COMMON MODELLING INTERFACE FOR UTILTY SIMULATION TOOLS AND ACTIVE DISTRIBUTION NETWORKS

Graeme Bathurst
TNEI Services Ltd – UK
graeme.bathurst@tnei.co.uk

John Heath
TNEI Services Ltd – UK
john.heath@ipsa-power.com

ABSTRACT

As distribution networks become more active, utility planners need to be able to adequately model the changing behaviour of their networks. This paper discusses the issues facing manufacturers, utilities and software vendors in dealing with this changing environment. A Common Modelling Interface is outlined that provides a structure that allows the same model to be used in a range of simulation platforms. Results are shown for a transient stability implementation using the same model in two widely used commercial available analysis programs.

INTRODUCTION

There has been considerable discussion recently of the need to move the operation of distribution networks towards a more active management regime, particularly to deal with increasing levels of distributed generation. In order to ensure the best use of these new technologies and to allow efficient network design, it is essential that the Utility network planners are able to simulate these new configurations using their existing simulation tools.

As a vendor of a commercial software package IPSA+, which is widely used within the UK distribution community, we are faced with the interesting challenge of providing a solution that is flexible but still easy to use, as well as a solution that protects the IPR interests of the various technology manufacturers.

There are a number of possible approaches that could be utilised to achieve the desired result namely; extending existing built-in control functions, implementing low level scripting or control macros, or utilising an existing software technology called a Plug-in.

Our experience as both developers and consultants has helped us to develop an approach that enables re-use of the same developed source code across a range of applications.

This paper gives an overview of the issues facing Utilities, Manufacturers and analysis tool vendors with regards to active distribution networks as well as a possible solution for all parties called the Common Modelling Interface.

ACTIVE NETWORK ISSUES

Distribution systems are on the whole operated as passive networks providing 'distribution' of power from bulk supply points to the individual end users. With the increasing levels of distributed generation and requirements to push networks harder, distribution networks are likely to become more active in nature.
Some examples of loadflow controllers that need to be modelled for planning studies are:

- Master-slave transformer controls
- Area load or power-flow controllers
- Demand side management systems
- Generation management systems

The issue of custom devices also enters the area of fault level (short-circuit) analysis for the modelling of devices such as active fault current limiters and inverter fed generation. Typical examples of where transient stability controllers or full machine models are required are for pitch regulated, DFIG and inverter fed wind turbines.

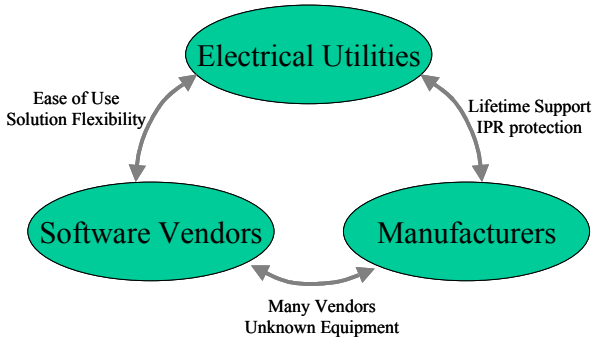

Figure 1 Principle parties and issues

There are three principal parties involved in the representation of any new technology within planning studies (Figure 1); technology manufacturers, software vendors, and most importantly the end-user Utilities. Each party has different requirements that must be resolved in order to achieve an outcome that is satisfactory for all. The following is a discussion on the development of the open-source common modelling interface that we are proposing.

Utilities

Planning engineers need to be able to represent behaviours that impact on loadflow, fault level and transient stability. They often do not have specialist modelling skills in-house, or the time required, to implement complex controllers. A preferred approach would be the provision of documented black boxes or otherwise prepared solutions.

Manufacturers

Technology manufacturers need to protect their IPR and ensure that their controllers or equipment are correctly represented. They also need to keep the amount of support of simulation tools to a sensible economic minimum.

Analysis tool vendors

Vendors need to provide and create low maintenance, flexible and usable solutions to their clients. It is key to their business to enable their users to model their real distribution networks.

EXISTING SOFTWARE TECHNOLOGIES

The issues facing power system analysis tool vendors are not uncommon to the issues faced in the wider software community. Catering for unknown new technology is difficult to achieve using built-in fixed control structures, and so a more customisable approach is required. Specifically, there are a number of standard techniques that can be used to inject external custom code into the core analysis algorithms; Scripted models, Hard linked models and Plugin models.

Scripted models

Analysis programs that provide an internal scripting or macro language often also provide the capability to write complete control models in this language. This enables a high degree of model customisation in the analysis, in particular for performing multiple simulations. The problem with scripted languages, however, is that they are interpreted not compiled, and are usually significantly slower than compiled code options (even when byte-compiled). They also do not provide any real protection of the manufacturer's IPR as the source code is usually open.

Scripting is often best suited to the customisation of sets of solutions rather than detailed equipment modelling within the core of the algorithm.

Hard-linked models

A more efficient approach is to write the model in a compiled language and link it to the core algorithm using a set of pre-defined routines and data structures (termed an Application Programming Interface or API) in the analysis program. This technique also necessitates the inclusion of a compiled-in model definition, and possibly call-backs in the analysis program. This gets over the problems of run-time speed, and protects the manufacturer’s IPR.

The nature of the API and definitions requires that the model and analysis program be linked against each other prior to run-time. This means the addition of each new model essentially creates model specific 'custom' versions of the analysis tool. Even though the models may be packaged as separate libraries, e.g. as a Dynamic Link Library (DLL) on Windows platforms, the model will only run with the 'custom' version of analysis program. This makes support and distribution of models difficult.

Plugin models

In widespread use in many other software applications is the use of plugin libraries. This is a technique whereby there is no hard-linkage between the external compiled library and the main application. This means that the application will run without the presence of the library (termed the plugin). If, however, the library is found at run-time, it is dynamically linked to the appropriate routines in the main application, thereby extending its functionality.

This tried and tested architecture seems to be the best solution for the requirements of simulating active networks. It combines the flexibility of the scripted approach, with the speed and security of hard-linked models. In fact it may be relatively simple to re-use the code of a hard-linked model by embedding it inside a plugin code harness.

Essentially it will allow controllers to be written externally to the main program using a defined plugin API, compiled into DLL's (SOs for Linux) and then linked dynamically, on demand, by the user for the network simulation. All that needs to be distributed for each model is a single DLL. For manufacturers this is particularly interesting because it protects their IPR within compiled code, but it also allows them to be sure that their controllers have been correctly implemented. It also allows the development of a common code for many different analysis platforms.

A STANDARD INTERFACE

Provision of paper based models is not viewed by TNEI as the best way forward for complex systems given the difficulties in creating a coherent document that allows third parties implement it correctly as they are often open to interpretation. Recreating the same model multiple times could be considered as wasteful of engineering time in a world where engineers are becoming a scarce resource.

For IPSA+, the decision was made to implement the Plugin approach as this was seen as providing the best solution overall. Approaches similar to this have been used in a number of other programs for user-defined models in transient stability. However, the view was taken that to resolve the active network simulation issues, the Plugin needed to be extended to all aspects of simulation, i.e. loadflow, short-circuit, transient stability, harmonics and protection.

In the process of developing this, it was recognised that it would be possible to develop a common model structure capable of being used by many other simulation tools. This Common Modelling Interface (CMI) is described in the following.

Common Modelling Interface

The proposed Common Modelling Interface (CMI) is designed to make the creation of control and equipment models in vendor-neutral form very simple. It will use the same core model code for each CMI compliant analysis program. Each analysis program vendor will supply an adaptor that allows the manufacturer's CMI model to connect directly to the vendor's software.

There is already a lot of commonality between analysis tools in terms of how models are specified. A common interface can be readily defined by the combining the common attributes and functions for each vendors model interface. The following groups of related functions are required:

Informational - model name, version, parameter name and types
Parameter setting - specify the model parameters
Simulation - initial conditions, time and step length and calculation operations

It is also necessary to define a naming convention for both the externally visible API and the internal routine names. This is to ensure that there are no name clashes when the model is used in a hard-linked vendor environment.

Once a CMI model is compiled, the manufacturer has an object file or library that can be combined with vendor supplied adaptor code to produce a re-distributable component for each vendor's program. The CMI adapters would be made publicly available from the individual software vendors so that the manufacturers can independently produce the Plug-ins. For building and testing only the CMI is required by the manufacturer (Figure 2).

CMI Benefits

Some of the key benefits for the manufacturers are the avoidance of the need to support a large number of different simulation programs, confidence that their controller is being correctly represented and protection of their IPR. For the analysis program vendors, this approach also has a significant benefits as it reduces the burden of providing and maintaining a large number of bespoke ancillary controllers, being able to support their end users. For the Utilities, this approach provides a reliable means to model a wide variety of different and new technology controllers and equipment. It means that they do not have to source specialist modelling expertise and they can be sure that the controllers or equipment have been tested and validated.

Key Requirements

There is a wide range of different analysis programs and methodologies, which each have their own advantages. For the implementation of CMI models, the following core functionality is required:

- Core code commonality between programs
- IPR protection for manufacturers
- Validity guarantees
- Documented black boxes

In addition, for Utility confidence, it is important that there is adequate documentation on the functionality of the "black-box" along with validation results. In general, Utilities do not like hidden surprises or un-explained behaviour within models.

For confidence of lifetime support and general portability, it may be necessary to place the core function code in ESCROW. This will ensure that Utilities are not left with un-maintainable or non-portable black-boxes in their analysis programs in the event of an equipment manufacturer or software vendor going out of business. An ESCROW for software is a legal arrangement where the source code is deposited with a nominated third party. The ESCROW agreement allows the release of the source code to the user in the event that the manufacturer goes out of business or fails to maintain support.

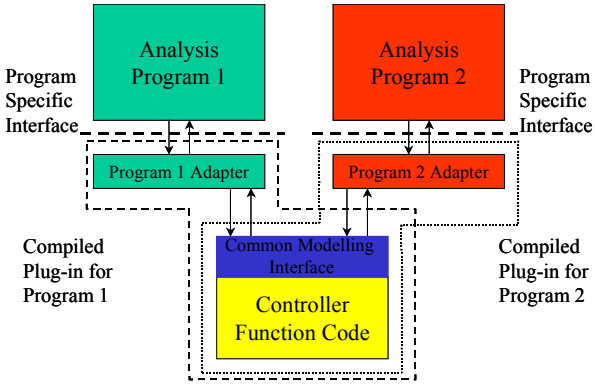

Figure 2 General Plug-in structure

Loadflow Interface

The loadflow interface requires the following functions:

- Set model parameters
- Initialisation from flat or previous solution
- Set busbar type
- Set convergence control tolerances
- Set present iteration inputs such as voltages, currents or other required solution variables
- Set present iteration sensitivities (if required)
- Calculate Jacobian entry
- Calculate & get new output variables
- Get internal variables

The loadflow implementations are likely to be more complex than the transient stability implementations as they are a part of the solution of a non-linear set of equations. In the short-term it is likely that external controllers will be implemented as ancillary equations around the primary iteration with possible variable inputs to some of the primary loadflow controls. Again though, these options should be left to the individual analysis

program vendors who are in the best position to make such decisions about implementation in their programs.

Fault level Interface

The fault level interface is more difficult to define given the wide range of short-circuit methodologies and non-linear equipment responses. The fault level interface requires the following functions:

- Set model parameters
- Set fault type & duration
- Set external impedance (if required)
- Get equivalent impedance

Transient Stability Interface

The transient stability interface requires the following functions:

- Set model parameters
- Initialisation from loadflow solution
- Set solution time and time-step
- Set present time inputs such as voltages and currents
- Calculate & get state variable derivatives
- Calculate & get new algebraic variables
- Set state variables

The program specific API then performs the linking between the simple derivative and algebraic calculations and the particular integration and time-stepping behaviour of the specific analysis program.

IMPLEMENTATION

The CMI approach has been tested and proven for transient stability analysis of wind turbine models. A common model code for transient stability simulation was created for one of the major wind turbine manufacturers. This code was validated against their detailed simulation and measurement results.

Two CMI adapters were created to link this code to two commercially available and widely used analysis programs; IPSA+ and PSS/e. The compiled form of the model for IPSA+ comprises of a DLL based Plug-in technology. The compiled form of the model for PSS/e comprises of two object (obj) files that form part of the hard-linked DSUSR.DLL file.

The complexities of the individual implementations are all managed within the program specific adapter. Once this has been created, it is then a simple matter of combining with the custom manufacturer code. Provided that the interface is correctly defined, the manufacturer or analysis program source code does not have to be a specific language.

This test showed quite successfully that the common model approach works with two quite different implementations of the same style of analysis without any significant increase in computational burden. For example, IPSA+ uses a variable time-step trapezoidal integration, whereas PSS/e uses a fixed time-step 2nd order Euler integration. Figures 3 and 4 show the active power responses of a DFIG wind turbine riding through a network fault in IPSA+ and PSS/e respectively. This is a CMI based model with the same core source code.

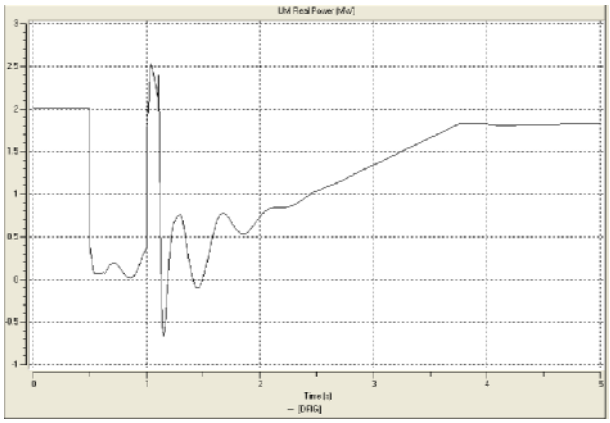

Figure 3 Illustrative MW response of a CMI based DFIG wind turbine model in IPSA

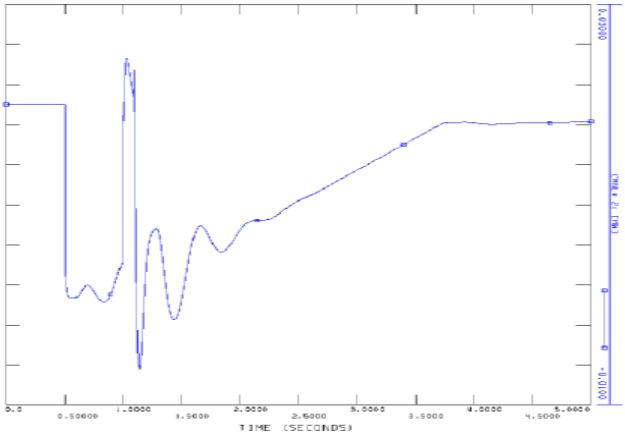

Figure 4 Illustrative MW response of a CMI based DFIG wind turbine model in PSSE

SUMMARY

A key requirement for any significant uptake of the much-discussed active distribution networks is the ability of Utilities to incorporate these controls within their network planning analysis. This paper is proposing a Common Modelling Interface approach as a possible way to minimise a number of the key issues facing the Utilities, Manufactures and Analysis program vendors.

It has been shown that it is possible to develop CMI based models that can be used with a range of power system analysis tools. The full implementation will require an industry standard to be developed for the CMI. It will also require the analysis tool vendors to open access to their analysis cores via appropriate CMI adaptors.