Towards Open Systems: A PSCAD/EMTDC to MATLAB Interface

A.M. Gole A. Daneshpooy
University of Manitoba,
Winnipeg, MB., CANADA

Abstract: **The open systems approach provides hooks to a software package that makes it readily interfacable to other software packages. THE PSCAD/EMTDC electromagnetic transients simulation program through its FORTRAN interface allows for the introduction of such hooks. Similarly MATLAB also allows communication with other packages. Using these features a MATLAB interface to PSCAD/EMTDC was developed, which is accessible using PSCAD's graphical user interface (GUI). In addition to the description of the interface, a typical simulation example of a Fuzzy-Logic based controller for an ac-dc controlled rectifier is presented.**

Keywords: MATLAB, Electromagnetic Transients Simulation, Open Systems, Fuzzy Logic, Power Electronics, Graphical User Interface

I. INTRODUCTION

There is a growing trend in software development towards an Open Systems approach. In this approach, each piece of software is designed to satisfy a certain protocol that makes it possible for these programs to be interfaced easily. The recent inclusion of the Dynamic Data Exchange (DDE) capability in Operating Systems such as Windows 95 and interprocess communications in Solaris and other Unix implementations make such developments easily implementable. This makes it possible for programs developed by different vendors to be used together with full interaction.

The emtp-type program PSCAD/EMTDC [1] has been designed to take advantage of such an approach. The popular mathematics and control systems design software package MATLAB [2] also has this capability. In this paper we describe the interfacing of these two packages. This allows for the powerful mathematical and control functions, including the various toolboxes available in MATLAB to be used with the fast and powerful electrical-network modeling capability of the emtp-type program. It must be noted however, that the solution speed is considerably slower for the hybrid system than for a pure PSCAD/EMTDC run. Thus we recommend this approach as an intermediate one in which several control algorithms available from MATLAB are to be evaluated. After a given approach is selected to be the one used in the final control system, it is usually best to write a PSCAD/EMTDC model for that selected controller. This gives the fastest simulation speeds.

The implementation is carried out on a SUN Microsystems computer running the Solaris operating system. The interprocess communication capability is used for communicating between the two software packages, so that two programs that are running simultaneously can exchange data with each other.

This synergy of an electromagnetic transients program with MATLAB has several advantages. Although the power system network equations can be programed into MATLAB, there is as yet no comprehensive MATLAB toolbox that will suffice for industry applications. Also, emtp-type programs [3] are optimized and specialized for power-network simulations. They usually run faster than interpreted MATLAB code, if the latter is used for their modelling. The Graphical User Interface (GUI) available with an emtp-type program such as PSCAD/EMTDC is also customized for power-industry applications [1] and is thus more straightforward to use than the SIMULINK GUI available in MATLAB. On the other hand, MATLAB offers a large library of control functions, and through its toolboxes, it also offers a wide range of pre-programmed algorithms. Thus when investigating a new control technique for power systems, it is convenient to use such toolboxes.

Previous work on the subject includes writing of the emtp-type algorithm using MATLAB [4] . An approach similar to the one used here has been reported with the ATP program[5]. However in this paper we describe an automated approach where the new component and the necessary interface files can be generated automatically. The resulting component is then available in graphical form as a block in PSCAD/EMTDC

II. PSCAD/EMTDC TO MATLAB INTERFACE

A. PSCAD/EMTDC and MATLAB Description

Data input to PSCAD/EMTDC is carried out using a GUI interface called DRAFT [1]. After drawing the circuit diagram and entering values via pop-up menus that are selected by clicking on the component's icon, the resulting schematic diagram is translated into a data file which includes the fixed network data and a FORTRAN file called

DSDYN that contains dynamic models such as control blocks. This interface is a powerful feature as it allows users to develop their own models in FORTRAN and add these to the PSCAD/EMTDC component repertoire. This feature has been used to establish the communication between the two programs.

MATLAB on the other hand is an interpreted language which contains a large number of mathematical functions and control system blocks. These are invoked using statements entered directly onto the MATLAB command line or directly called through a "m-file". One feature available in MATLAB is that the MATLAB engine can be started from a user written FORTRAN or C program. This feature allows the communication with PSCAD/EMTDC to be established.

B. Structure of the Interface

The structure of the interface is as shown in Fig. 1. As described in the previous section PSCAD/EMTDC has a fortran file called DSDYN through which external FORTRAN subroutines can be called. A Fortran subroutine is therefore written which starts the MATLAB engine and sets up the data communication pipe between this subroutine and the MATLAB engine. The m-file containing the MATLAB commands is also passed to the MATLAB engine. The MATLAB part of the simulation is of course independently written and stored in this m-file.

It should be noted that the FORTRAN subroutine is not user-written but is automatically generated using a program developed by the authors. Also, it is possible in a network environment to start the MATLAB engine on a separate computer (from the one running PSCAD/EMTDC); even when the two machines have different architectures. The resultant parallelism gives a speed-up in the total execution time.

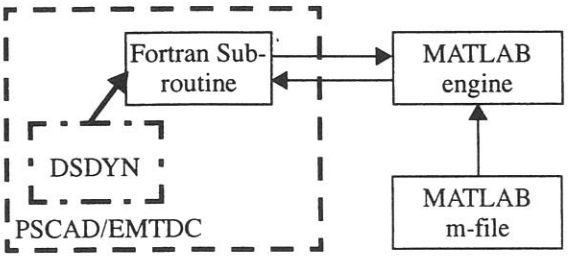

Fig. 1. Structure of PSCAD/EMTDC-MATLAB interface

As both the MATLAB and EMTDC blocks are concurrently solved, there is in principal no time-step delay in the solution (unless a user introduces a lag possibily due to sloppy programming during component design). Also as no special conversion of data is used during the exchange of information, there is in essence no interface error.

C. Development of the MATLAB Block

The MATLAB block in PSCAD/EMTDC is seen as a graphical icon on the DRAFT palette. Connections to this block from other PSCAD/EMTDC blocks is made by dragging and dropping connecting wires. As described earlier in subsection B an associated FORTRAN subroutine and the corresponding m-file make up the full definition of the MATLAB component as shown in Fig. 2.

A new MATLAB block can be developed using a program written by the authors. The program asks for the name of the new component and the number of inputs and outputs and their names. A graphical icon of the block is then automatically generated along with an empty m-file, which is opened for user input in a text-editor shell. The user should then enter the appropriate MATLAB statements into this m-file. Fig. 3 shows a typical PSCAD/EMTDC case in which the MATLAB component labelled "Fuzzy" is being used. Inputs or outputs can be scalers or arrays. The input in Fig. 3 is an array of two components (i and di/dt), and the output is a scalar ($\Delta\alpha$).

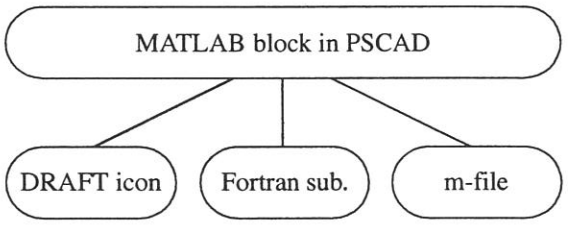

Fig. 2. The PSCAD MATLAB block and its constituents

Once the component is developed, it may be used later merely by dragging and dropping the icon on the DRAFT palette.

III. SIMULATION EXAMPLE

The example presented here describes the application of the PSCAD/MATLAB interface to the case of a ac-dc converter being controlled by a Fuzzy Logic based controller. The Fuzzy Controller is modelled using the MATLAB Fuzzy Logic toolbox. The schematic diagram can be seen in the PSCAD DRAFT palette shown in Fig. 3, with the associated m-file shown in Fig. 4. The strategy is to follow the

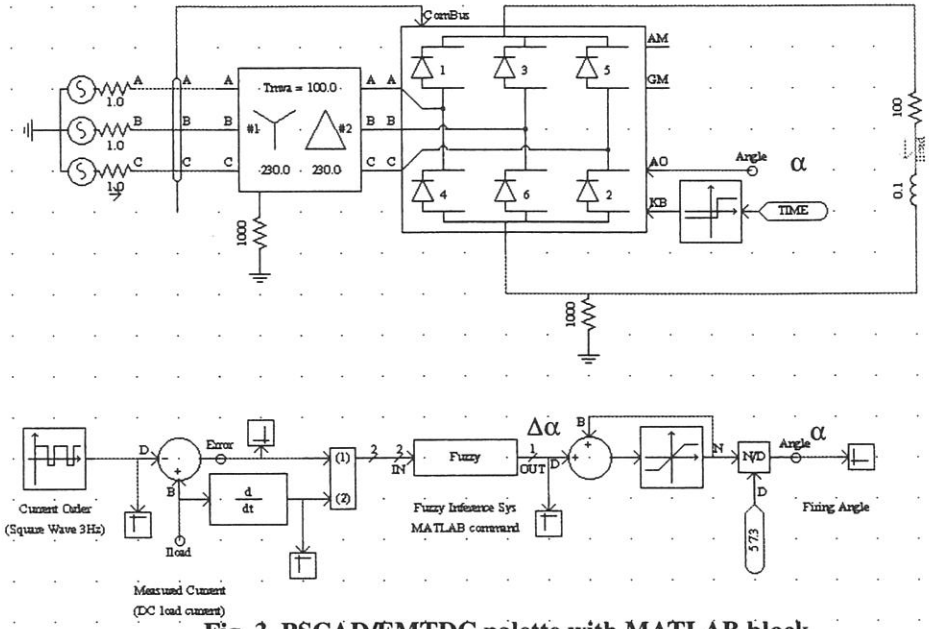

Fig. 3. PSCAD/EMTDC palette with MATLAB block

given reference current order with maximum speed and minimal overshoot. The current is controlled via the firing angle α of the rectifier. In the simulated case, the current reference is a square wave of 3 Hz frequency which oscillates between 1.5 kA and 1.0 kA. The Fuzzy Controller either increases or decreases α in each time-step of the digital control system. The rate of change of α is a function of both the current error and the rate of change of error, which are classified in terms of fuzzy values such as ***"high"," low"*** and ***"okay"*** (for current) and ***"positive" or "negative"*** for di/dt . The controller works on the basis of the following 5 fuzzy rules:

1. IF current = ***high*** THEN Δα = ***decrease fast***
2. IF current = ***low*** THEN Δα = ***increase fast***
3. IF current = ***okay*** THEN Δα= ***no change***
4. IF (current = ***okay***) AND (rate of change is ***negative***) THEN Δα = ***decrease slowly***
5. IF (current = ***okay***) AND (rate of change is ***positive***) THEN Δα = ***increase slowly***

The membership functions [6] for current, rate of change and output are shown in Fig. 5. The entire fuzzy inference evaluation is carried out in a single function called ***evalfis*** available in the Fuzzy Logic Toolbox[7]. The required m-file that has to be entered by the user is thus very straightforward. Fig. 4 shows a fragment of the m-file with the key functions used in it. The variable ***rect*** is a matrix which includes the data for the fuzzy inference algorithm and is read once at the beginning of the run.

The results from the simulation are seen in Fig. 6. It can be seen that the load current follows the desired reference with minimal overshoot. In addition to the reference and actual current, the graph shows the angle, the error and the rate variables.

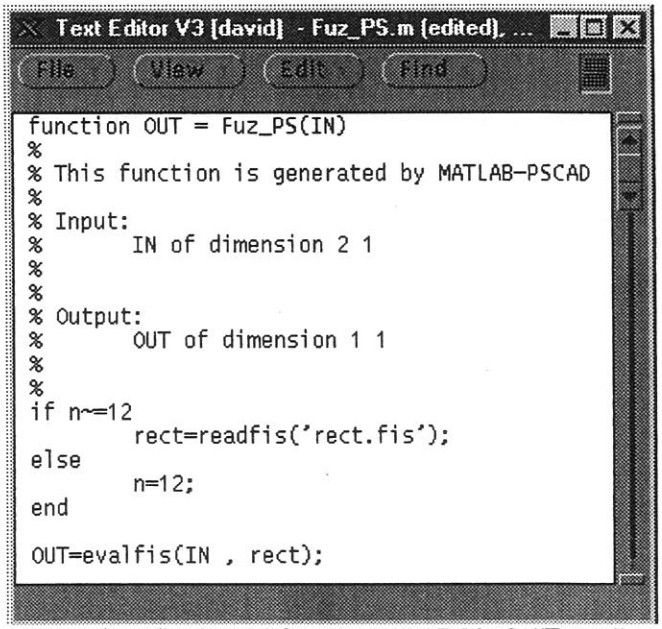

Fig. 4. m-file for PSCAD/MATLAB block "Fuzzy"

Also note that any of the MATLAB graphical functions can also be invoked from PSCAD. For example, the plot in Fig. 5 for the fuzzy membership functions is MATLAB

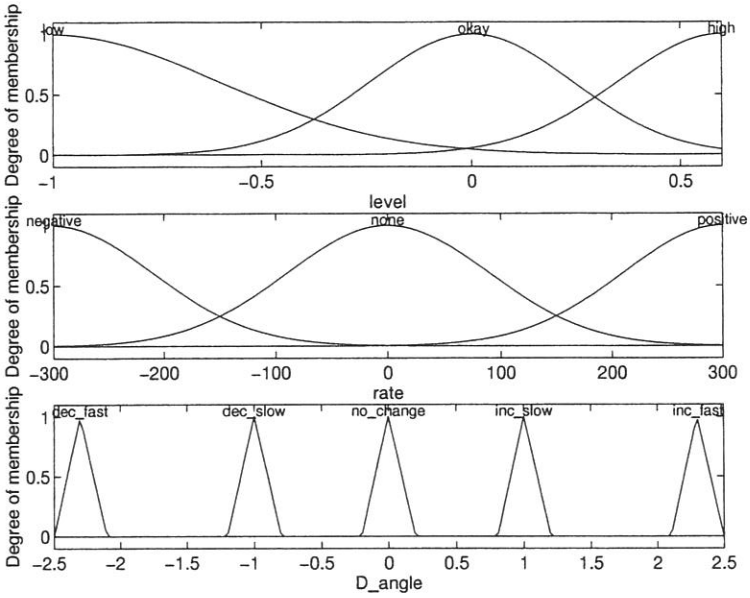

Fig. 5. Fuzzy membership functions

generated. Similarly the surface plot showing the output versus the two inputs is also directly generated by MATLAB and is shown in Fig. 7. These provide valuable diagnostic tools for analyzing the controls.

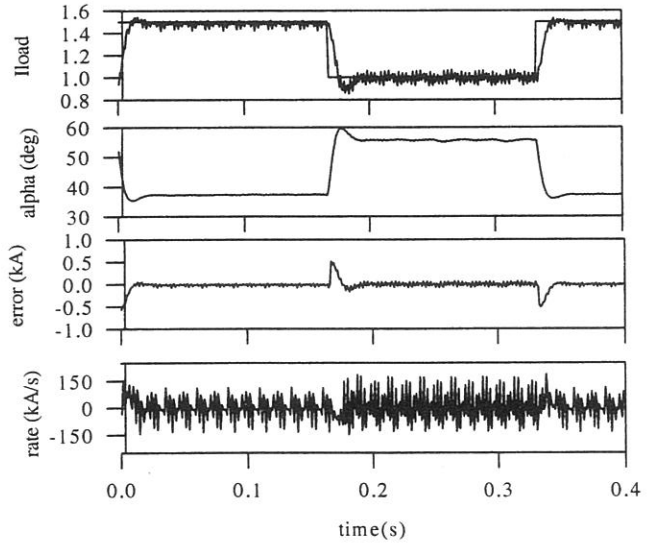

Fig. 6. Fuzzy controller simulation waveforms

The overall simulation time required for the example is larger than that if the Fuzzy component were directly programmed in FORTRAN as a standard PSCAD block. The interface allows one to investigate a large number of control possibilities pre-programmed in the MATLAB toolboxes. However, once the control strategy is finalized, it may be worthwhile to re-program the algorithm into a standard PSCAD/EMTDC block written in FORTRAN. This also allows for the block to be compiled and made into a library within PSCAD/EMTDC and thus greatly speeds up the simulation process.

In our example, the EMTDC solution took 1.5 s and the MATLAB part took about 60 s on a DEC Alpha 250 workstation. However this may not be a fair evaluation as there is some fixed overhead associated with the inclusion of the MATLAB engine. When the problem size is doubled, the EMTDC solution time is doubled, but the overall simulation time only increases by about 15%.

IV. CONCLUSIONS

The powerful control system modelling capabilities in MATLAB were accessed through the PSCAD/EMTDC simulation program. The FORTRAN Interface available in PSCAD/EMTDC coupled with the data exchange over interprocess communication pipes on the Unix platform (or DDE on a Windows 95 platform) allows for such an interface to be easily constructed.

A special program that performs the automatic generation of the component icon and which associates the corresponding m-file with the icon, was found to greatly simplify the process of new component design. The newly developed component is then accessible through the graphical user interface of PSCAD/EMTDC just like any other component in its repertoire.

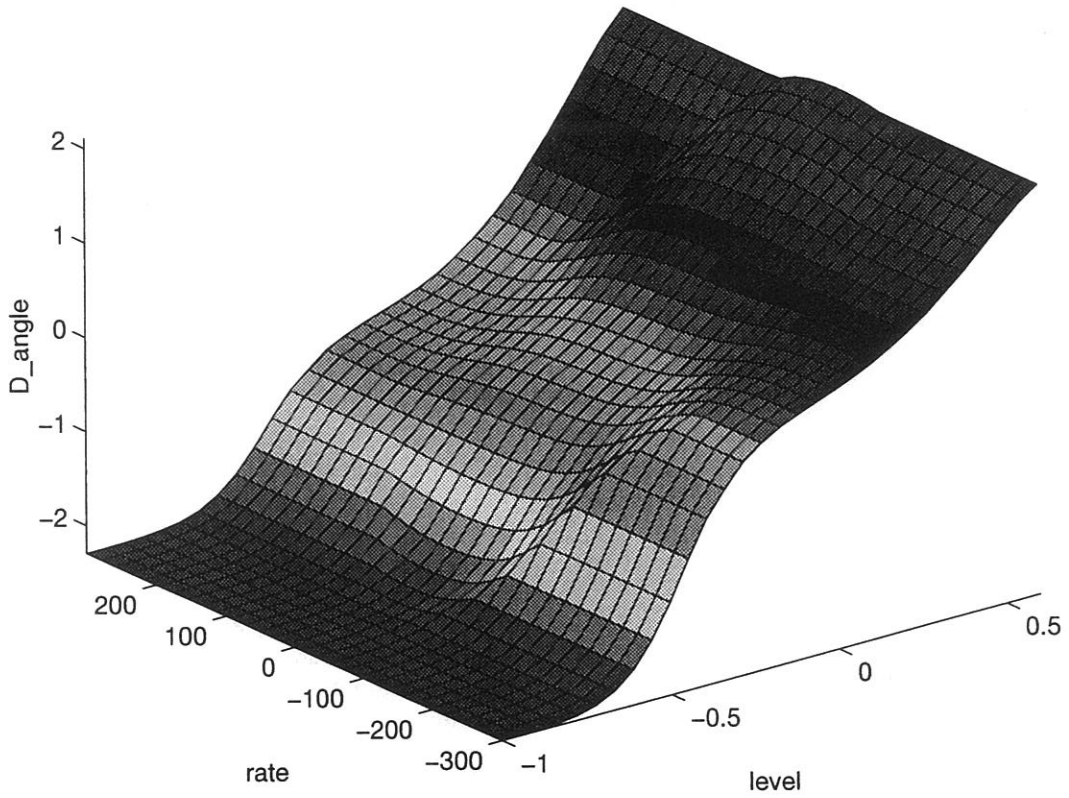

Fig. 7. Fuzzy input/output surface plot

The technique allows for immediate access to a whole range of pre-developed MATLAB control libraries as shown by the example of the Fuzzy Logic based controller for an ac-dc converter.

The CPU time required for the combined PSCAD/EMTDC-MATLAB simulation is slower than if the MATLAB component were modeled directly in PSCAD/EMTDC. Thus the above simulation approach is recommended during the evaluation stage in which several different control algorithms are being investigated. For the fastest possible runs, the component should directly be written into PSCAD/EMTDC.

V. REFERENCES

[1] A.M. Gole, O.B. Nayak. T.S. Sidhu, M.S. Sachdev, "A Graphical Electromagnetic Simulation Laboratory for Power Systems Engineering Programs", IEEE PES Summer Meeting, Portland, OR, July 1995.

[2] The Mathworks Inc., "MATLAB reference guide", The Mathworks Inc., October 1992.

[3] Dommel, H.W., "Digital Computer Solution of Electromagnetic Transients in Single- and Multi-Phase Networks", IEEE Trans. PAS, Vol. PAS-88, No. 4, April 1969, pp 388-399

[4] Mahseredjian, J. and Alvarado, F.; "The Design of Time Domain Simulation Tools: the Computational Engine Approach" Proceedings of the International Conference on Power System Transients, Lisbon, Portugal, Sept. 3-7, 1995, pp 493-498

[5] Kezunovic, M., Chen, Q.," A Novel Approach for Interactive Protection System Simulation", IEEE T&D Conference, Los Angeles, September 1996

[6] Kosko, B. Neural Networks and Fuzzy Systems. Englewood Cliffs, Prentice Hall (1992).

[7] The Mathworks Inc., "Fuzzy Logic TOOLBOX", The Mathworks Inc., January 1995.