Recommendations and comments from OPAL-RT Technologies to AEMO guideline on model compatibility (Feb 8, 2023)

Submissions to the first round of consultation are welcome until **10 February 2023**, via email to PSMGReview@aemo.com.au

See AEMO | PSMG Review Consultation

OPAL-RT RECOMMENDATIONS on the document - Amendments to the Power System Model Guidelines published on December 2nd 2022.

Section 3.2 of the document describes that manufacturers (OEM) must provide controls in the form of DLLs compatible with a new open source interface that AEMO and NSP will develop. OPAL-RT thinks AEMO should rather use a standard interface that already exists and that is supported by various software vendors. Ex:

- CiGRE B4.82 WG - Use of real code in EMT Models for Power System Analysis
- IEC 61400-27-1 - 2015-02 - Wind turbines – Part 27-1: Electrical simulation models – Wind turbines
- Simulink Coder / Embedded Coder
- FMI standard (https://fmi-standard.org/)

COMMENTS

Normally standard interfaces will provide the following functionalities, which doesn't seem to be included in the method proposed by AEMO.

- DLL control block must provide a set of functions to perform the main task of the simulation execution flow:
  - initialize the control code,
  - update states and derivatives,
  - validate parameter values,
  - change parameter values before or during simulation,
  - take and restore snapshot of the states of the control.

  Note : Documentation must also clarify when each function is executed in the PSCAD context (DSDYN, DSOUT, DSDYN_BEGIN, DSOUT_BEGIN)

- DLL control must provide a standard interface to get/set parameters that have different data types (double, integer, Boolean,) and data size (scalar, vector, matrix).
- DLL control must handle multiple instances of the same control within the same DLL. We expect having a data structure for each DLL instance.
- DLL control must support being configured with a given sample time independent of the control itself. Sampling time will be defined in the PSCAD model. This will allow to have multiple controls in the same models and share the same time step. It is complex to have multiple sampling time in the same model when each control doesn't share the same base rate.
- DLL control must be able to produce interpolated firing gate signal **and** duty cycle that could then be used by the simulation tools for feeding converters modeled with average and detailed switching functions so the solver could compensate for events occurring between time steps.

- DLL control should not contribute to the PSCAD electrical matrix using the branch segment.
- DLL control should provide a function to take and restore snapshot of the control, so a simulation could be started from a specific network condition.
- DLL control could be initialized using the result values of a load flow simulation. This way control logic and states could have the proper initial values according to the loadflow.
- Control should be provided as 32 bits and 64 bits DLL to support future PSCAD 64 bits applications. We believe that having only the 32 bits version of the DLL is risky at long term.

Additional constraints based on our experience and knowledge need to be also considered:

- If a plant model contains multiple control DLL, they should not communicate data using global variables or shared memory. They should communicate data using standard signals.
- Performance of DLL should be optimized to make sure it runs fast, ideally faster than the sample time on an average processor technology.
- DLL should not rely on operating system functionality like multithreading, locking, semaphore, and should not allocate memory during simulation.
- Plants should minimize the number of control DLL to integrate inside the final plant models.

Figure 1 below shows a quick comparison of the various interfaces available.

	AEMO	CIGRE B4	IEC 61400-27-1	FMU	Simulink
Execution Flow					
Instantiate / Initialize	n	y	y	y	y
Initialize control from load flow results	n	n	n	n	n
Validate parameters	n	y	y	y	y
Update states and derivatives	n	y	y	y	y
Output	y	y	y	y	y
Get/Set parameters	n	y	y	y	y
Take/Restore snapshot	n	y	y	y	y
Terminate / Free Instance	y	y	y	y	y
GetModelInfo	n	y	y	y	n
Parameters					
Double data type	n	y	y	y	y
Integer data type	n	n	n	y	y
Boolean data type	n	n	n	y	y
Vectors	n	n	y	y	y
Matrices	n	n	n	y	y
Units	n	y	n	y	n
Signals					
Integer data type	y	n	n	y	y
Double data type	y	y	y	y	y
Boolean	n	n	n	y	y
Vector	n	y	y	y	y
Matrix	n	n	n	y	y
Unit system	n	y	n	y	n
Multirate inputs/outputs	n	n	n	y	y
General configuration					
Configurable fixed sample time	n	y	y	y	y
Dll interface version number	n	y	y	y	y
Model version number	n	y	y	y	y
Other capabilities					
Logging / Verbosity	n	y	y	y	y
Error reporting	n	y	y	y	y
Provide a standard API to access the control (C/C++)	n	n	n	y	y
Provide a configuration file describing the DLL interface (i/o, param, …)	y	n	n	y	y
Standardized "packaging and distribution" management system (source code, headers, binaries, resources, doc,)	n	n	n	y	n
Support of multiple control instances within the same DLL	n	y	y	y	y
Interoperability - 32/64 bits library	possible				
Produce interpolated firing signals and duty cycle	possible				

Figure 1 - Comparison of existing standard interface for integrating control into simulation software