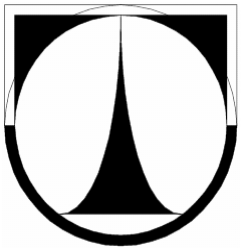
30. Setkání kateder mechaniky tekutin a termomechaniky

22.-24.6. 2011

Špindlerův Mlýn

Jednotlivý příspěvek ze sborníku



TECHNICKÁ UNIVERZITA V LIBERCI

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

User's Code Implementation for TRNSYS Simulations

Lubomír KLIMEŠ[1]

[1] Ing. Lubomír Klimeš, Energy Institute, Faculty of Mechanical Engineering, Brno University of Technology, Technická 2896/2, 616 69 Brno, yklime17@stud.fme.vutbr.cz

Abstract: *The purpose of the article is to introduce two possible approaches that can be used for a user's own code implementation in the simulation software TRNSYS. In particular, the connection between the numerical computing software MATLAB and TRNSYS is introduced and discussed. Further, the implementation of user's type for TRNSYS by using the programming language C++ is adumbrated and both the presented methods for a user's code implementation in TRNSYS are compared to each other. Finally, the benefits and drawbacks of both approaches are listed and discussed.*

1 Introduction

The simulation software TRNSYS (abbreviation TRaNsient SYstems Simulation) belongs to the class of simulation tools that are designated for modelling of dynamic systems. In particular, the TRNSYS software is widely being used by engineers to analyse various dynamic systems of buildings, especially from the energy point of view. By the TRNSYS software, a model including energy balances, solar and photovoltaic components and systems of heating and ventilation can be utilized, solved and analyzed, see [1].

The TRNSYS software uses a module-based approach, and thus the user assembles a model by adding various components (modules) from the prearranged toolbox to the workspace. These components are called *types* in TRNSYS and are identified by their numbers. Afterwards, the connections between particular components are defined by the user and the simulation can be performed.

The described module-based approach is sufficient so far as the components toolbox of TRNSYS software integrates all components that the user needs for the simulation model. A modelling problem can occur in the case there is no type integrated in the components toolbox for a desired action. This situation may be solved by the implementation of user's own code and its connection to the simulation software TRNSYS. This approach allows the user to program, utilize and control entirely the numerical procedures and computations in the implementation part.

Due to these reasons, two possible approaches how the user's code can be implemented are discussed below – the connection between the TRNSYS software and the numerical computing environment MATLAB via Type155, and the user's own type implementation by using the programming language C++.

2 User's Code Implementation in MATLAB via Type155

The software MATLAB (abbreviation MATrix LABoratory) is a powerful numerical computing tool for solving various scientific and engineering problems. The handling with MATLAB combines the scripts writing similar to programming in ordinary languages, and utilizing of prearranged numerical procedures and functions, see [2].

The TRNSYS software allows the connection with MATLAB via build-in Type155, see [3] and Fig. 1. Thus, the user's code is implemented in a MATLAB m-file and the Type155 in TRNSYS is linked to it. When the user's code in MATLAB is required to be performed, the TRNSYS

software starts MATLAB and the desired m-file is executed.

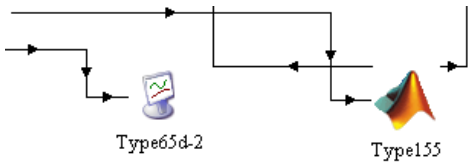


Figure 1: Type155 for the connection with MATLAB

The Type155 allows the user to define the numbers of inputs and outputs that are established for the link between TRNSYS and MATLAB. The inputs from TRNSYS are automatically passed into the MATLAB code by the vector `trnInputs` and its elements may be used for desired calculations and other procedures. Conversely, the outputs calculated by MATLAB have to be passed back to TRNSYS by the vector `trnOutputs` with a predefined length.

Apart from the vector `trnInputs` with inputs, TRNSYS also passes into MATLAB other additional and useful parameters: the simulation time `trnTime`, the TRNSYS info array `trnInfo` containing information for implementation details and the simulation start and stop time `trnStartTime` and `trnStopTime`, respectively. All these parameters can be used in the implementation without any initialization, and moreover, the array of parameters `trnInfo` contains necessary information needed for the identification of the call (the first iteration, the last iteration, etc.) and also the numbers of inputs and outputs. On the other hand, besides the output vector `trnOutputs` MATLAB returns back to TRNSYS also an additional parameter `mFileErrorCode`. This parameter is designated for debugging and the run-time error identification and it can be used to identify where the code in MATLAB crashes.

The most significant benefit of using the connection between MATLAB and TRNSYS for the user's code implementation is the fact that all the tools, procedures and functionality of MATLAB can be used for the computations. Hence, functions for various numerical methods, plotting and visualization of results as well as tools of specialized toolboxes can be easily utilized. On the other hand, the usage of that approach requires the user to be familiar with MATLAB and scripts writing. The main drawbacks of using MATLAB are as follows. From the user's point of view, the most annoying disadvantage is especially the complicated configuration of the link between TRNSYS and MATLAB depending on both software versions. Further, for simulations in TRNSYS including Type155 the installed and runnable MATLAB software is required. Owing to the price of licences for MATLAB, this disadvantage may play a crucial role. The third significant drawback of using (not only) MATLAB is a very poor capability for code tuning and debugging since the breakpoints, debugger and other tuning facilities from the MATLAB environment are not available in TRNSYS simulations.

3 User's Type Implementation by Using C++

The purpose of the user's type implementation is to produce a module that can be added to the toolbox of components in TRNSYS, and thus also the handling with the user's type does not differ in comparison to standard components in TRNSYS, see [4]. Hence, from the user's point of view this approach of code implementation is very intuitive and the user's module can be easily utilized by others. An example of the user's Type250 in the toolbox of standard components in TRNSYS is illustrated in Fig. 2 and its connection to the model is shown in Fig. 3.

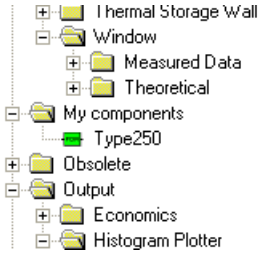


Figure 2: The user's Type250 embedded in the TRNSYS toolbox

In order to implement a user's type, the so-called proforma, see [5], has to be created in TRNSYS that contains all the information and specifications about new module, for instance its type number, its description, the numbers of parameters, inputs and outputs, external files, etc. Once the proforma is defined the TRNSYS software allows to create a skeleton of user's type for C++ (and Fortran) programming language, see [6].

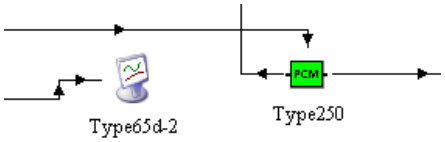


Figure 3: A user's Type250 in a TRNSYS model

The skeleton of new type in C++ is prepared to be used in a Microsoft Visual Studio 2003 project that is configured to produce a DLL file containing the compiled code of new user's module. Afterwards, the dynamic link library, DLL, with the proforma can be easily distributed to other users who can immediately utilize the new module.

The prototype of the user's module has in C++ source code the following form,

```
extern "C" __declspec(dllexport)
  int TYPE250( double &time,
               double xin[],
               double xout[],
               double &t,
               double &dtdt,
               double par[],
               int info[],
               int icntrl);
```

where `time`, `xin`, `par` and `info` are the simulation time, inputs, parameters and the info array, respectively, passed from the simulation in TRNSYS. After desired calculations, the outputs are passed back to TRNSYS by using the pointer `xout`. Other useful functions and parameters from TRNSYS can be easily utilized in the C++ source code by including the attached header file `TRNSYS.h`.

Due to the C++ programming language used for the implementation of user's type, the most considerable advantage is a wide variety of possibilities from the programmer's point of view given by the C++ programming language itself, see [6]. Moreover, this approach requires no additional and expensive software in contrast to the connection between TRNSYS and MATLAB for which MATLAB is needed to be installed. On the other hand, the author of new type has to be familiar with programming in C++ that is more difficult than scripts writing in MATLAB. The most significant drawback of user's type implementation in C++ is the fact that all computations and numerical procedures must be programmed or provided by another approach by the programmer in contrast to the utilization of build-in functions and procedures in MATLAB. The identical disadvantage as in the case of MATLAB and TRNSYS connection is debugging and code tuning because breakpoints, debugger and other facilities from Visual Studio cannot be used.

4 Results and Discussion

Both the methods and their implementations were successfully tested on a TRNSYS model of solar collector where the user's code was used for the numerical computing of heat transfer in a wall containing phase change materials, see [7, 8]. Owing to the phase changes, latent heat, its storage and release, particular numerical methods and techniques based either on the enthalpy approach or on the effective heat capacity approach, see [9], were needed to implement and utilize in order to solve numerically the heat conduction in a wall. The particular results of the solar collector simulation with phase change materials will be published soon.

It has been shown by numerical experiments that both the methods give the corresponding results in a comparable time, and thus the particular approach to be used depends especially on the user's preferences, knowledge of writing

scripts in MATLAB or programming in C++ and the software equipment. Both the user's code implementations can be distributed to other users. However, the important difference is that the m-file with a code for MATLAB is open for other users and it can be directly edited and modified by them whereas the user's module in the form of compiled dynamic link library, DLL, and its implementation are closed, and therefore the direct modification is not possible. On the other hand, the source code of the user's module can also be provided and distributed to other users but in addition a C++ compiler is at least required to recompile the DLL file containing the module. From the programmer's point of view, the main disadvantage of both the presented approaches is a difficult and poor possibility for debugging and code tuning, and hence these standard programming activities must be performed by other untypical techniques.

5 Conclusion

The presented two approaches – the connection between TRNSYS and MATLAB via Type155, and the user's type implementation by using the programming language C++ (or Fortran) – are valuable tools for the user's code implementation in the simulation software TRNSYS. This technique is needed to be utilized especially in the case the standard TRNSYS toolbox does not contain a desired component. The numerical experiments performed on the simulation of solar collector showed that there is no significant difference between both the approaches, and therefore the particular choice depends mainly on preferences and knowledge of users, and available software tools. The future activity will be aimed at the implementation of the module for solving the heat transfer in composed walls consisting of several layers. Although the TRNSYS software includes a module for this purpose, it is based on transfer functions and materials with heat conductivities close to zero cause numerical difficulties, typically the division-by-zero errors.

Acknowledgement. The research leading to the presented results was supported by the grants of the Grant Academy of the Czech Republic GAČR106/09/0940, GAČR106/08/0606, GAČR P107/11/1566, by the project OC10051 of Czech Ministry of Education and by the project of the specific research BUT BD13102003. The author, the holder of Brno PhD Talent Financial Aid sponsored by Brno City Municipality, also gratefully acknowledges for that financial support.

6 References

[1] ŠOUREK, B.; KOREČKO, J. *TZB-info* [online]. 30.7.2004 [cit. 2011-05-06]. Simulační prostředí TRNSYS. Dostupné z WWW: <http://www.tzb-info.cz/1956-simulacni-prostredi-trnsys>.

[2] LYSHEVSKI, S. E. *Engineering and Scientific Computations Using MATLAB*. 1st edition. New Jersey: John Wiley and Sons, 2003. 240 s. ISBN 978-0471462002.

[3] *Standard Component Library Overview*. Documentation of TRNSYS 17, Volume 3. Solar Energy Laboratory, University of Wisconsin-Madison.

[4] *Programmer's Guide*. Documentation of TRNSYS 17, volume 5. Solar Energy Laboratory, University of Wisconsin-Madison.

[5] *Getting Started*. Documentation of TRNSYS 17, Volume 1. Solar Energy Laboratory, University of Wisconsin-Madison.

[6] PRATA, S. *C++ Primer Plus*. 5th edition. Indianapolis: Sams Publishing, 2004. 1224 s. ISBN 978-0672326974.

[7] CENGEL, Y. A. *Heat Transfer : A Practical Approach*. 2nd edition. New York: McGraw-Hill, 2003. 1024 s. ISBN 978-0070115057.

[8] KUZNIK, F.; VIRGONE, J.; ROUX, J.-J. Energetic efficiency of room wall containing PCM wallboard : A full-scale experimental investigation. *Energy and Buildings*. 2008, Volume 40, Issue 2, s. 148-156. ISSN 0378-7788.

[9] KUZNIK, F., et al. A review on phase change materials integrated in building walls. *Renewable and Sustainable Energy Reviews*. 2011, Volume 15, Issue 1, s. 379-391. ISSN 1364-0321.