УДК 681.5:621.311.24:004.941 DOI 10.36910/775.24153966.2025.82.13

**Н.В. Курилко, Р.М. Федоришин**

*Національний університет «Львівська політехніка»*

## МОДЕЛЬНО-ОРІЄНТОВАНЕ ПРОЄКТУВАННЯ ТА МІЖПЛАТФОРМНЕ ВПРОВАДЖЕННЯ КОНТРОЛЕРІВ ВІТРОВИХ ЕЛЕКТРОСТАНЦІЙ

***У статті запропоновано модельно-орієнтований підхід до розробки систем керування вітровими електростанціями (ВЕС), що забезпечує адаптацію алгоритмів до технічних і регуляторних вимог, зокрема до мережевих кодексів. Основна увага приділена створенню контролера ВЕС у середовищі MATLAB/Simulink з автоматичною генерацією коду (C/C++, Structured Text) для розгортання на вбудованих пристроях і ПЛК. Запропоновано створення DLL-бібліотек відповідно до IEC 61400-27-2:2020 (Annex G) для сумісності з симуляторами PowerFactory та PSCAD. Наведено результати валідації моделі (бібліотеки). Визначено переваги такого підходу для прискорення розробки й сертифікації контролерів ВЕС, а також визначено перспективи інтеграції з аероеластичними моделями та цифровими подвійниками вітрових турбін та іншого обладнання ВЕС.***

***Ключові слова:*** *вітрова електростанція, модельно-орієнтоване проєктування, Simulink, PSCAD, DIgSILENT Power Factory, контролер ВЕС, IEC 61400-27-2, DLL, генерація коду, сертифікація, цифровий двійник.*

***N.V. Kurylko, R.M. Fedoryshyn***

## MODEL-BASED DESIGN AND CROSS-PLATFORM DEPLOYMENT OF WIND POWER PLANT CONTROLLERS

***This paper presents a model-based development approach for wind power plant (WPP) control systems, emphasizing the adaptation of control algorithms to meet specific technical and regulatory requirements, including grid codes. The WPP controller is implemented in the MATLAB/Simulink environment, enabling automatic code generation in C/C++ and Structured Text for deployment on embedded platforms and programmable logic controllers (PLCs). To ensure interoperability with industry-standard simulation tools such as DIgSILENT PowerFactory and PSCAD, the approach includes generating dynamic link libraries (DLLs) conforming to the interface specification defined in IEC 61400-27-2:2020 (Annex G). The paper details the verification process by comparing the DLL-based implementation against the original Simulink model and demonstrates successful integration within external simulation environments. The proposed methodology accelerates the development, testing, and certification of WPP controllers. Future work includes extending the framework to support integration with aeroelastic models and digital twin platforms.***

***Key words:*** *wind power plant, model-based design, Simulink, PSCAD, PowerFactory, WPP controller, IEC 61400-27-2, DLL, code generation, certification, digital twin*

**Introduction.** Modern wind power plants (WPPs) encounter a range of technical, operational, and regulatory challenges when developing and implementing control systems [1]. One of the most critical tasks is ensuring compliance with grid-code requirements that govern active-power limitation, reactive-power control, voltage stabilisation, and frequency response. Because these requirements differ across regional standards, designing universal control algorithms becomes significantly more complex [2]. Furthermore, the diversity of WPP architectures—turbine numbers, electrical-mechanical characteristics, and layout configurations—adds another layer of difficulty for standardised solutions [3].

Direct testing of new control algorithms on real installations demands substantial resources owing to long validation cycles and strict safety rules [4]. A model-based design approach therefore gains relevance: it allows engineers to adapt algorithms to a specific WPP configuration and to conduct exhaustive virtual tests before field deployment.

During certification, developers are often required to supply controller models compatible with leading industry simulation suites such as DIgSILENT PowerFactory, PSCAD, and MATLAB/Simulink [5]. The international standard IEC 61400-27-2:2020 (model validation) [6] and national guidelines such as FGW TR4 Rev. 9 (Germany) [7] explicitly stipulate that WPP models be provided as dynamically linked libraries—e.g., Windows DLLs or shared objects on Linux-based platforms—to facilitate verification, testing, and certification.

Current development workflows therefore rely on unified software frameworks that can automatically generate these libraries, greatly simplifying integration into diverse simulation and hardware-in-the-loop environments [8].

A parallel objective of WPP control is to maximise energy yield while reducing structural loads on turbine components. Meeting this goal requires aeroelastic models that capture the wind-structure interaction with high fidelity. However, integrating controllers with aeroelastic-capable tools such as OpenFAST or FAST.Farm remains challenging because no universally accepted interface exists for real-time co-simulation [9].

**System Requirements Definition**. Considering the technical and operational challenges faced by modern wind power plant (WPP) control systems, a comprehensive model-based approach to the development of control algorithms is proposed. Below, we define the main requirements for the proposed control system.

The WPP control algorithm should be developed in an environment that allows modeling the behavior of real-world systems and conducting initial functional testing, taking into account local requirements and the specific parameters of the particular WPP. This approach aligns with the principles of model-based design, which are widely applied in wind energy engineering [10]. A critical requirement is the ability to quickly adapt control algorithms through visual programming to different grid codes, WPP architectures, turbine types, and auxiliary equipment. An important requirement is also the integration of the developed algorithms with leading simulation environments commonly used in the energy sector, such as DIgSILENT PowerFactory, PSCAD, and MATLAB/Simulink [6].

The proposed solution should support a modular architecture for control algorithms, simplifying their adaptation to real-world operational changes and facilitating updates to individual system components. A key condition for practical deployment is the ability to port the developed algorithms to industrial target platforms, particularly to industrial controllers executing C/C++ code or to programmable logic controllers (PLCs), in accordance with IEC 61400-27-2:2020 [6] and technical recommendations for model certification [7].

**Analysis of recent research and publications.** The development pipeline based on *dynamically linked libraries* (DLLs) – model creation in MATLAB/Simulink, automatic C/C++ code generation, and compilation into DLLs that follow Annex G of IEC 61400-27-2 – is documented by Chmielewski et al. [8] and remains the de-facto recipe for producing controller models that run unchanged in PSCAD or DIgSILENT PowerFactory. Experiments on a room-temperature control loop showed that the DLL produced identical temperature and current traces in all three simulators, confirming cross-platform consistency.

An important extension of this idea is to couple aero-elastic farm simulators with controller prototypes in Simulink. A full MPI/MEX interface between FAST.Farm and MATLAB/Simulink has been released open-source (FASTFarm2Simulink) and described in detail by its authors; the repository includes example co-simulations of a 10-turbine farm and benchmarks a speed-up factor of 100× relative to high-fidelity CFD models [9]. A separate academic implementation – Smits (2023) – gives a step-by-step configuration of the same interface and demonstrates active-power control of a 10-turbine array under time-varying inflow [11].

Because DLLs must be verified against *all* target simulators, IEC 61400-27-2 has shifted research attention from mere code portability (IEC 61400-27-1) to rigorous model-validation test cases. Comparative studies confirm that DLL controllers compiled from one source achieve virtually identical transient responses in PSCAD, DIgSILENT and PSS®E, provided Annex G entry-points are respected [8].

Parallel and distributed execution is another fast-moving topic. The FASTFarm2Simulink/MPI framework [9] parallelises the super-controller, individual turbine models and wake dynamics into separate processes, cutting multi-hour sequential runs down to minutes – a decisive advantage when tuning large parameter sets.

Beyond controller verification, researchers are linking aero-elastic solvers to layout-optimisation and wake-model tool-chains written in Python. Rodrigues et al. (2024) integrate OpenFAST-derived loads into the *PyWake / TOPFARM* stack and show that gradient-based, parallel optimisation trims computation time for 150-turbine layouts by two orders of magnitude [13].

The same push for unification underlies the rise of digital-twin platforms. Branlard et al. (2024) assemble a real-time twin for the TetraSpar floating prototype by merging OpenFAST linearisations with SCADA streams; the twin predicts tower-fore-aft fatigue loads within 10 % of field measurements, enabling proactive maintenance scheduling [14].

Trend summary

- Model unification. Annex G-compliant DLLs now compile once and run anywhere, provided validation tests (IEC 61400-27-2) are passed.
- Cross-platform co-simulation. MPI-based links let Simulink controllers drive FAST.Farm or OpenFAST in parallel, accelerating design-of-experiments runs.
- Python tool-chain coupling. OpenFAST outputs feed PyWake/TOPFARM for gradient-based farm layout or control optimisation.

- Digital twins. Physics-based twins that fuse OpenFAST models with live SCADA data are moving from concept to validated prototypes.

These developments reinforce our choice to follow a model-based design workflow: generate Annex G DLLs from a single Simulink source, validate them once, and reuse the same implementation in PowerFactory, PSCAD, hardware-in-the-loop benches, and eventually in an on-line digital twin of the plant.

**Development of the Solution.** To implement the above-mentioned requirements, this work proposes the development and testing of wind-power-plant (WPP) control algorithms within the MATLAB/Simulink environment. The choice of this platform is based on two key advantages. First, Simulink is a scalable and flexible tool that enables modelling of a wide range of tasks — from real-time control-system analysis and hardware-in-the-loop (HIL) testing to fault-condition analysis, power-system integration, algorithm optimisation, and comprehensive model validation [13]. Second, the platform supports automatic code generation in C/C++ and Structured Text, providing a direct transition from the modelling phase to deployment on embedded or industrial controllers [14]. This approach significantly shortens the development cycle, reduces errors during control-logic transfer, and ensures consistency between the model and its implementation.

The proposed concept is implemented as a modular model, illustrated in Fig. 1. The central component of the system is the Wind Power Plant Controller, which coordinates the operation of all wind turbines, responds to commands from the grid operator, and ensures that the plant operates in compliance with relevant grid-code requirements. The controller generates control signals based on the current state of the plant, meteorological conditions, and dispatch constraints, aiming to optimally distribute power, regulate reactive power, and maintain voltage and frequency levels at the point of common coupling.

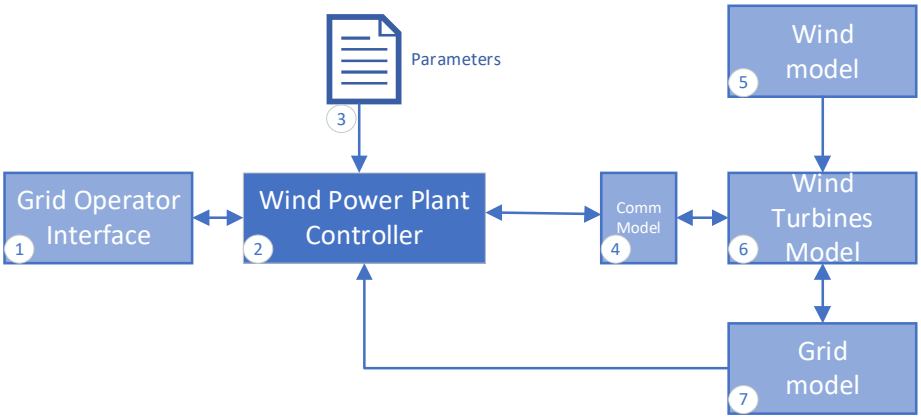


*Fig. 1.* **Structural diagram of the wind-power-plant model**

The diagram (Fig. 1) includes the following key components:

1. **Grid Operator Interface** model, which enables communication for transmitting constraints and control commands to the WPP controller, as well as receiving feedback on the current state of the power plant.
2. **Centralised WPP controller**, which directly manages wind-turbine operation by distributing power and performing the necessary control functions.
3. WPP-controller **Parameters**, enabling real-time adjustment of system settings in accordance with current grid-code requirements or internal technical constraints of the plant.
4. **Communication** interface **model** between the WPP controller and turbines, which enables simulation of signal delays and analysis of their impact on overall system performance.
5. Wind-field model **(Wind Model)**, which generates wind-speed and -direction signals needed for assessing wind-turbine performance and control-system operation.
6. Aggregated **Wind-Turbines Model** (Wind Farm Model), which simulates mechanical and electrical processes within turbines under various operating modes and load conditions.
7. **Grid model**, which enables analysis of grid parameters — voltage, frequency, current, active and reactive power — under both normal and fault conditions.

The proposed model, implemented in Simulink, provides a comprehensive approach for testing wind-power-plant control algorithms. It allows for thorough assessment of the effectiveness, robustness, and compliance of control strategies with applicable regulatory requirements. This approach significantly

accelerates the design, verification, and deployment phases of WPP control systems, which is crucial for improving the reliability and adaptability of modern wind-energy installations.

**Integration of the WPP Controller Model into Specialized Simulation Environments.** Annex G of the IEC 61400-27-2:2020 standard defines an extended interface for data exchange between models implemented in different simulation environments. This annex specifies the C programming language as the primary implementation language, although other languages are also permitted. A model that supports this interface can be compiled as a dynamic (DLL) or static library and integrated into any software environment that complies with the interface requirements [6].

Such models are most commonly compiled as DLLs for the Windows operating system. However, with proper compiler configuration and build-environment setup, they can also be adapted for use with other operating systems or embedded platforms. This approach is widely used to protect intellectual property and to ensure reproducibility when transferring models across simulation tools.

A key advantage of the Annex G interface is its flexibility, which allows the implementation to be tailored to the technical and functional needs of the target environment without modifying the underlying control logic.

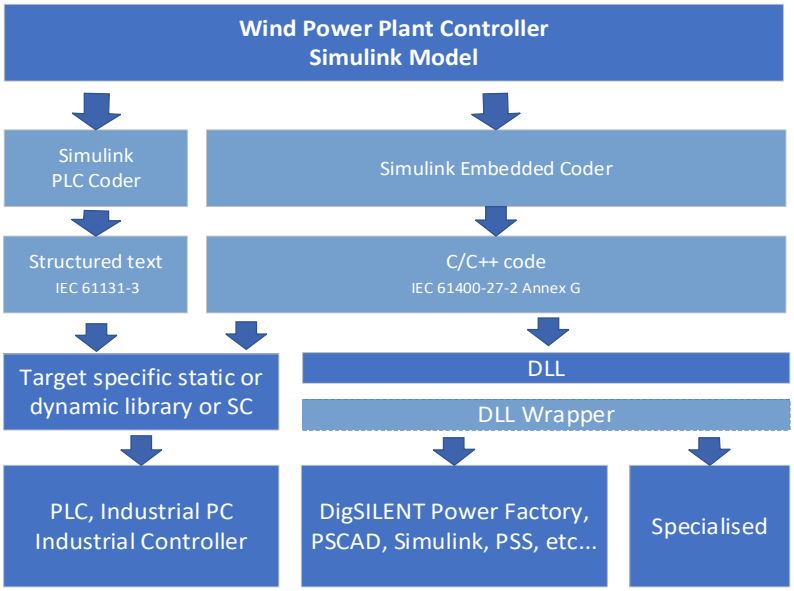


*Fig. 2.* **Code-generation workflow for the WPP controller**

Figure 2 shows the workflow for generating code from a Simulink-based WPP-controller model. Two core toolboxes are used:

- **Simulink PLC Coder** automatically produces Structured-Text (ST) code compliant with IEC 61131-3 [15, 13]. The ST code can be loaded directly into industrial programmable-logic controllers, enabling real-time deployment of the control algorithms.
- **Simulink Embedded Coder** generates C/C++ code for embedded systems [14]. To maximise portability, the generated code follows the unified interface of Annex G in IEC 61400-27-2 [6], so that the same source can target multiple hardware platforms.

The resulting C/C++ code may be compiled as a static or dynamic library for industrial computers or other controllers. The same source can also be built as a Windows DLL, enabling plug-and-play integration with simulation suites such as DIgSILENT PowerFactory [17], PSCAD [19], or even another Simulink session.

In summary, this workflow supports a seamless transition from a high-level Simulink model to industrial implementations: from Structured-Text code for PLCs to dynamic libraries for DIgSILENT/PSCAD/Simulink, ensuring a single, validated code base throughout the entire development cycle.

**Integration of the WPP Controller Model into DIgSILENT PowerFactory** Power-generation equipment operated by transmission-system operators (TSOs) is routinely modelled in DIgSILENT PowerFactory. German TSOs (50Hertz, TenneT, Amprion, TransnetBW) and several other ENTSO-E members—including Elia, RTE, REE and EirGrid—explicitly recommend PowerFactory for dynamic studies in their technical guidelines [17]. National Grid ESO in the United Kingdom follows the same practice during grid-code compliance assessments. PowerFactory owes its popularity to flexible RMS

solvers for electromechanical analysis [17] and EMT solvers for detailed fault studies [18]. Since the 2022 release, external models compiled as DLLs that implement the Annex G interface of IEC 61400-27-2 [6] can be attached directly to objects of type *TypMdl*.

Integration steps are straightforward: Create a new *TypMdl* object; Point the *File* field to the DLL; Populate parameter and initial-value tables. Map the model's inputs and outputs. Because no source-code changes are required, any controller that conforms to Annex G [6] plugs into PowerFactory "as-is."

**Integration of the WPP Controller Model into PSCAD.** PSCAD is a specialised EMT platform that TSOs and OEMs use to validate WPP controllers under both normal and stressed grid conditions. Unlike RMS-oriented tools, PSCAD performs fixed-time-step electromagnetic calculations, capturing fast transients that are critical for converter-interfaced wind turbines [19].

TSOs such as 50Hertz, TenneT, TransnetBW and EirGrid rely on PSCAD for controller testing, harmonic analysis and Fault-Ride-Through (FRT) evaluation. Its component-level electromagnetic detail helps detect over-voltages, resonances and converter–grid interaction issues early in the design phase [19].

During certification, PSCAD typically assesses FRT, harmonic content, transient stability and controller performance to verify compliance with IEC 61400-27-2 [6]. Recent PSCAD versions allow C/C++ DLLs, but the tool does not natively recognise the Annex G interface. Therefore, an integration wrapper is required. The wrapper exposes three entry points—Init(), Step() and Terminate()—that PSCAD calls at every time step, translating signals between the Annex G convention and PSCAD's internal format [19]. This extra layer ensures that Annex G-compliant controllers can still be evaluated in PSCAD without rewriting the underlying control logic.

**Case Study.** To validate the proposed strategy for developing a wind-power-plant (WPP) control system, we built a modular simulation model in MATLAB/Simulink. The architecture follows IEC 61400-27-1 [5] and comprises five key blocks (Fig. 3):

1. **Wind Model** — generates individual wind-speed signals per turbine and the farm-average speed. The model captures both long-term climatology and short-term turbulence, enabling realistic controller tests under variable inflow conditions [10].
2. **Grid Operator Interface** — emulates dispatch commands, issuing active-power limits (CTRL_P_Limit_pct), reactive-power set-points (CTRL_Q_Sp), mode selection (CTRL_Q_Mode) and global stop commands (CTRL_StpCmd).
3. **Grid Model** — computes point-of-common-coupling (PCC) quantities: active power (PCC_P), reactive power (PCC_Q), frequency (PCC_Frq) and voltage (PCC_V).
4. **Wind Park Controller** — a centralised algorithm that distributes power among turbines according to their availability and PCC conditions. Outputs are per-turbine active-power limits (ToWTs_P_lim), reactive-power set-points (ToWTs_Q_sp) and stop signals (ToWTs_Stp).
5. **Wind Farm Model** — aggregates mechanical and electrical dynamics of individual turbines, feeding real-time states (WTs_PCtrl_State, WTs_QCtrl_State) back to the controller.

The closed-loop arrangement allows us to test power-sharing, voltage/frequency support and fault-ride-through logic without hardware. By mirroring dispatch scenarios and grid events in a single Simulink workspace, the workflow shortens controller tuning cycles while ensuring full traceability to IEC 61400-27-1 requirements [5].