The correct citation for the paper is:

Pedersen, C.O., D.E. Fisher, R.J. Liesen, R.K. Strand. 2003. ASHRAE Toolkit for Building Load Calculations. *ASHRAE Transactions*. 109(1) :583-589.

Reprinted by permission from ASHRAE Transactions (Vol. #109, Part 1, pp. 583-589).

CH-03-9-4 (RP-987)

ASHRAE Toolkit for Building Load Calculations

Curtis O. Pedersen, Ph.D.
Fellow ASHRAE

Daniel E. Fisher, Ph.D., P.E.
Member ASHRAE

Richard J. Liesen, Ph.D.
Associate Member ASHRAE

Richard K. Strand, Ph.D.
Member ASHRAE

ABSTRACT

The loads toolkit represents the third element of TC 4.7's toolkit trilogy. Following the primary and secondary system toolkits, it represents an important step in making computer technology available to a broader segment of ASHRAE engineers. One common characteristic of engineers is that they like to see and know how things work. This is why they generally are skeptical of black box calculation programs and why the toolkits are important. They make the underlying physics of the processes used in load and energy calculation procedures open and available for inspection and understanding.

This paper describes the toolkit components and the their organization on the CD format being distributed by ASHRAE.

INTRODUCTION

The famous British author Len Deighton is known for writing his mystery thrillers as trilogies. TRP987 represents the third element of TC 4.7's toolkit trilogy. This trilogy is a replacement for two earlier publications, Energy Calculations 1 and 2 (ASHRAE 1976,1978). While probably not destined to be as widely read as Deighton's books, it does represent an important step in making computer technology available to a broader segment of ASHRAE engineers. One common characteristic of engineers is that they like to see and know how things work. This is why they generally are skeptical of black box calculation programs and why the toolkits are important. They make the underlying physics of the processes used in load and energy calculation procedures open and available for inspection and understanding. This essentially makes them the opposite of a mystery novel.

It is logical that the load toolkit has been left to be the third member of the TC 4.7 toolkit trilogy. The load calculation toolkit brings together all the parts of the simulation represented by the other two toolkits (ASHRAE 1998; Brandemuehl et al. 1993). The procedures represented in the load calculation usually are the only ones that introduce time dependence into the problem, and they also represent the processes that must reconcile the interactions of system, plant, internal loads, lights, solar, people, and other things all operating together as a system in the broad sense of the word. The routines of the first two toolkits generally can stand alone and produce some useful results by themselves. The routines represented by the load toolkit are much more closely coupled and generally need to interact to produce useful results.

The goal when producing a set of routines for a toolkit should be to use very straightforward procedures that show the physics clearly and do not rely on obscure numerical analysis tricks to produce results. The processes involved in the load calculation procedure are clearly ones for which it is most difficult to separate the physics from the numerical analysis. For example, if an iterative solution technique is used to update the surface temperatures in a thermal zone, the solution is much more stable and robust if the radiant heat exchange among the surfaces is linearized. In this case, the increased stability does not come from the linearization process itself but from the fact that the equation can be made more explicit by factoring out the participating surface temperature. It is a clear case where the physics are obscured for the sake of the numerical method.

The component routines presented in the load toolkit have been written to maximize clarity even if the resulting algorithms are not computationally optimized.

Curtis O. Pedersen is professor emeritus, **Richard J. Liesen** is associate director, Building System Lab, and **Richard K. Strand** is assistant professor, School of Architecture, University of Illinois at Urbana-Champaign. **Daniel E. Fisher** is assistant professor, Oklahoma State University, Stillwater, Okla.

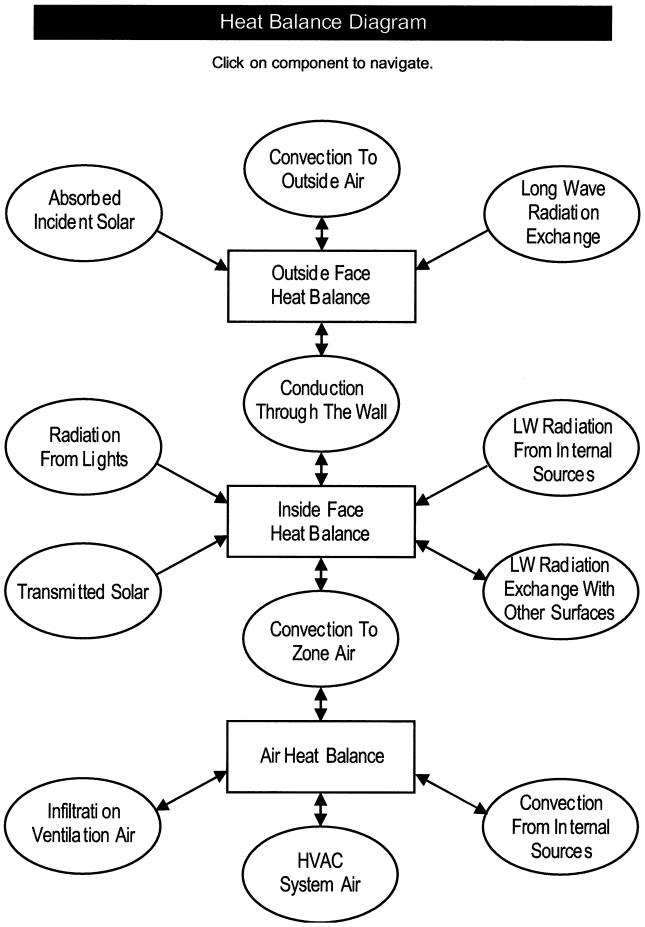

Figure 1 *Nonlinear navigation tool.*

TOOLKIT ARCHITECTURE

Overall Organization

The toolkit CD is organized as a hypertext document and contains both a linear (table of contents type) and a nonlinear navigational tool. The nonlinear navigational tool is shown in Figure 1. When this figure is displayed from the CD, clicking on any of the labeled components will take the user to the documentation, testing, and FORTRAN coding for that subject.

FORTRAN 90

As required by the project work statement, the toolkit is written entirely in FORTRAN 90. The source code adheres to standard FORTRAN 90 and eliminates all obsolete features. Standard FORTRAN 90 refers to the full American National Standard Fortran 90 language as defined in the American National Standard Programming Language Fortran 90, ANSI X3.198-1992, and International Standards Organization Programming Language Fortran, ISO/IEC 1539:1991(E).

The toolkit makes use of many of the new features available in FORTRAN 90. One of the most important is the new *module* program unit. Modules provide a convenient means of packaging related definitions and operations. Modules are the fundamental building block of the toolkit. Multiple modules are assembled to form the various toolkit components.

THE TOOLKIT COMPONENTS

The toolkit components have been prepared as a library of working code modules ready for immediate implementation as stand-alone performance models or as components to be integrated with a complete building simulation program. Component directories are organized into a hierarchical directory structure that associates closely related components in the heat balance diagram. The components and component structure are listed below.

Air Heat Balance

- Infiltration and Ventilation
- Internal Gains

Air Moisture Balance

Conduction Through Building Envelope

- Conduction Transfer Functions (CTF)
- LaPlace CTF

Exterior Heat Balance

- Environment Surface Temperature
- Exterior Convection
- Exterior Long Wave
- Ground Heat Transfer
- Shading
- Sky Temperature Models
- Solar Radiation

Interior Heat Balance

- Interior Convection
- ScriptF Routines
- View Factor
- Zone Longwave Radiation

Zone Heat Balance Model

- Heat Balance Equations

Utilities

- Psychrometrics
- General Utility

The majority of the components include a test driver program designed to exercise the modules of that component. The toolkit adopted the input scheme developed for the EnergyPlus program (Crawley et al. 2000). This involves an input data dictionary (IDD) that describes the format of the input data file (IDF).

The typical component directory contains the following:

- The FORTRAN 90 source files for the component modules and test driver (*.f90)
- The component test driver executable (*.exe)
- An IDD file (ToolkitTest.idd)
- An IDF file (in.idf)
- A test driver output file (TestResults.out)

Within a component there may be several models to accomplish the same purpose. The user can select the desired model by modifying the appropriate USE statement in the code. For example, changing

```
USE ExteriorConvectionMod, ONLY:
CalcHcOut => CalcHcOutByDOE2model
```

to

```
USE ExteriorConvectionMod, ONLY:
CalcHcOut => CalcHcOutByBLASTmodel
```

switches from the DOE-2 model to the BLAST model for calculating the exterior convective heat transfer coefficient.

PROGRAMMING CONVENTIONS

The goals of the toolkit were used to establish programming conventions followed during the component development. They resulted in specific philosophy and format considerations that are explained below.

Code Development Philosophy

The following coding criteria were prioritized in descending order of importance:

- Level 1: Readability, Testability, Maintainability, Robustness, Reliability
- Level 2: Portability, Reusability
- Level 3: Speed, Size

In general, clarity of the toolkit code took precedence over computational efficiency.

Format and Comments

All code is placed in "free-format" as opposed to the fixed format used by FORTRAN 77 and other versions of FORTRAN. In addition, the following guidelines were followed for all free-formatted code in the toolkit:

- An effort was made to confine each line to 80 characters. This allows most code to be seen on a standard size screen and be printed without resorting to micro-fonts or landscape mode.
- Column 6 is not associated with a continuation line in free format. To continue one line onto the next line, an ampersand character "&" is placed at the end of a line to be continued.
- To help visually distinguish between FORTRAN 90 keywords and other code elements such as comments and names, FORTRAN 90 keywords are in all uppercase while other elements are in mixed case.
- Tab characters are not allowed in any source code file.
- Only "!" is valid for indicating comments. Endline comments are allowed and encouraged.

Guidelines from *Code Complete* (McConnell 1993) were followed for endline comments. Several suggestions are repeated here:

- Use endline comments to annotate data declarations.
- Use endline comments for maintenance notes (bug fixes, for example).
- Use endline comments to mark ends of blocks.
- Avoid endline comments that merely repeat the code.
- Avoid endline comments for multiple lines of code. In other words, avoid using a comment at the end of one line that applies to several lines of code.

No lines extend past 132 characters.

Programming Safety

FORTRAN 90 supports many safety features, such as argument checking across procedures, IMPLICIT NONE, and INTENT attributes that enable the compiler to find many programming errors. The following rules were followed for Toolkit code.

- IMPLICIT NONE was used to require all variables to be explicitly declared.
- INTENT was used to specify how a procedure uses a function argument. INTENT can be one of the following:

  IN: The argument is an input to the procedure. It cannot be redefined or become undefined while the procedure runs.

  OUT: The argument is an output of the procedure. The procedure must define the argument before it uses the argument. Any actual argument associated with the argument must be definable. On invocation of the procedure, the argument is always undefined.

  INOUT: The argument can both receive data from and return data to the invoking program unit.

- PUBLIC or PRIVATE were used to specify the access attribute of the data objects within a module.
- The SAVE attribute was used to preserve the value of a variable after the execution of a RETURN or END statement in a procedure.

Naming Conventions

In all of the naming conventions listed below, the limit on the length of names is 31 characters, and spaces were not allowed as valid characters in any name.

Variable Naming Convention. In general, variable names were selected to form identifiable, mnemonic descriptions of the variables. Since the variable names tend to appear in the code much more often than subroutine or module names,

logical abbreviations are thus encouraged. Typically, lengthy words were shortened to between three and five characters to make a logical yet concise name for the various program variables. Common state or process variables may begin with a one-character letter to designate the variable. For example, temperatures begin with a "T." Although some FORTRAN 90 compilers may be case insensitive, both uppercase and lowercase characters are used in the toolkit to form identifiable words within variable names, for example, "ToutsideAir." Variable names are consistent among the toolkit module routines.

Subroutine Naming Convention. Subroutine names were constructed using the verb-predicate rule. Every subroutine performs some action (the "verb") on a particular item or data set (the "predicate"). The subroutine name is thus constructed using the verb-predicate combination to arrive at a unique name for a particular algorithm. The verb is the first part of the name followed by the predicate. For example: CalculateViewFactor.

Module and Source Code File Naming Convention. Since modules typically are associated with objects or data groups, the name that is selected for a module should refer to the object or data grouping. Data-only modules use "Data" in the name after a logical descriptive term or terms. All modules have a suffix "Mod." For example, "ExteriorConvectionMod" and "InputDataMod."

Since the module is the basic program unit (except the toolkit driver main program), source code files use the name of the module as the base name with ".f90" as the file extension.

Component Documentation

The algorithms are presented in an unambiguous form for direct implementation and verification. The publication documents each model, giving mathematical and physical descriptions as well as algorithmic procedures. It is accompanied by a fully debugged and tested module. Each algorithm is clearly written in pure FORTRAN 90. The module library is a set of tested working programs for immediate implementation as stand-alone performance models or as components to be integrated with a complete building simulation system. Illustrative examples of component integration are also included.

Each component in the toolkit is described in a standard format with the following eight elements (similar to the formats used in the secondary HVAC toolkit [Brandemuehl 1993; ASHRAE 629-RP]):

1. General description of the component.
2. Physical and mathematical description of the component model.
3. References to the engineering literature.
4. Description of the algorithm, giving step-by-step calculation procedures.
5. Nomenclature list of FORTRAN and mathematical variable names.
6. Application example to illustrate use of the algorithm for a typical analysis.
7. FORTRAN 90 module statements of the model with full documentation.
8. Test input data and the resulting output data to illustrate model application.

Module Template

```
MODULE <module_name
! Module containing the routines dealing with the
<module_name
! MODULE INFORMATION:
! AUTHOR <author>
! DATE WRITTEN <date_written
! MODIFIED na
! RE-ENGINEERED na
! PURPOSE OF THIS MODULE:
! Needs description
! METHODOLOGY EMPLOYED:
! Needs description, as appropriate.
! REFERENCES: none
! OTHER NOTES: none
! USE STATEMENTS:
! Use statements for data only modules
! Use statements for access to subroutines in other modules
IMPLICIT NONE! Enforce explicit typing of all variables
PRIVATE! Everything private unless explicitly made public
!MODULE PARAMETER DEFINITIONS
! na
! DERIVED TYPE DEFINITIONS
!MODULE VARIABLE DECLARATIONS:
!SUBROUTINE SPECIFICATIONS FOR MODULE
<module_name
CONTAINS
END MODULE <module_name
```

Subroutine Template

```
SUBROUTINE
! SUBROUTINE INFORMATION:
! AUTHOR <author>
! DATE WRITTEN <date_written
! MODIFIED na
! RE-ENGINEERED na
! PURPOSE OF THIS SUBROUTINE:
! This subroutine needs a description.
! METHODOLOGY EMPLOYED:
! Needs description, as appropriate.
! REFERENCES:
! na
! USE STATEMENTS:
! na
IMPLICIT NONE! Enforce explicit typing of all variables in
this routine
! SUBROUTINE ARGUMENT DEFINITIONS:
! na
! SUBROUTINE PARAMETER DEFINITIONS:
! na
```

```
! INTERFACE BLOCK SPECIFICATIONS
! na
! DERIVED TYPE DEFINITIONS
! na
! SUBROUTINE LOCAL VARIABLE DECLARATIONS:
! na
RETURN
END SUBROUTINE
```

Example Subroutine (Partial)

```
SUBROUTINE CalculateHcInVentilatedRooms(Hour, HcIn,
TsIn, Tref, Tilt)

!** PURPOSE OF THIS SUBROUTINE:
    ! To calculates interior surface convective heat transfer
coefficients

    IMPLICIT NONE! Enforce explicit typing of all variables in this routine

!** SUBROUTINE ARGUMENT DECLARATIONS:
    LOGICAL            :: GetInput = .TRUE.
    REAL, INTENT (OUT)  :: HcIn ! Convective heat transfer coefficient of inside
                                ! surface, W/(m^2 C)
    INTEGER, INTENT (IN) :: Hour ! current simulation hour
    REAL, INTENT (IN)    :: TsIn ! Zone inside surface temperature
    REAL, INTENT (IN)    :: Tilt ! The tilt angle of the surface, in degree unit
    REAL, INTENT (INOUT) :: Tref ! The reference temperature (Zone air), C.

    !** SUBROUTINE INTERNAL PARAMETERS:
    REAL, DIMENSION(24)  :: ACH     ! Room ventilative flow rate in air changes per hour
    REAL, DIMENSION(24)  :: DeckTemp ! The cold deck temperature

    !** FLOW:
        !Get the input
      IF (GetInput) THEN
        CALL GetConvectionInfo(DeckTemp,ACH)
        GetInput = .FALSE.
      END IF
```

USING THE TOOLKIT COMPONENTS

Sample Zone Comparison

While the production of a working program was not the purpose of this project, it was necessary to test the components together in a sample zone calculation. Chapter 13 of the toolkit document contains a working version of a zone calculation using a steady periodic solution scheme to solve for the surface temperatures and the zone air temperature. The sample also has incorporated a "system capacity" scheme to show how the zone temperature drifts if the system capacity is not sufficient to maintain the setpoint temperature. If the toolkit components are used in a loads/energy calculation application, this capability should be expanded to include a more complex system model.

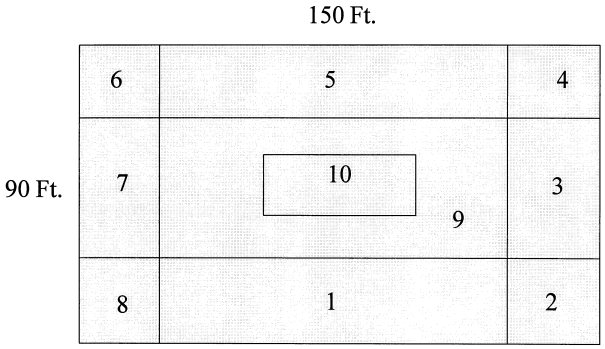

Figure 2 *TC 4.1 building floor plan.*

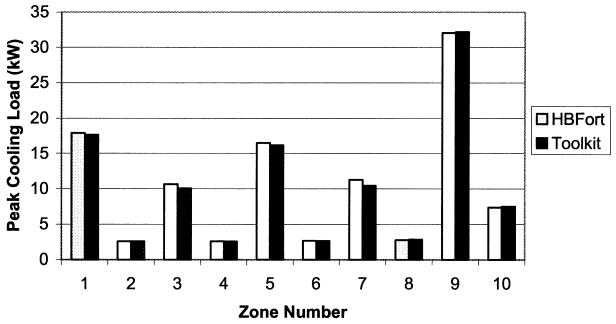

Figure 3 *Results, July cooling load comparison.*

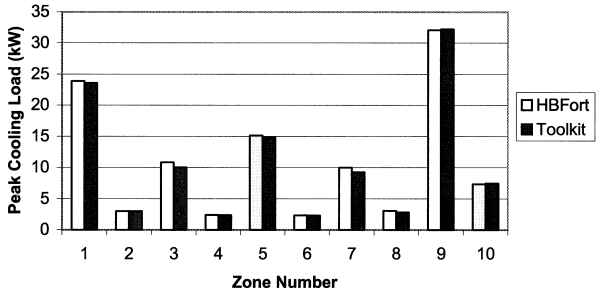

Figure 4 *Results, October cooling load comparison.*

The zone model calculation presented in the toolkit was compared against the heat balance program called HBFort that is distributed with ASHRAE's *Load Calculation Principles* (Pedersen et al. 1997, 1998). The example used was a load calculation method comparison used by TC 4.1. Several different load calculation programs have been used on this example so it was used to validate the accuracy of the toolkit sample zone model.

It is a ten-zone building located in Los Angeles, California. The floor plan of the building is shown in Figure 2.

The comparative results are shown in Figures 3 and 4.

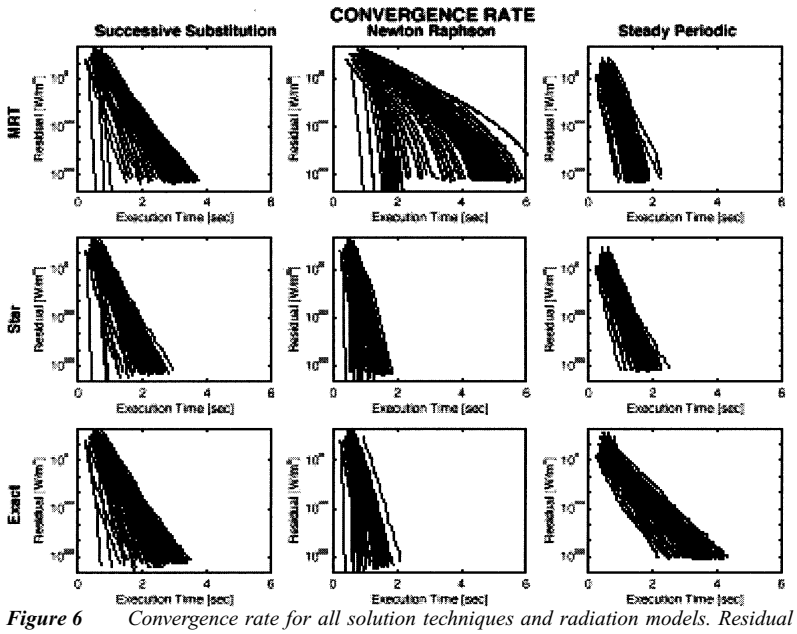

Figure 6 *Convergence rate for all solution techniques and radiation models. Residual [W/m3] vs. elapsed time [sec.] each time represents a single run. Not y-axis is Log scale.*

Figure 5 *Toolkit example with different solution schemes and radiation models (Asmundsson 2000).*

Comparing Solution Techniques and Models

The toolkit components were used to compare solution techniques and internal radiation models under the same conditions (Asmundsson 2000). Figure 6 from this comparison is shown in Figure 5. This figure compares three internal long wave length radiation models, the MRT method, the Star method, and the basic radiosity calculation method (called exact) for a large number of cases. The comparison also shows the effect of using successive substitution, Newton-Raphson, and a special steady periodic solution technique to solve the heat balance equations to various levels of convergence. The convergence is specified as the sum of the residuals of the heat balance equations. While the purpose of including this example here is not to draw conclusions, it is interesting to note that the successive substitution method, the least sophisticated method, performs quite well for all radiation models. Asmundsson's complete thesis is on the CD.

CONCLUSIONS

The load toolkit's components provide a valuable resource toward making the heat balance load calculation procedure more readily available to ASHRAE members. The heat balance procedure has been adopted by TC 4.1, Load Calculations, and is presented in the *2001 ASHRAE Handbook—Fundamentals* as the preferred method. This toolkit will hopefully assist application developers to incorporate the method.

ACKNOWLEDGMENTS

The toolkit is the result of ASHRAE 987-RP, sponsored by TC 4.7, Energy Calculations. The authors are grateful to ASHRAE for the support and appreciate the special efforts to review and comment on the components by the monitoring committee, D. B. Crawley, Chair.

REFERENCES

ASHRAE. 1976. *Energy Calculations 1—Procedures for Determining Heating and Cooling Loads for Computerized Energy Calculations, Algorithms for Building Heat Transfer Subroutines*. Atlanta: American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.

ASHRAE. 1978. Energy Calculations 2—Procedures for Simulating the Performance of Components and Systems for Energy Calculations. Atlanta: American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.

ASHRAE. 1998. A Toolkit for Primary HVAC System Energy Calculations, 665-RP Final Report. Atlanta: American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.

Asmundsson, Jakob. 2000. A study of solution techniques and process models for heat balance based thermal load calculations. M.S. Thesis, Department of Mechanical and Industrial Engineering, University of Illinois, Urbana.

Brandemuehl, M.J., S. Gabel, and I. Andersen. 1993. A toolkit for secondary HVAC system energy calculations (629-RP). Atlanta: American Society of Heating, Refrigerating, and Air-Conditioning Engineers, Inc.

Crawley, D.B., L.K. Lawrie, C.O. Pedersen, and F.C. Winkelmann. 2000. EnergyPlus: Energy Simulation Program. *ASHRAE Journal* 42(4): 49-56.

McConnell, S. 1993. *Code Complete*. Redmond, Washington: Microsoft Press.

Pedersen, C.O., D.E. Fisher, and R.J. Liesen. 1997. Development of a heat balance procedure for calculating cooling loads. *ASHRAE Transactions* 103(2): 459-468.

Pedersen, C.O., D.E. Fisher, J.D. Spitler, and R.J. Liesen. 1998. *Cooling and Heating Load Calculation Principles*. Atlanta: American Society of Heating, Refrigerating, and Air-Conditioning Engineers, Inc.

BIBLIOGRAPHY

ASHRAE. 1992. An annotated guide to models and algorithms for energy calculations relating to HVAC equipment, 530-RP Final Report. Atlanta: American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.

ASHRAE. 1994. Annotated guide to models and algorithms relating to building load calculations, 741-RP Final Report. Atlanta: American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.

Liesen, R.J., and C.O. Pedersen. 1997. An evaluation of inside surface heat balance models for cooling load calculations. *ASHRAE Transactions* 103(2): 485-502.

McClellan, T.M., and C.O. Pedersen. 1997. Investigation of outside heat balance models for use in a heat balance cooling load calculation procedure. *ASHRAE Transactions* 103(2).

Spitler, J.D. 1998. Quantitative comparison of North American and U.K. cooling load calculation procedures—Methodology. *ASHRAE Transactions* 104.

Walton, G.N. 1983. *Thermal Analysis Research Program Reference Manual*. NBSSIR 83-2655. National Bureau of Standards. March 1983.

DISCUSSION

Michael J. Witte, Principal Engineer, GARD Analytics, Inc., Park Ridge, Ill.: (1) Does the driver software run 24 hours a day? (2) How does the driver software initialize the building conditions? (3) Is repeating the day to initialize conditions the official way to apply the heat balance method for calculating design cooling load as specified in TC 4.1's handbook chapter?

Curtis Pedersen: (1) Yes, the sample zone assembly runs a 24-hour design day. (2) The user can select the number of initialization days. Usually, six or eight days are used. (3) Yes, that is the recommended procedure.

Robert Sonderegger, Chief Application Scientist, Silicon Energy Corp., Alameda, Calif.: Would it be possible to convert the Fortran routines into a more regular environment such as Microsoft Excel?

Pedersen: There are source code conversion packages available for some languages. Fortran90 was explicitly specified as the language of choice in the project work statement. Prior experience attempting to submit a project deliverable to ASHRAE in Microsoft Excel was rejected by the sponsoring committee as being commercial. That was a different technical committee, but the sponsoring TC for this project did not consider a language other than Fortran90.

Arden C. Degner, Oak Creek, WI: Make this part of a total building heat balance program to be useful to practicing engineers using standard Microsoft to use latest ASHRAE calculation mode.

Pedersen: See the response to question # 2 above. We expect third party developer to use the modules of the toolkit and provide user software that uses the latest ASHRAE calculation mode.