Automation of AEMO Model Acceptance Testing and Benchmarking – A PowerFactory and PSS/E case study

Luke Robinson
DIgSILENT Pacific
Melbourne, Australia 3004
Email: luke.robinson@digsilent.com.au

***Abstract*—Accurate, robust and functionally correct models of generator control systems are required to enable utilities and network operators to determine operational limitations and ensure stable operation of the power system. Often the models used by these organizations are developed by third parties, and without a stipulated set of simulation case studies can result in a series of updates and revisions to model source code as it is tested by model end user(s). This is particularly relevant for simulation programs that require the model writer to produce complex programs to describe control system model behavior. It is not uncommon for the end user to receive user-defined models that produce initialization errors under certain operational conditions, or become numerically unstable when tested beyond the limits of test cases performed by the model developer. There are clear benefits associated with pre-defining a set of case studies to be performed by model developers, and AEMO has thus developed new model acceptance test guidelines [1]. This places additional onus on the model developer to demonstrate model performance and robustness for a range of time-domain simulation events. This paper commences with a description of the model acceptance tests required, describes dynamic model development in DIgSILENT PowerFactory, and provides an overview of the process of converting models for use in other programs. The paper then demonstrates a procedure for automating simulation studies to demonstrate model performance. In a case study, automation scripts are developed in both PowerFactory and PSS/E to read study case parameters from a CSV file and produce results that demonstrate compliance with AEMO's model acceptance test guidelines, and additionally demonstrate alignment between the models.**

I. INTRODUCTION

The model acceptance test cases described in AEMO's Dynamic Model Acceptance Guideline are used to validate model usability and robustness. The model setup is a generic single machine infinite bus (SMIB) case with pre-defined short-circuit ratio and system X/R parameters, as shown in Figure 1. The model does not provide for a connection point specific assessment - that is carried out independently of the model acceptance tests. For this reason, machine saturation is disabled in both models, and PowerFactory d and q axis sub-transient parameters are modified in order to align with the simplified generator model typically used in PSS/E dynamic simulations. Simulation events relevant to synchronous plant excitation systems are as follows:

- Response to system faults with 0% and 70% residual voltage (defined as the voltage remaining at the step-up transformer HV terminals during application of the fault with the generator out of service).
- Step changes in AVR reference voltage to reach limiters.
- Step changes in AVR reference voltage to engage excitation limiters.
- Step changes to grid voltage.
- Step changes to grid voltage angle.

The latter two types of events can be implemented by switching in and out infinite sources with specific scheduled voltages and reference angles. For other technologies and control systems, step changes to active and reactive power set point and grid frequency may also be required.

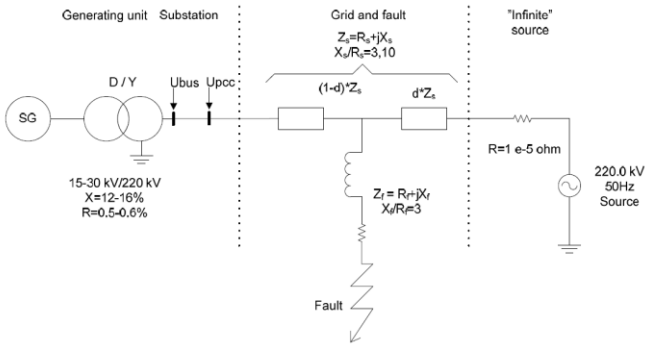

Figure 1. SMIB model setup

II. DYNAMIC MODEL DEVELOPMENT

A. *PowerFactory model development*

DIgSILENT PowerFactory facilitates control system model development using block diagrams. There is a library of standard functions, and it is rare that the user needs to modify standard blocks. This allows quick implementation of custom control system models. A common control system block is the time delay block shown in Figure 2. The user simply places pre-defined blocks such as this, and connects them together until the model has been defined. There are two key steps in the model development process:

1. Develop the control system block diagram that represents the functionality of the physical plant, thus defining how the simulation program calculates derivatives and other algebraic functions.

2. Define calculation of control system model initial conditions.

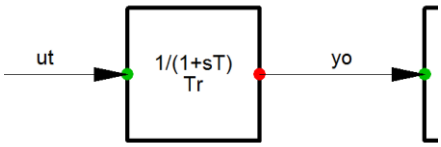

Figure 2. Time delay transfer function

The block diagram is defined as a Type, and each instance of this type (called an Element) can adopt different values for 'Tr'.

B. *Model conversion*

Conversion of models between simulation programs is simply a matter of mapping model constants, states, variables and equations for calculation of initial conditions and derivatives. In programs where there are standard functions for individual control system implementation the process is simplified.

C. *Conversion to PSS/E*

To illustrate the conversion process, translation of a simple time delay block from PowerFactory to PSS/E is described. In PowerFactory, the time delay block is represented by the block shown in Figure 2. Within the block, calculation of the transfer function is according to the equations below, where x is the state variable, x. is the derivative of the state variable, yi is the input to the block, T is the time constant, and yo is the block output. All instances of time delay blocks in a model use these same variable definitions, it is only the value substituted for T, and the particular name given to the state variable that are unique. All internal parameters are automatically available for reporting and plotting, but must be selected as variables for recording.

x.=(yi-x)/T

yo=x

In PSS/E the block can be implemented identically. STATE(K) is the state variable, DSTATE(K) is the derivative of the state variable, YI is the input to the block, CON(J) is the time constant, and YO is the block output. This code is duplicated for each instance of a time delay block, and the input, output, state, and constant are renamed in each instance. In PSS/E the convention is to name constants, states and variables associated with each model sequentially. The next reference to a state would use the term K+1, the next CON would use J+1, etc. To access the internal parameters for later reporting or plotting, they must be assigned to a VAR. The convention for naming of VAR's is to use indices commencing with VAR(L).

DSTATE(K)=(YI-STATE(K))/CON(J)

YO=STATE(K)

For the case where the time delay block is connected to a terminal voltage input signal, the block is initialised by setting the state equal to the input signal. In PowerFactory this is:

inc(x) = ut

Similarly in PSS/E it is:

STATE(K)=ECOMP(I)

Where in PowerFactory 'ut' is commonly used for the terminal voltage signal, and in PSS/E it must be ECOMP(I).

Further to the dynamic model conversion, machine parameters must be converted from 'accurate' to 'classical' when converting from programs such as PowerFactory and TSTAT (used by TransGrid) to PSS/E format. The procedure for machine parameter conversion is described in Kundur [2].

III. MODEL TESTING

Prior to performing model acceptance tests, open loop tests were completed with sinusoidal input signals applied to verify the response of individual blocks. This is not strictly necessary, but is a useful approach to verifying model functionality, particularly of non-windup limiters. There are different approaches to implementation of non-windup limiters, and the approach recommended in Kundur [2] and IEEE 421.5 [3] was adopted in this case. For small disturbances alternative implementations would yield similar results, but for large disturbances the differences can be significant. The implementation adopted is shown in Figure 3.

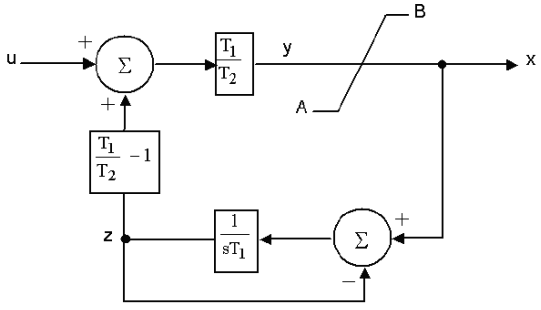

Figure 3. Implementation on lead-lag non-windup limiter

IV. STUDY CASES AND AUTOMATION

A. *Study cases*

The study cases described in section I were considered for model acceptance testing of an ABB Unitrol F excitation system, which included automatic voltage regulator, power system stabiliser, and under- and over-excitation limiters. Scripts in both DIgSILENT Programming Language (DPL) for PowerFactory and Python for PSS/E refer to study case parameters stored in a CSV file. Table 1 shows the parameters associated with a fault case of 0.12 s duration at the transformer HV terminals. Note the parameter for acceleration factor 'accel' is only relevant to PSS/E simulations.

Table 1 – Study case 1

Item	Duration	Residual	SCR	XR	Power	Step	Accel	Reactive
1	0.12	0	5	3	1	2	1	0

B. *Automation scripts*

The DPL script reads simulation case parameters from the CSV file and modifies network source impedance, fault impedance (for fault cases) and simulation events accordingly. The script can optionally read in simulation results from other programs or from commissioning tests for comparison. After each simulation, a WMF file is exported that includes a plot page from PowerFactory with the key quantities of interest:

- Machine terminal voltage.
- Machine active and reactive power.
- Machine angle (of particular interest for the system angle step change study cases).
- Machine field voltage and field current (of particular interest given that these quantities are used as input to the AVR and over-excitation limiter respectively).

The Python script similarly reads in simulation parameters from the CSV file and iterates through the study cases.

Automation of model assessment in this way significantly reduces the time required to run simulation cases. Although there is overhead associated with developing the scripts, they may be reused for subsequent revisions of the model, or for future benchmarking activities. There are clear benefits of this approach over running hundreds of study cases manually.

V. SIMILATION RESULTS

Figure 3 shows simulation study results for case 1. The results show a numerically stable response, and additionally show almost identical responses in PowerFactory and PSS/E.

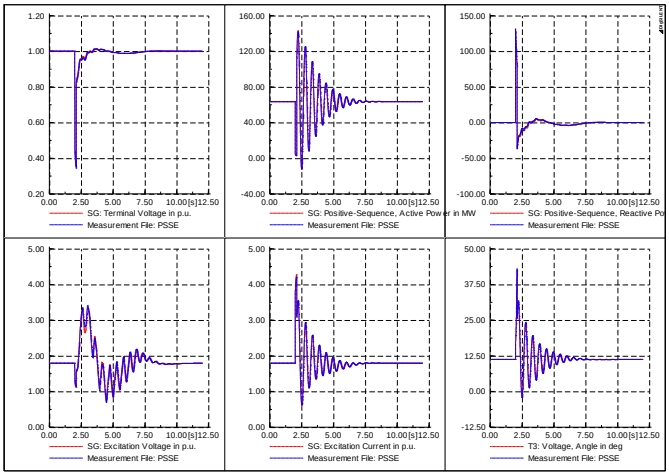

Figure 4. Case 1 study results

VI. SUMMARY AND CONCLUSION

The development of control system models for dynamic simulation has been presented, and the procedure for automating model performance assessments with respect to the AEMO guidelines has been described. PowerFactory is a powerful integrated simulation program that facilitates assessment of transient performance, small signal stability, power quality, protection coordination, and also provides other calculation and database management functions. In part, the objective of this paper is to demonstrate the relative ease with which models can be converted between platforms. This provides engineers with additional flexibility in their software choice. For example, consider the choice of an engineer to either:

1. Develop a control system model in PowerFactory and perform time-domain simulations, eigenvalue analysis, power quality assessment, and protection coordination studies. Subsequently convert the model to a format required by the end user, such as PSS/E.
2. Draw a block diagram in a graphical program. Develop detailed model source code based upon this block diagram. Subsequently convert the model to a program suitable for eigenvalue analysis (e.g. PowerFactory or MUDPACK), and to formats suitable for power quality assessment and protection coordination studies.

This paper has also demonstrated an automated procedure for thoroughly testing model functionality and performance that could be used to assess conversion of models from and to PowerFactory format.

REFERENCES

[1] AEMO, *Dynamic Model Acceptance Guideline*, 2013.
[2] Kundur, P., Power System Stability and Control, 1994.
[3] IEEE, *IEEE Recommended Practice for Excitation System Models for Power System Stability Studies*, 2005.