SIMULATION SYNERGY: EXPANDING TRNSYS CAPABILITIES AND USABILITY

Timothy P. McDowell[1], David E. Bradley[1], Jeff W. Thornton[1] and Michael Kummert[2]
[1]Thermal Energy System Specialists, LLC, Madison, WI
[2]Solar Energy Lab, University of Wisconsin, Madison, WI

ABSTRACT

Developers of simulation packages are now able to take advantage of the increase in available desktop computing power to expand the capabilities and usability of their programs. This paper will illustrate these opportunities by discussing the different techniques the developers of the TRNSYS software package have used to try and create a synergy between TRNSYS and external programs and between the developers and users of the program.

INTRODUCTION

With the increase in available desktop computing power, there has been an increased interest in expanding the capabilities of whole building energy simulation programs. For example, users are no longer satisfied to run separate programs to perform airflow analysis and energy analysis. They would rather have one program perform both analyses at the same time. These desires have led the program developers to look at methods of expanding their packages in ways that don't over-complicate either the code itself or the user input required to perform a simulation.

The developers of TRNSYS, the Solar Energy Laboratory at the University of Wisconsin-Madison, the Centre Scientifique et Technique du Batiment in France, and Transsolar Energietechnik in Germany, have utilized a variety of approaches to expand the capabilities and the ease of use of the software package. These approaches vary from adding new functionality into the program source code, to adding links to external programs, to creating new programs to simplify the input of simulation parameters.

The intent of using these different approaches is to create a synergy between many divergent programs and techniques that work together to provide a wide variety of capabilities to the overall software package.

While the techniques described in this paper are specific to the TRNSYS software package, they are also generic concepts that can be applied to many of the other whole building energy simulation programs. However, many of them are programming language, compiler, and operating system specific and also depend on the internal coding of the simulation package. It is beyond the scope of this paper to provide instructions on how to program these techniques, as many are too complex to be clearly articulated in a limited space. However, they all take advantage of standard techniques that are well documented in the programming literature.

TRNSYS OVERVIEW

TRNSYS (Klein 2000) is a transient system simulation program with a modular structure that was designed to solve complex energy system problems by breaking the problem down into a series of smaller components. Each of these components can then be solved independently and coupled with other components to simulate and solve the larger system problem. Components (or Types as they are called) in TRNSYS may be as simple as a pump or pipe, or as complicated as a multi-zone building model. The entire program is then basically a collection of energy system component models grouped around a simulation engine (solver). The simulation engine provides the capability of interconnecting system components in any desired manner, solving differential equations, and facilitating inputs and outputs.

The modular nature of the program makes it easy for users to add content to the program by introducing new component models to the standard package.

ADDING COMPONENTS

One of the features of TRNSYS is that its modular nature allows users to add content to the package by introducing new component models to the standard package. The traditional method of adding a component model to the TRNSYS package was for the new component to be written in FORTRAN following a standard format. Then the entire code needed to be recompiled to create the new version with the added

component. Many users felt that the restriction to using FORTRAN was archaic and the developers felt that the recompilation of the kernel source code was redundant. So a new method of adding components to the TRNSYS package was developed that can be used in place of the traditional method, while retaining the traditional method for those who prefer it. The new method takes advantage of dynamic link libraries (dll) in a technique known as 'drop-in dlls' to develop a multi-dll, distributed architecture (See Figure 1).

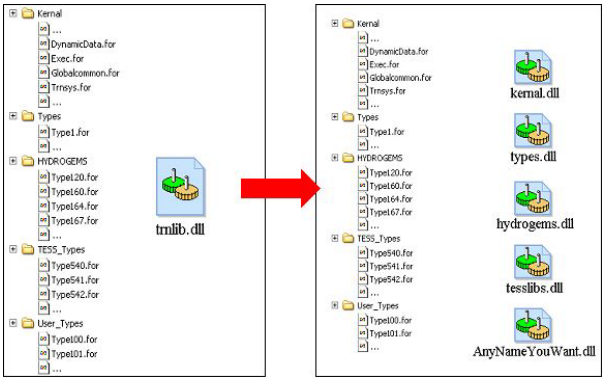

Figure 1 TRNSYS single dll architecture on the left and multi-dll distributed architecture on the right

In this method the user can create the new component using any programming language that can produce a dll and which provides for importing information to and exporting information from other dlls. This dll is then placed in a specific directory in the TRNSYS hierarchy. When the simulation is started, the TRNSYS kernal searches the dlls in this directory for any exported components to be included in the simulation. One advantage of this method is that the kernal does not need to be recompiled to add content to the package. Another advantage is that the user is no longer limited to the use of FORTRAN for creating new components. Any programming language, such as C or C++, can be used to create the dll that is called by the TRNSYS FORTRAN kernal code automatically. The disadvantage of this technique is that it requires the addition of the exporting commands in the component. These commands are often compiler and operating system specific.

Another issue arose as some users had begun their projects by first developing stand-alone models in other software packages and then later wanting to connect these models into a TRNSYS simulation. They felt that the need to recreate their models in TRNSYS format was too time consuming. Rather they wanted some method of connecting their already created models into the TRNSYS simulation structure. One approach to this would be to add code to the TRNSYS kernel to perform the same function as the steady-state modeling program. Since this requires rewriting algorithms that have already been programmed and tested in a different modeling package, the TRNSYS developers felt this approach unnecessarily reinvents the wheel. A different approach is to continue to allow the stand-alone modeling program to perform its task and create a linking structure between TRNSYS and the other program. This idea poses different difficulties than linking in components from dlls. In this case the components do not follow the TRNSYS conventions and they are often housed in unmodifiable external programs not dlls. To allow for such models to be included in TRNSYS simulations it was necessary to create components in the TRNSYS code that explicitly link to the external programs, execute the models and provide some method of data transfer between the programs. So far these links have been created for Engineering Equation Solver (EES) (Klein 2004), MATLAB™ and Simulink™ (Mathworks 2003), and Microsoft Excel™ (Microsoft 2004). These components work by having TRNSYS call the external program at each iteration and executing the model in its program. The results are then passed back to TRNSYS where they can be used in a TRNSYS simulation like any other results from a standard component. A disadvantage of this technique is that in order to create a communication pathway between two programs requires the use of operating system specific commands. This techniques also leads to a longer simulation time than if the model had been package into a normal TRNSYS component.

ADDING EXTERNAL CONTENT

Sometimes the desired added content already exists in a separate program and previously it was too computationally intensive for both the TRNSYS and additional analyses to be carried out at the same time (i.e. airflow simulation, daylighting, or CFD modeling). But with the increase in computer power available to the common user, the possibility of combining these analyses now exists.

There are many different techniques for combining analysis programs but they fall into two general categories: adding the algorithms of one analysis into the algorithms of the other or linking the two programs together externally.

These can be demonstrated by looking at the combining of building energy and airflow analysis in TRNSYS. Two widely used airflow analysis progams – CONTAM (Dols 2002) and COMIS (Feustel 1998) – have been integrated into TRNSYS using two very different methods. The COMIS-TRNSYS link was

developed using the more standard method where the algorithms in COMIS were directly integrated into the building model of TRNSYS. In essence, the COMIS code has become a part of the TRNSYS building model code. The advantage of this technique is that the TRNSYS solution engine takes care of all of the iterative solution schemes. The disadvantage is that the airflow code is spread throughout the building model code and is present even when airflow analysis is not desired. This makes determining the source of problems in the airflow or building analysis more difficult. The CONTAM-TRNSYS link, on the other hand, was done by utilizing TRNSYS's modular structure. A stand-alone version of the CONTAM code (called AIRNET (Walton 1989)) was turned into a TRNSYS component by including some 'wrapper' code that includes the standard calling structure used by TRNSYS. This airflow component can then be included in a TRNSYS simulation like any other component. The advantages of this technique are that the airflow modeling code is included whole as it is used in its stand-alone form and the component is only linked into the simulation when airflow calculations are to be performed. The disadvantage of this method is that the airflow model carries out its own iterative solution at each iteration of the TRNSYS simulation. This can lead to convergence issues during the simulation. At the same time, this allows for some novel solution techniques to combat the convergence problems. The linking of the airflow and building energy components can be adjusted so that they both come to an iterative solution at every timestep or the outputs from one component from the previous timestep can be used at the current timestep as inputs to the other component (i.e. onion vs ping-pong (Hensen 1996)).

These same techniques are being explored for adding additional external programs to the TRNSYS package such as CFD analyses.

EASE-OF-USE ENHANCEMENTS

One of the long-standing drawbacks to the large simulation packages is the difficulty in creating the input files that controlled the simulation. Typically these files are text documents containing large quantities of numbers that set the parameters and connections that control the simulation with little, if any, description as to what they mean. The beginning user was often overwhelmed with the concept of starting a simulation and determining what someone else had done in a simulation was difficult. So developers began to add ease-of-use enhancements to their packages to increase the usability of the packages. These enhancements are often referred to as front-ends or graphical user interfaces (GUIs). For TRNSYS the first generation of these enhancements were IISibat and Presim for the general simulation and Prebid for the building model, all released in 1996.

The simulation front-ends allow for the links between components to be graphically shown and the simulation parameters entered (See Figure 2). The building model front-end allows for the input parameters to be grouped in logical ways for user entry. For example the physical properties of the building materials are entered and then grouped together to form the wall structures used in the simulation and all of the zone parameters are entered in a single screen. The front-ends then create the text files needed by the simulation engine and execute the simulation program.

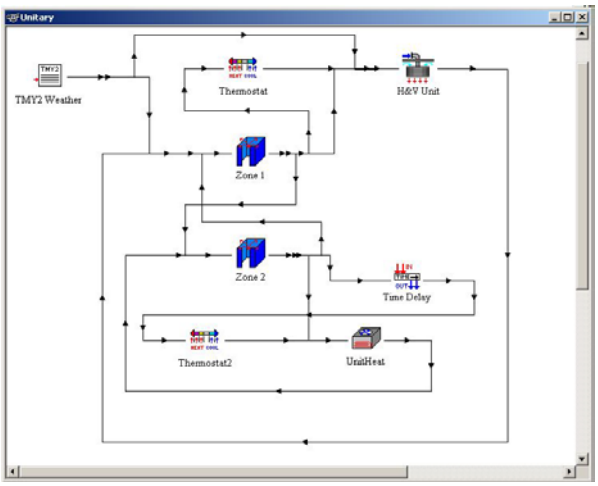

Figure 2 An example of a project in the IISiBat front-end program

While these first-generation programs were dramatic improvements over the text file entry of the simulation package, they were just baby steps down the path of usability. The lessons learned and experience gained from the use of these front-ends has led to the development of the next generation of ease-of-use enhancements: TRNSYS Studio, TRNBUILD, and SimCad. In the first-generation of simulation front-ends the parameters and variables continued to be entered by the user in tables or lists with only minimal commentary as to their meaning. A new alternative has been added where applications can be created, by the user or the developer, which gather this information from the user in more descriptive or meaningful ways (TRNSYS Studio plug-ins). For example instead of filling out a list of the width, length, and depth of a solar collector, these numbers can be entered on a diagram of the collector which clearly shows what is meant by those dimensions on a collector (See Figure 3). The old method for entering

the parameters has also been retained for users that are experienced and comfortable with the parameter lists.

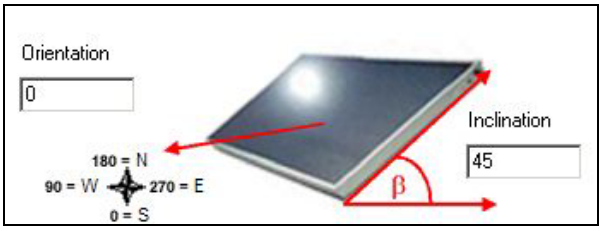

Figure 3 An example of an application for entering the parameters of a solar collector model

One of the drawbacks of the first generation building model front-end was that it did not provide for a direct link between the CAD plans of a building and the building model simulation parameters. Information was manually taken from the drawings and entered in the building model front-end. SimCad represents a step toward generating the building model parameters directly from the CAD drawings (See Figure 4). Instead of representing a building wall as two parallel lines in a purely pictorial representation, SimCad represents a wall as an object. As such, a wall (or another building element) can have associated properties such as area, azimuth, layer composition, etc. While the SimCad user builds a graphical representation, they simultaneously build the basis of the building's thermal model. In the current version, additional information such as heating and cooling set points still need to be entered into the building model front-end, however, SimCad is seen as a first step toward taking all of the building model information directly from the CAD drawings.

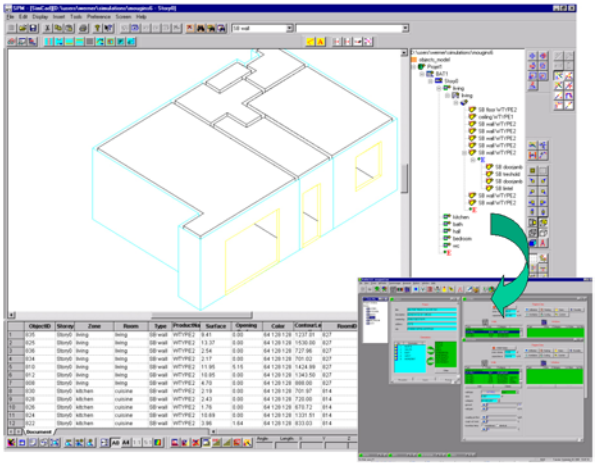

Figure 4 An example of a project in the SimCad building front-end program

Ease-of-use enhancements are not limited to the simulation package itself, but can also be applied to external programs that enhance the overall packages. For example, GenOpt® (Wetter 2004) is a generic optimization program developed by Lawrence Berkeley National Laboratory (LBNL) that can be used to run optization studies with simulation packages. The GenOpt® program is an external program that controls the simulation program inputs based on an user-entered error function calculated by the simulation package. It has been developed in such a way that it can be used with most of the simulation packages available today. While it is a very powerful program it is also somewhat complicated to figure out how to apply it to your specific simulation package. So a program called TrnOpt has been developed to provide an easier link between the two programs. This program parses the simulation file to determine the available variables that can be used for an optimization study and provides a straightforward way of selecting the variables, the optimization method, and the parameters of the optimization. It then creates the appropriate simulation and optimization files and then lets the programs do their processes. While this does make it easier to get started with the optmization process, it does not recommend any optimization method or parameters for a given situation. It is still important that the user understand how both the simulation program and the optimization program work in order to select the most appropriate method and parameters.

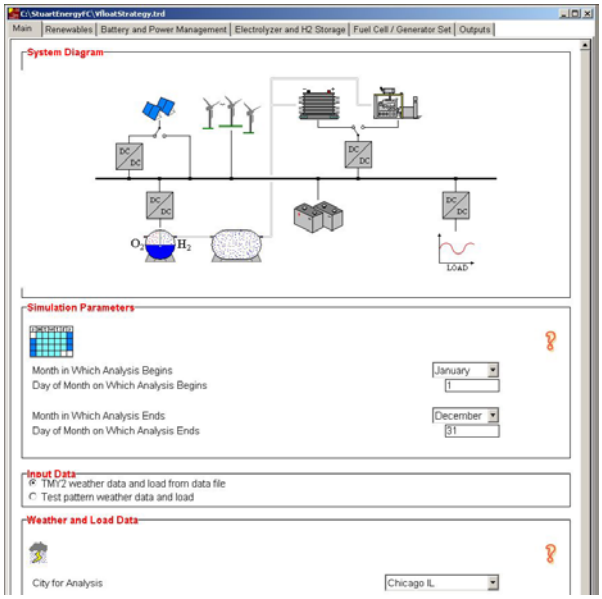

Figure 5 An example of a project in the TRNSED front-end program

END-USER ENHANCEMENTS

The typical end product of a simulation study is a large quantity of data that are packaged into a report and submitted to a client. But what if the client would like to perform some parametric study of their own without

having to learn how to use the simulation package itself? One method of doing this is to have the client provide all of the details of the parametric study and then provide the client with the results of all of the runs. This can lead to many iterations back and forth with the client because the details are not well known before the study is begun. The TRNSYS developers have created a different method in the form of a front-end program called TRNSED (See Figure 5).

The concept behind TRNSED is that a TRNSYS user can modify the text based simulation input file in such a way as to hide unnecessary simulation details, show only those aspects of the simulation that are of interest and give the end client the ability to make targeted modifications to the simulation and rerun it themselves. In so doing, the text based input file is recast as a dedicated front end specifically designed for that simulation.

OVERALL SIMULATION ENVIRONMENT

While all of the programs and enhancements discussed are a part of or connected to the TRNSYS program, most are still separate programs which are called in some sequence to perform the simulation. The long-term vision for the TRNSYS package is to develop an over-all simulation environment which will encompass the links to external programs, the front-ends, and the user enhancements with the TRNSYS simulation engine. This simulation environment would continue the TRNSYS concept of separate components that are linked together into a single entity and easy addition of new components to the package. Thus as new links and enhancements are developed they can be quickly integrated into the overall simulation environment. The first generation of this process will be released with version 16 of TRNSYS in the form of the TRNSYS Simulation Studio program.

CONCLUSION

As the available computing power continues to increase it will create more and more opportunities for the capabilities and features of simulation packages to be expanded. It is important that the developers of these packages not lose sight of other products already available and the end-users of their packages. There is no reason to re-invent the wheel. If there is already a product that can perform the desired function it may be adequate to develop an external link to that program rather than create new code to perform the same analysis in their software package. Adding more and more features to simulation packages is wonderful unless it is done in such a way to limit their use to the 'expert' users of these packages. If enhancements are not made which make it easier for these packages to be learned and used then the new features will do little to impact the simulation community as a whole. While this paper has discussed enhancements to the TRNSYS package, these issues and improvements are not limited to this one specific simulation package. Developers of all of the simulation packages are dealing with the same opportunities and difficulties and continue to strive to bring the best products possible to the simulation community. Finally it is also important that the users of these programs not assume that these ease-of-use enhancements have done their simulation job for them. Any development work, from writing the models to creating front-ends, involves assumptions being made by the developers. If the users do not understand the concept in the models and the assumptions made in the models or the developers do not make the limitations clear, then the program may be used to simulate a process that is not adequately covered by the simulation.

ACKNOWLEDGMENT

The authors would like to thank the members of the TRNSYS Development Group – Solar Energy Laboratory at the University of Wisconsin-Madison, the Centre Scientifique et Technique du Batiment in Nice, France, and Transsolar Energietechnik in Stuggart, Germany for their assistance.

REFERENCES

Dols, W.S. and G.W. Walton. 2002. *CONTAMW 2.0 User Manual,* NISTIR 6921.

Feustel, H. 1998. *COMIS – Am International Multizone Air-Flow and Contaminant Transport Model*. LBNL Technical Report LBNL-42182. Lawrence Berkeley National Laboratory.

Hensen, J. 1996 "Modelling coupled heat and air flow: ping-pong vs. onions." 16th AIVC Conference, Air Infiltration and Ventilation Centre.

Klein, S. 2000. *TRNSYS – A transient system simulation program*. Engineering Experiment Station Report 38-13. Solar Energy Laboratory, University of Wisconsin-Madison.

Klein, S. 2004. *EES – Engineering Equation Solver*. F-Chart Software.

The Mathworks, Inc, 2003 – MATLAB and Simulink are registered trademarks

Microsoft Corporation, 2004 – Microsoft Excel is a registered trademark

Walton, G., 1989. *AIRNET – A Computer Program for Building Airflow Network Modeling*. NISTIR 89-4072 National Institute of Standards and Technology.

Wetter, M. 2004. GenOpt® *Generic Optimization Program User Manual Version 2.0.0*, LBNL Technical Report LBNL-54199, Lawrence Berkeley National Laboratory.