. This early (and to date) BLAST user often is a totally novice computer user. Also, the person is usually performing the energy analysis as an "extra" part of the design effort (in addition to the usual job of mechanical engineer or architect) and, though he or she may view energy analysis as an important part of the design, has little or no time or money allotted to the assignment. This emphasis on energy analysis in design stresses that the designer must accomplish the energy analysis requirements in the least amount of time. Because of this stress on fast results, the designer will seldom have time to extensively research, or in fact read at all, the user documentation for a computer program to accomplish the energy analysis. In fact, design time pressures allow little time to meet any predefined requirements let alone be creative with determining the optimal energy model for the facility.

Recent emphasis, since release of BLAST version 3.0 in 1981 (3), by the developers / maintainers has been on understanding the energy analysis users' needs and improving the user interaction with the BLAST computer program. In this vein, two input preprocessors, text oriented and graphic oriented, have been developed to create the BLAST input models which were previously created on cards, using text editors, or using another kind of preprocessor. These preprocessors create their own stored building models and can be termed "indirect" preprocessors. After initial entry of a building or building portion, each preprocessor can be used in an "editing" fashion to further refine the model. Outputs from these preprocessors are BLAST input files in the standard BLAST input language format. This method of preprocessor development allows more flexibility to the user and provides more user feedback via the BLAST input file than a direct interface to the BLAST simulation portions.

Text Preprocessor

The BLAST Text Input Preprocessor (BTEXT) was released for use in March 1984 (4). It was originally used with a week long introductory BLAST course sponsored by the U.S. Army Corps of Engineers. The students in this course were Corps designers with typically little or no computer experience. Versions of this course presented previous to March 1984 had required the participants to enter several building models by typing in the entire BLAST syntactical requirements. In addition to novice computer users, the BLAST course teachers uncovered not unexpected poor typists (hunt and peck, slow, inexperienced), but found enthusiastic attempts to use the program even so. The March 1984 experience showed the possible worth of implementing BTEXT as a general BLAST input preprocessor. Even with expected errors in having a program generally used for the first time, the students were able to complete the course workshops with little help (typing and syntax interpretation) from the instructors. Thus, the learning could be concentrated on the real issues -- creating energy analysis building models and learning BLAST intricacies.

The BTEXT preprocessor interacts with the user in a menu-driven fashion to create a building model. Once the user has selected a menu item, the program usually prompts the user for further typed entries. The BTEXT building model can be described over several computer sessions. After each session, a BTEXT created model of the building

is stored on-line on the computer. Once the building model is created to the user's satisfaction, a BLAST input model can be created. Then, if any further changes need to be done, the user may edit the BLAST input file using a system text editor or, if no changes are necessary, the user may execute BLAST with the model produced by BTEXT. With the former option, the familiar BLAST user can tailor the BTEXT produced model to meet his / her needs whereas the latter option allows both familiar and novice BLAST users to produce syntactically acceptable BLAST models with relative ease.

Other aspects of the perceived user requirements have been implicit in the BTEXT development. Because the user does not typically want to read the entire BLAST documentation in detail, BTEXT allows the user to obtain extra information to any BTEXT request. One type of information tells the user exactly how the BTEXT request relates to the BLAST program; direct referrals to the BLAST documentation are often supplied as well as brief explanations. Another idea behind BTEXT development was to produce, for the user, the maximum BLAST input from the minimum information requested of the user. Thus, BTEXT becomes an interpreter of many parts of the BLAST documentation for the user.

For example, to specify an air handling system to BTEXT, the user need only pick the BLAST system type, the thermal zones served by the system, and minimal types of information applicable to the selected system (e.g. cooling coil type, zone supply air volumes). From this information, BTEXT produces a BLAST input model that contains all the air handling system parameters that apply to the selected system. Because BLAST has many different air handling systems, different parameters apply to different systems. BTEXT knows which parameters are supposed to apply to the selected system and, thus, tailors the BLAST input to only the parameters needed. The user does not have to extensively search through the BLAST documentation for the applicable system parameters; he or she need only research the parameters displayed in the BTEXT produced input file to determine actual values for the building. These values may be changed, if necessary, by using the computer system text editor.

BTEXT allows this minimal entry scheme in both the air handling system and central plant equipment portions of the BLAST input model. The BTEXT entries for the other parts of the BLAST input (e.g. library modifications and thermal zoning) are more completely controlled by the user. Virtually any parameter of the building geometry and internal load specifications may be entered by the BTEXT user. Future enhancements in BTEXT will implement similar control over the air handling system and central plant portions of the BLAST input.

BTEXT is designed to meet the needs of a range of users, from novice to more sophisticated. The user can execute BTEXT from a normal, ASCII interactive terminal on a variety of computer hosts (mainframe or mini). Plans are underway to implement BTEXT on a standard business sized personal microcomputer (IBM PC or equivalent).

Graphic Preprocessor

Another indirect BLAST input preprocessor has employed a more direct graphic approach to a geometric building model. This preprocessor has been developed as a part of the Computer Aided Engineering and Architectural Design System (CAEADS) work at the US Army Corps of Engineers Construction Engineering Research Laboratory (USA-CERL). The CAEADS system approaches the building design process in a multidisciplinary fashion. Each of the disciplines involved in the design effort introduces some new information to the total building design; yet each discipline needs information from others to address his or her portion of the design or analysis of the design. The CAEADS system has attempted to incorporate this data into a single design data base for a facility. To date, stress in the CAEADS software development has been placed on the concept, or 35%, portion of the design. Basic to the CAEADS system is to use the common data base to afford analysis by various designer disciplines. These analysis programs, such as structural analysis or energy analysis, are viewed as "off

the shelf" software packages being driven by the CAEADS users and data base information. BLAST was chosen as the prototype program for energy analysis from the CAEADS data base.

This BLAST input preprocessor uses two of the CAEADS modules: ARCH and ENERGY (5,6). The BLAST input preprocessor in this case transfers building geometric, material, and hvac specific data from the CAEADS data base of design information to the BLAST input file. Two kinds of entries are accomplished using the CAEADS modules before executing BLAST. First, the building floor plan, with room, door, and window placement as well as material selection, has been laid out by the project architectural section. Secondly, the energy engineer has overlaid on this plan the thermal zoning appropriate for the room controls and hvac equipment selections to be studied. Standard graphic (Tektronix) terminals are used as designer sketchpads to enter the information to the CAEADS data base. Again, the user picks from a menu of choices, but more data is entered by "pointing and selecting" than by answering preprocessor questions. The geometrical information, such as length of walls, size of windows and doors, are an inherent part of the user's entry whereas attributes of the various design elements, such as thickness of walls or wall composition, can be assigned during initial entry or reassigned at any later time.

This method of creating the BLAST input model offers immense flexibility to the designer. A major portion, and major stumbling block to users, of the BLAST input specification is the geometric description of the building. This geometric description is automatically (and correctly) created by the CAEADS / BLAST interface module. Thus, the designer can study, for very little investment of time, as many different room, thermal zone, building orientations as are warranted.

The CAEADS based input preprocessor for BLAST was initially released in September 1983. Since then, based on user suggestions and continuing CAEADS development, further refinements in the produced BLAST input model have been made. After undergoing alpha testing in one US Army Corps of Engineers District, this preprocessor is now a key part of a five Corps District CAEADS beta test. (Alpha tests are one on one tests between software developers and potential users. These potential users become an essential part of the development procedure. Beta tests are tests of software that shows applicability during alpha tests, but software development indicates further refinement may be necessary for wide usage.)

PREPROCESSOR EXPERIENCES

The learning process of understanding the energy analysis users' needs continues. Certainly, the user's perception of creating input models has changed since BLAST's introduction. Several activities, including the development of the input preprocessors, has probably helped the perception of BLAST in the user community. Efforts of getting the users more involved in the management of and the enhancements to BLAST have been successful. BLAST has been converted to be portable in the virtual memory minicomputer environments that are now popular, and within budgetary scopes, with architectural / engineering firms. Computer sophistication in the general populous, too, has increased and thus, less resistance to computer programs and computer terminology can be expected. Additional BLAST enhancements have included an executive summary report (7) and reports emphasizing easier interpretation of energy analysis simulation results. The key to assisting the user is determining what the user needs from the software and then trying to satisfy those needs.

In the case of the input preprocessors, the user needed to expedite the time required to obtain results from the energy analysis simulation. To meet this user requirement, an obvious starting point was to speed the learning curve for the user or to assist the user in creating an input model more quickly. Once this irritation is past, he or she can get on to the true issues -- determining the impacts of the design.

Though it's hard to quantify the benefits from the two input

preprocessors, several indicators are evident. In BLAST courses prior to the introduction of BTEXT, the majority of student time was spent typing the BLAST syntax and then correcting the inevitable typing mistakes. This time, also, was spent primarily in creating the BLAST geometrical description. Little or no time was eventually spent in being able to concentrate on using the program or the energy analysis results.

In contrast, during the training course which introduced BTEXT, typing remained a factor; however, now the user received immediate feedback when an error was entered. BTEXT checks certain entries to make sure they will yield valid BLAST input. For example, specifying a window that is larger than its wall is flagged as an error by BTEXT and the user is required to reenter the correct value. Thus, the input preprocessor can give immediate response to the user on many kinds of errors. This immediacy helps the user by speaking to the subject at hand rather than waiting until BLAST has processed the incorrect input and the user has lost track of what the input should be. Thus, even at its introduction, BTEXT enabled the new users to concentrate on getting the job of energy analysis done rather than being frustrated in typing or other nonessential elements of the task.

In another recent example, an architectural / engineering firm that had never used BLAST before was awarded a Corps contract to perform the design energy analyses. They had no time for a formal training course and would have to approach the analyses armed primarily with the BLAST user documentation and assistance from the BLAST Support Office. The firm was given access to BTEXT. They were able to complete an initial analysis within a week and the entire analyses within a month. This example contrasts previous experiences where a firm might have only accomplished the reading of the documentation within the month and still not have been able to create BLAST input models.

Equally, the graphic preprocessor enables the user to quickly create several BLAST input models from the same building floor plan. The user can also quickly create studies of several floor plans in early planning stages of the building. This ability allows the designer to make an informed energy decision about a plan.

Several experiences illustrate the power of using the graphic preprocessor to create the BLAST input models. Both examples involve in house designs within the Corps District offices. Though these are isolated cases, they do not seem unusual.

The first example illustrates the ease of using a graphic means to enter geometric data. A Corps District mechanical engineer was given a tutorial introduction to entering a building floor plan by entering a typical administration and classroom facility (existing) that needed to be modeled using BLAST. The engineer was led step by step in entering the floor plan; this took about four hours. On the following day, the engineer entered an alternative floor plan for the administration and classroom facility. This entry took only two hours. The engineer, an experienced BLAST user, estimated that preparing an equivalent BLAST input file would have taken at least two days. In addition, the graphic method allows the user to easily use BLAST in a room by room analysis and later create a less complicated block thermal zone model; the CAEADS interface properly accounts for all internal surfaces in the resultant BLAST input model.

In another example, the CAEADS system has been undergoing alpha testing at a Corps District office since January 1983. Using a multidisciplinary approach to building design, the architects have been responsible for entering the building floor plans and assigning the various building attributes. The energy engineers draw from this information to create the BLAST input models. In this approach, contrary to the previous example, the engineer is obtaining the geometric model at very low cost. Statistics produced from the alpha test reinforce this notion; in addition, the engineers feel enough time is allowed to study multiple alternatives resulting in better quality designs in all aspects (not just energy). Energy analysis becomes part of the normal design

process with CAEADS, not a "burdening" extra task.

The major drawback of each preprocessor remains the inability to create the entire energy analysis model within the preprocessor itself. The user usually must edit the BLAST input file that is created and modify as necessary. In this respect, BTEXT creates a better geometric model than does the graphic module. Since the CAEADS modules being used do not yet implement the full three dimensional model of the building, the information provided to the BLAST interface is incomplete in the actual size of windows (width is correct, but height is unknown). Neither preprocessor gives the user any power over changing the parameters of the air handling system or central plant. Both produce similar "complete" air handling system input models, as described above.

Efforts continue in both preprocessor developments. At user suggestion, both BLAST models produced are becoming more completely controlled within the preprocessor rather than relying on modifying the model with a text editor. Future enhancements will continue in this vein with increased emphasis on checking the entries within the preprocessor in order to provide the immediate feedback capability. Another area that may be addressed would be the interfacing of the two preprocessor data bases such that BTEXT could read / write an equivalent CAEADS data base and vice versa.

CONCLUSIONS

The energy analysis tool user needs to assure costs associated with performing energy analyses do not consume an inordinate portion of the design budget. Because the user may, through desire or requirements, study several energy design alternatives, this process needs to be efficiently accomplished. When using any computer program that must interpret an entire model before actual calculations - typical for energy analysis simulations - getting the best model before each simulation execution is important.

The two BLAST input preprocessors, text based and graphic oriented, allow the user to interact with the building model before actual BLAST simulations are done. In a more global view, these two preprocessors advance the idea of creating input for an application program by storing models of the facility in their own format and, as a by product, creating the BLAST syntax input model.

Thus, in addition to making it easier for a user to execute the BLAST energy analysis program, these preprocessors can also be viewed as prototype developments in creating building models for any energy analysis simulation program. For example, a BTEXT created building model could be used to execute various programs; thereby the user would be able to use a BTEXT - like preprocessor to create a standard building model to be used in multiple application programs (i.e. BLAST, DOE2, TRACE). Continued development of preprocessors with this emphasis could be generally beneficial to energy analysis simulation field.

REFERENCES

1. D. C. Hittle, The Building Loads Analysis and System Thermodynamics (BLAST) Program, Version 2.0, Volumes I and II, Technical Report E-153 (U.S. Army Construction Engineering Research Laboratory [USA-CERL], 1979)

2. E. Sowell, The Building Loads Analysis and System Thermodynamics (BLAST) Program, Version 2.0: Input Booklet, Technical Report E-154 (U.S. Army Construction Engineering Research Laboratory [USA-CERL], 1979)

3. D. Herron, G. Walton, and L. Lawrie, The Building Loads Analysis and System Thermodynamics (BLAST) Program Users Manual -- Volume I Supplement (Version 3.0), Technical Report E-171 (U.S. Army Construction Engineering Research Laboratory [USA-CERL], 1981)

4. The Building Loads Analysis and System Thermodynamics (BLAST) Text Input Preprocessor (BTEXT) Users Manual, (Draft)

5. ARCH - Architectural Layout Module of CAEADS Users Manual, (Draft)

6. ENERGY - Energy Interface Module of CAEADS Users Manual, (Draft)

7. J. Amber, D. Leverenz, and D. Herron, Automated Building Design Review Using BLAST, Technical Report E-85/03 (U.S. Army Construction Engineering Research Laboratory [USA-CERL], 1985).

ACKNOWLEDGEMENTS

The input preprocessors described were programmed by Larry Owens, Doug Hilmes and the CAEADS development team at USA-CERL. Without their efforts, the successes being enjoyed could not have happened.