Digitalization of Large-Scale Testing Facilities for the Wind Industry: DIGIT-BENCH Digital Twin

Digitalization of Large-Scale Testing Facilities for the Wind Industry: DIGIT-BENCH Digital Twin

Elif Ecem Baş[1*], Giuseppe Abbiati[2], Cláudio Ângelo Gonçalves Gomes[3], Uwe Jassmann[1], Peter Gorm Larsen[3]

[1] R&D Test Systems, Hinnerup, Denmark
[2] Department of Civil and Architectural Engineering, Aarhus University, Aarhus, Denmark
[3] Department of Electrical and Computer Engineering, Aarhus University, Aarhus, Denmark
[*]E-mail: eeb@rdas.dk

Keywords: test bench; digital twin; virtual testing.

Abstract

Testing of large wind turbine components plays a central role in delivering reliable yet cost-effective technology. However, these experiments are often lengthy and costly. Fatigue testing of a wind turbine blade might take up to 12-14 months, whereas highly accelerated lifetime testing of a nacelle demands 6-8 months. The exchange of simulation models and data between OEMs and test facilities is recognized as a critical factor in the planning of an experimental campaign. In fact, OEMs are typically very protective of their industrial secrets, and sharing such sensitive information may constitute a threat. It follows that the use of simulation models to enable more effective experimentation is not pursued efficiently.

Digital twins are emerging as a key enabling technology to improve the operation & maintenance of test benches for the wind industry. A digital twin combines physical systems and their digital models into a cyber-physical system to provide functionalities that cannot be attained by either physical or digital assets independently. The seamless integration of the test bench with digital models offered by a digital twin is expected to enhance the interaction between OEM and test bench operators.

This proceeding illustrates the status of the development of the DIGIT-BENCH digital twin developed by R&D Test Systems (R&D) and Aarhus University to serve large-scale test facilities for the wind industry. The digital twin utilizes FMI-based co-simulation to enable the coupling of physical/digital components in an industrial-secret-friendly environment. The digital twin concept is demonstrated on a 2-degrees-of-freedom test bench installed at Aarhus University.

1. Introduction

To reduce the levelized cost of energy, wind turbines (WTs) are getting bigger and bigger. Testing of large WT components plays a central role in delivering reliable yet cost-effective technology. However, these experiments are often lengthy and costly. Fatigue testing of a wind turbine blade might take up to 12-14 months, whereas Highly Accelerated Lifetime Testing (HALT) of a nacelle demands 6-8 months. The exchange of simulation models and data between Original Equipment Manufacturers (OEMs) and test facilities is a critical factor in the planning of an experimental campaign. In fact, OEMs are typically very protective of their industrial secrets, and sharing such sensitive information may constitute a threat. It follows that using simulation models to enable more effective experimentation is not pursued efficiently. Moreover, personnel training and test rehearsal typically conceal non-negligible risks of damaging the test bench. In response to these challenges, Clemson University's Wind Turbine

Drivetrain Testing Facility at the South Carolina Electric & Gas Energy Innovation Center in North Charleston pioneered the adoption of system-level simulations to improve the operation of two nacelle test benches [1]. Along the same lines, system-level simulation tools have been developed for the drivetrain test bench of the DyNaLab of the Fraunhofer Institute for Wind Energy Systems IWES, Bremerhaven, Germany [2].

The Digital Twin (DT) paradigm is emerging as a natural evolution of pure numerical simulations to improve the Operation & Maintenance (O&M) of test benches for the wind industry. A DT combines physical systems and their digital models into a cyber-physical system. A true DT is obtained when physical and digital assets influence each other (e.g., a digital model updated with experimental measurement is used to control the physical asset). A DT provides functionalities that cannot be attained independently by either physical or digital components. When data exchange is from physical to digital assets only, the term *digital shadow* is more correct. However, in the authors' experience, the term DT is typically used to indicate a digital shadow [3]. Besides O&M, the seamless integration of the test bench with digital models offered by a DT enables the adoption of hybrid numerical-experimental testing methodologies such as *virtual testing* [2] and *hardware-in-the-loop testing* [4].

This proceeding illustrates the status of the development of DIGIT-BENCH DT. DIGIT-BENCH is a DT framework for large-scale test benches for the wind industry that is being developed across R&D A/S Test Systems and Aarhus University. Section 2 provides an overview of the DT's architecture and showcase virtual testing as an application example. Section 3 draws conclusions and future perspectives.

2. DIGIT-BENCH Digital Twin

The block diagram of Figure 1 illustrates the architecture of DIGIT-BENCH DT. As one can see, the DT consists of three main building blocks: the *physical space*, the *digital space*, and the *database*. The physical space is a collection of physical assets, such as the reaction frame, the Test Loading Unit (TLU), and the Device Under Test (DUT). The digital space is a collection of computational models crafted to simulate/visualize the physical space. Numerical simulations are constructed to predict the behavior of a physical asset, while CAD/BIM models are used to visualize a physical asset. The database is used to store data from both physical and digital space. The DT is meant to support an application toolchain for the test facility. As of today, the focus of the development team is on *virtual testing* and *hybrid testing* applications. Based on Wagg et al. [5] classification, DIGIT-BENCH DT is a *simulation* DT. However, the aim is to reach the *intelligent* level by monitoring the physical space and updating the digital space accordingly.

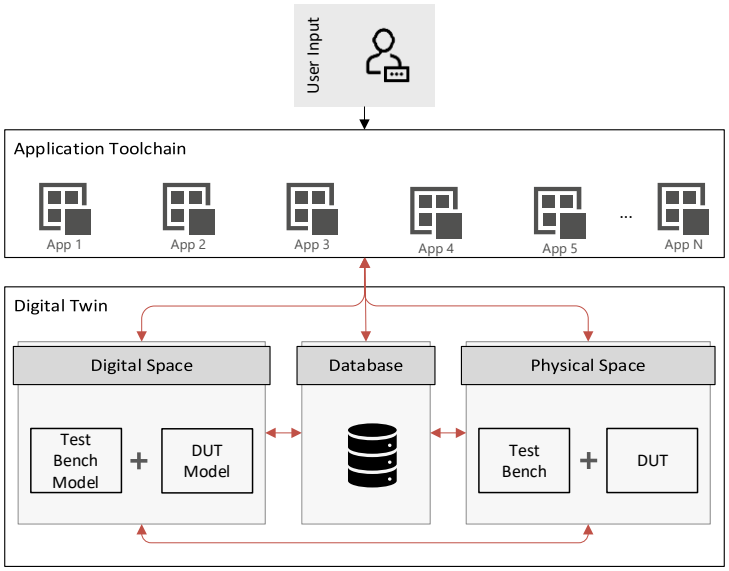


Figure 1 DIGIT-BENCH Digital Twin architecture.

Two important aspects substantially condition the specific choices associated with implementing the proposed architecture, which are commented on herein.

1. Firstly, in an experimental campaign, it is often the case that the OEM, which provides the DUT, is unwilling to disclose entirely both their numerical models and the data produced by testing. In fact, disclosure is perceived as a threat to protecting industrial secrets.
2. Secondly, simulation software licensing models that favor customer lock-in[1] and/or expensive fees expose the test facility to additional risk, which, in the authors' opinion, is a major bottleneck to adopting DTs.

In response to the challenge posed by (1), the application toolchain of DIGIT-BENCH DT relies on *co-simulation* as defined in the Functional Mock-up Interface (FMI) [6]. In response to the challenge posed by (2), open-source tools are examined for the technology stack to implement co-simulation.

In relation to (1), Co-Simulation is a methodology for crafting simulations by coupling simulation models crafted using domain-specific tools. The FMI standard defines how to encapsulate simulation models into standardized containers to form Functional Mock-up Units (FMUs) and how FMUs exchange signals. The FMI standard is organized under the roof of the Modelica Association [7]. Currently, the FMI standard is officially supported by 170+ simulation tools and provides templates for supporting FMU import/export in several programming languages. In essence, the FMI standard defines interface and container specifications that an FMU must comply with to be linked with other FMUs within a Co-simulation. In practice, an FMU is a *.ZIP file that encapsulates either the binaries of a simulation model or the Application Programming Interface (API) necessary to communicate with a physical system (e.g., a control system), and an *.XML file describing the FMU input/output variables. An FMU can be either generated using the FMU exporter of specific simulation software – when available – or programmed directly using existing software libraries such as UniFMU [8]. Co-simulation can be distributed over the internet, meaning that the stakeholder running the Co-Simulation might not have physical access to all FMUs. This architecture is useful to avoid disclosing models among stakeholders when industrial-secrete protection is critical (especially important for the DUT). Noteworthy, Co-simulation was mentioned already in [1] as a possible solution to all aforementioned issues, but neither [1] nor other works further elaborate this concept.

In relation to (2), Co-Simulation is executed by an *orchestrator*, which is a computer program that initializes all the FMUs and coordinates their execution and data exchange following the specific co-simulation *scenario*. Specifically, the co-simulation scenario describes the inter-connections between input and output variables of each FMU while the orchestrator software enforces these input-output relationships at each step of the co-simulation using a numerical solver for nonlinear algebraic equations (e.g., based on Jacobi or Gauss-Seidel algorithm) [9]. MATLAB/SIMULINK [10], which features FMU import as SIMULINK block, is often used as an orchestrator. In this work, co-simulation is implemented using Maestro v2, which is supported by the not-for-profit INTO-CPS association [11] and FMPy [12]. The overall philosophy is to minimize the risk of customer lock-in.

2.1. Physical space

DIGIT-BENCH DT is developed upon an existing 2-DoFs test bench available at Aarhus University, which is depicted in Figure 2. The test bench consists of a steel reaction frame equipped with a 2D parallel manipulator based on two linear actuators with force transducers, which form the Test Loading Unit (TLU). Device Under Test (DUT) consists of a 500 mm cantilever beam clamped at the frame and subjected to bending and axial loading. Both actuators can be used either in displacement- or force-control mode. An INDEL RT system controls both axes and performs data acquisition. The INDE RT system mirrors all hardware resources to a standard Windows PC via a Python API.

Test bench plus DUT form the *physical space*. It is important to remark that the test bench and DUT have been designed to support R&D in the areas of hybrid testing and digital twins. For this reason, the DUT is a simple and cheap metal plate that can be easily replaced.

[1]Customer lock-in refers to the situation where a customer using a product or service cannot transit to a competitor's product or service without substantial costs, technical incompatibilities, or other significant barriers.

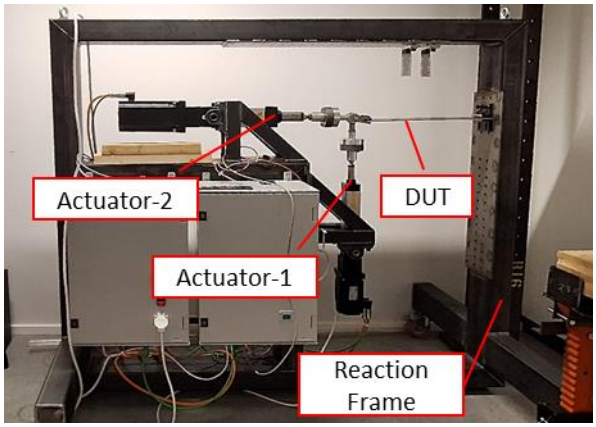


Figure 2 2-DoFs test bench installed at Aarhus University, Department of Civil and Architectural Engineering

2.2. Digital space

The models that form the "Digital Space" are three simulation models and one CAD model, which are packed as FMUs.

- The finite element model of the DUT is implemented for structural analysis of the DUT. The FMU of this model is called *FMU-DUT*.
- The 2DoFs Multi-Body Dynamics (MBD) model is implemented to simulate the dynamic behavior of the test bench manipulator, i.e., TLU. This model is under the FMU and is called *FMU-TLU*.
- The finite element model of the reaction FRAME of the test bench is implemented for structural analysis. This model is wrapped as *FMU-FRAME*.
- The 3D model of the test bench is modeled in the gaming engine Unity to visualize the setup in 3D. This model is wrapped as *FMU-UNITY*.

The digital space models are kept as simple as possible to understand how the different software components communicate with each other. Therefore, simulation models are characterized by static behavior. This is necessary to speed up the DT's software architecture. A detailed description is given for each FMU in the following.

(a) FMU-DUT

The structural analysis model of the FMU-DUT is based on the Finite-Element Method (FEM) and relies on linear elastic Bernoulli 2D beam elements. As shown in Figure 3, the tip displacements $\{d_x^C, d_y^C\}$ of point C are controlled (FMU input) while corresponding reactions at points C $\{f_x^C, f_y^C\}$ and D $\{f_x^D, f_y^D, m_z^D\}$ (FMU output) are computed by solving the static equilibrium of the system. The model of the FMU-DUT is reported in Figure 3. For the sake of clarity, the input variables of the FMU are colored green, while the output variables of the FMU are colored in red.

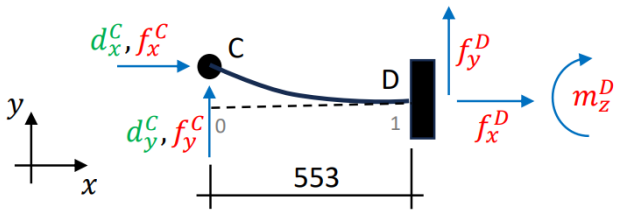


Figure 3 Numerical model of the tested cantilever beam (FMU-DUT). Dimensions are in mm. FMU input variables are in green, while FMU output variables are in red.

Accordingly,

$$\begin{cases} \mathbf{K}\mathbf{d} = \mathbf{Z}_\lambda^T \boldsymbol{\lambda} + \mathbf{Z}_f^T \mathbf{f} \\ \mathbf{Z}_\lambda \mathbf{d} = \bar{\mathbf{d}} \end{cases} \tag{1}$$

where $\mathbf{K}$ is the stiffness matrix of the model, $\mathbf{d}$ is the vector of nodal displacements and $\mathbf{Z}_f$ is a Boolean matrix that collocates the applied loads $\mathbf{f}$ to the correct entries of the static equilibrium equation. The Lagrange multipliers $\boldsymbol{\lambda}$ are introduced to enforce a prescribed nodal displacement $\bar{\mathbf{d}}$ to a subset of DoFs. The latter are extracted by means of a Boolean matrix $\mathbf{Z}_\lambda$, which is also used to collocate the Lagrange multipliers $\boldsymbol{\lambda}$ to the correct entries of the static equilibrium equation.

(b) FMU-TLU

The TLU model includes the two actuators that form a 2-DoFs model, which is the displacement of point C. The actuators are pinned at points A and B, which results in reaction forces at those points. The nodal coordinates of points A, B, and C are given in Table 1. As shown in the Figure 4, the displacement at the control point C $\{d_x^C, d_y^C\}$ is the input of the FMU, which is the test sequence, while the resultant reactions at point A $\{f_x^A, f_y^A\}$ and point B $\{f_x^B, f_y^B\}$ are computed by solving the inverse kinematics of the multibody simulation of the system.

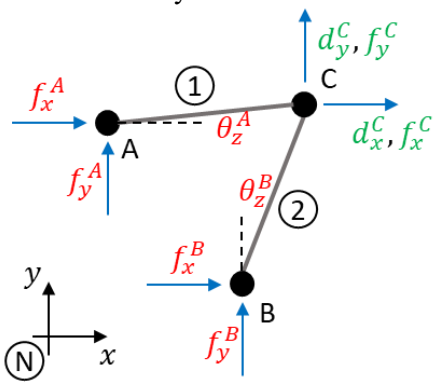


Figure 4 Numerical model of the actuation system (FMU-TLU). FMU input variables are in green while FMU output variables are in red.

Table 1 Nodal coordinates of the actuator joints ($q_1 = 0$, $q_2 = 0$)

Point	x coord. [mm]	y coord. [mm]
A	917	1262
B	1183	598.5
C	1183	1262

The loop closure residual equations defining the relationship between $\{d_x^C, d_y^C\}$ and the joint coordinate$\{q_1, q_2\}$, which coincides with the elongation of the TLU actuators, reads, (following equation 4.31 of [13]):

$$\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix} = \begin{bmatrix} q_1 + l_1(0,0) - l_1\left(d_x^C, d_y^C\right) \\ q_2 + l_2(0,0) - l_2\left(d_x^C, d_y^C\right) \end{bmatrix} \tag{2}$$

The expressions of the actuator lengths read,

$$l_1\left(d_x^C, d_y^C\right) = \sqrt{\left(x^C + d_x^C - x^A\right)^2 + \left(y^C + d_y^C - y^A\right)^2} \tag{3}$$

$$l_2\left(d_x^C, d_y^C\right) = \sqrt{\left(x^C + d_x^C - x^B\right)^2 + \left(y^C + d_y^C - y^B\right)^2} \tag{4}$$

The expression of the forces measured at the actuator rods as a function of the forces applied to the DUT are obtained following [13]

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \left(-\begin{bmatrix} \frac{\partial \varepsilon_1}{\partial q_1} & \frac{\partial \varepsilon_1}{\partial q_2} \\ \frac{\partial \varepsilon_2}{\partial q_1} & \frac{\partial \varepsilon_2}{\partial q_2} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial \varepsilon_1}{\partial d_x^C} & \frac{\partial \varepsilon_1}{\partial d_y^C} \\ \frac{\partial \varepsilon_2}{\partial d_x^C} & \frac{\partial \varepsilon_2}{\partial d_x^C} \end{bmatrix} \right)^T \begin{bmatrix} f_x^C \\ f_y^C \end{bmatrix} \tag{5}$$

The corresponding reaction forces transferred to the reaction frame through the actuator pins are obtained as,

$$\begin{bmatrix} f_x^A \\ f_y^A \end{bmatrix} = \tau_1 \mathbf{n}_1, \begin{bmatrix} f_x^B \\ f_y^B \end{bmatrix} = \tau_2 \mathbf{n}_2 \tag{6}$$

where the directions of the actuators are obtained as a function of the task displacement,

$$\mathbf{n}_1 = \frac{\begin{bmatrix} x^C + d_x^C - x^A \\ y^C + d_y^C - y^A \end{bmatrix}}{l_1(d_x^C, d_y^C)}, \mathbf{n}_2 = \frac{\begin{bmatrix} x^C + d_x^C - x^B \\ y^C + d_y^C - y^B \end{bmatrix}}{l_2(d_x^C, d_y^C)} \tag{7}$$

The multibody simulation model of the TLU is implemented using SymPy, which is a Python module for symbolic calculations [14].

(c) FMU-FRAME

The structural analysis model of the FMU-FRAME is based on FEM and relies on linear elastic Bernoulli 2D beam elements. The same modelling approach is followed as in DUT model. The model geometry is shown in Figure 5.

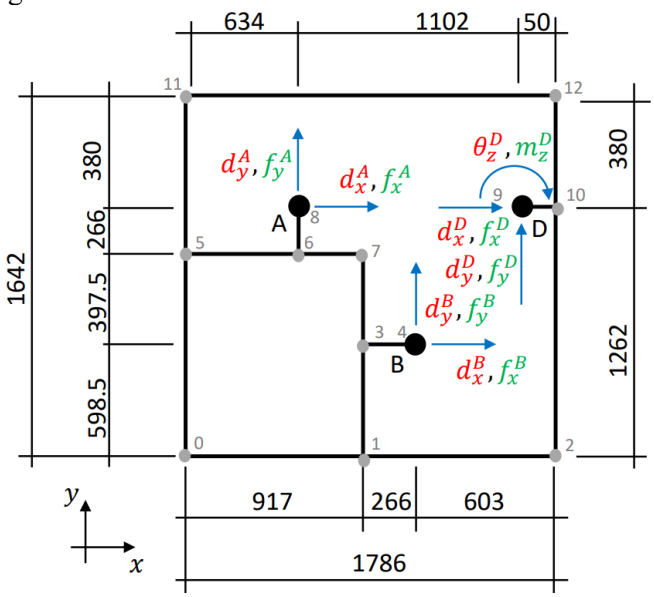


Figure 5 Numerical model of the reaction frame (FMU-FRAME). Dimensions are in mm. FMU input variables are in green while FMU output variables are in red.

The FMU-FRAME is coupled with both FMU-DUT and FMU-TLU, where their corresponding reactions at point A $\{f_x^A, f_y^A\}$, B $\{f_x^B, f_y^B\}$ and D $\{f_x^D, f_y^D, m_z^D\}$ are the inputs (FMU Input) of the FMU-FRAME. The corresponding displacements in point A $\{d_x^A, d_y^A\}$, B $\{d_x^B, d_y^B\}$ and D $\{d_x^D, d_y^D, \theta_z^D\}$ are the output of the FMU (FMU Output).

(d) FMU-UNITY

For the 3D visualization of the virtual testing application, 3D model is exported from the CAD model of the test bench as *.obj file. Unity [18], which is a professional game engine, is used to create and execute the 3D visualization functionality in the DIGIT-BENCH DT. Unity uses C# as scripting API, which is supported by UniFMU [8].

The ProBuilder package [16] in Unity is used to modify and manipulate geometrical objects. One can *ProBuilderize* an object to create modifiable objects from meshes. Once that is done, the meshed body is characterized by a surface associated with vertices. One can warp a geometrical object by controlling the displacement of vertices. As can be seen in Figure 6, the vertices of the geometrical object associated

with the DUT (the cantilever beam) are displaced proportionally to the tip displacement computed by the FMU-TLU using a polynomial interpolating function. Each actuator is composed of the main body and rod. The rod extension is parametrized as well as the rotation of the actuator. Extension and rotation are obtained from the FMU-TLU.

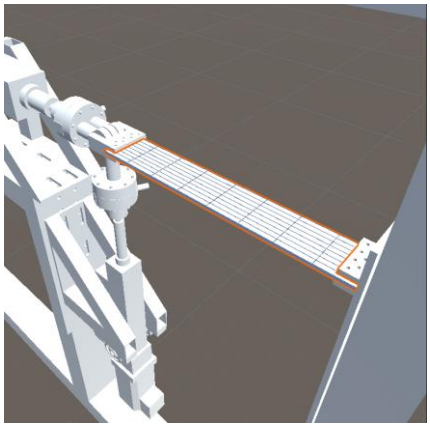
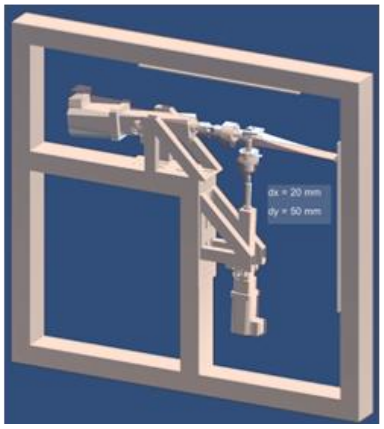
(a)  (b) 

Figure 6 (a) Close-up view of the DUT that is meshed to have flexible deformation. (b) Initial and deformed shape of the 3D model.

2.3. Database

To store and visualize the time history data produced by both the simulation and the experimental setup, InfluxDB [17] database is used. InfluxDB is an open-source time series database which is developed to store and retrieve time series data. **Figure** *7* outlines the architecture of the database. As one can see, the database consists of a *bucket* named *AUCAE2DOFTestBench* which, as the name suggests, we associate to a specific DT. The bucket contains multiple *measurements*. We have found it practical to define two *measurements* namely *digital space* and *physical space*, to group the data based on the DT component that produced it. A *measurement* is a collection of *points*, which comes from different signals. Each point has a time stamp, multiple *tags* and *fields*, which are specified as key-value pairs. Tags are metadata or labels, which facilitate sub-selections of points in the database query. FMU names are used as tag keys when the data is coming from the digital space, and sensor names are used as tag keys when the data is coming from the physical space. Tags are defined to be either *displacement* or *force* to further identify the metadata. Fields are actual measurements (e.g., displacement and force histories). To access the database, the user needs an InlfuxDB *API token*. DIGIT-BENCH queries and writes data from/to InfluxDB from Python using the *influxdb_client* module.

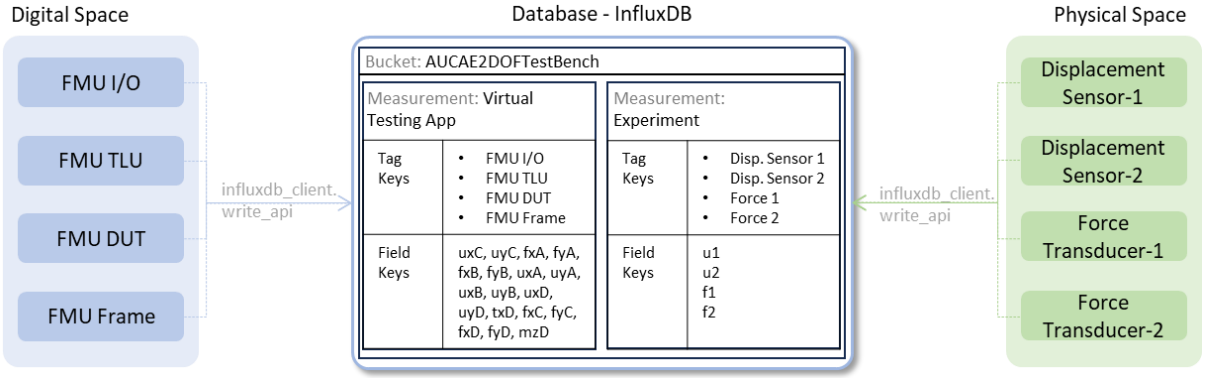


Figure 7 Architecture of the database.

In the current implementation, data streaming from the co-simulation to InfluxDB is implemented through a dedicated FMU. Meastro v2 also allows data to be streamed directly from the co-simulation to InfluxDB via the web interface. This option will be adopted in the future.

2.4. Application example: virtual testing

Virtual testing is a methodology for simulating an experiment prior to its execution. In the context of this work, virtual testing shall be intended as the simulation of a test sequence that consider the coupling of DUT and test bench using a DT. Having a simulation of a desired test sequence of the system helps inform both the test bench operator and the OEM (DUT provider) whether they would be able to execute the experiment or not. Moreover, virtual testing can reduce the likelihood of performing non-informative experiments.

The block diagram of the co-simulation used for the virtual testing application is shown in Figure 8. The input variables of the FMUs are defined as $\mathbf{x}^{(\cdot)}$, where the output variables are $\mathbf{y}^{(\cdot)}$, state variables are $\mathbf{w}^{(\cdot)}$ and $\mathbf{p}^{(\cdot)}$ is where all constant parameters are defined. The orchestrator's sampling time is defined to ensure the coupled simulation's stability and accuracy. In this application, all FMUs are characterized by static behavior and have one-way coupling. Enforcing displacement compatibility and force balance between FMUs simultaneously (tight coupling) requires the orchestrator to accommodate iterations (our future work will tackle this issue). In this case, the orchestrator's sampling rate can be freely adjusted without jeopardizing the simulation's stability and accuracy. When co-simulation combines dynamic FMUs, a time integration process is associated with each dynamic FMU. Therefore, in this case, the orchestrator's sampling time is defined by the stability and accuracy of each single FMU's dynamic response. If FMUs are two-way coupled (*tight coupling*), the interaction between the FMUs has to be accounted for in a stability and accuracy analysis. And therefore, the time step, Δt, is 10 msec in this application.

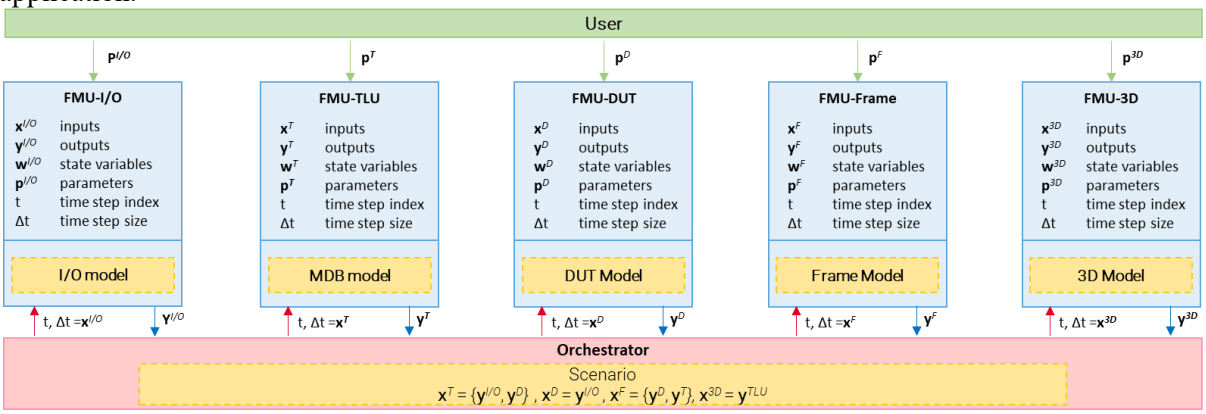


Figure 8 Block diagram of the co-simulation.

In detail, the co-simulation scenario is set up as follows; FMU-I/O includes the input signal of the simulation of the virtual testing application, which are the displacements signal that controls the FMU-TLU. The outputs of the FMU-TLU interact with FMU-DUT through the tip displacement and FMU-FRAME through the reaction forces at the connection points. Moreover, the outputs of the FMU-DUT feed FMU-FRAME, with its reaction forces. It is important to stress that in the present implementation, the DUT applies the load to the FRAME through the TLU, but the frame deformation does not affect TLU and DUT boundary conditions. The implementation of such two-way coupling, which, in the co-simulation community, is indicated as *tight coupling*, is a feature that is still seldom present in the available orchestrator.

It is important to mention that, despite neglecting the system's dynamics the outlined coupling scenario between FMUs is still valid for dynamic cases. Specifically, if dynamics are included, the static equilibrium of the DUT and the reaction frame shall be replaced by the equation of motion. Similarly, the kinematic model of the TLU can be expanded to consider the equation of motion. However, the interface quantities, namely displacement and forces, would remain the same.

The cockpit of the DIGIT-BENCH DT is framed in the InfluxDB time-series database, where the *dashboard* feature of the InfluxDB is used. The overview of the cockpit is presented in Figure 9. It is configured to enable visualizing the real-time response of both the simulation and the test bench.

Figure 9 DIGIT-BENCH DT Cockpit formed with InfluxDB dashboard.

Figure 10 shows a summary of both virtual testing and experimental campaign. Figure 10(a) shows the displacement time histories both in x and y direction, which is an input to the tip of the DUT. Figure 10(b) shows a snapshot from the 3D visualization of the deformed configuration at the point highlighted in Figure 10(a). Lastly, a zoomed-in photo of the physical setup is also presented in Figure 10(c), which was shot while the actual experimental was running. Figures 9 and 10 emphasize that plenty of tools exist to deploy a time series database, cockpit (Figure 9), and 3D visualization environment (Figure 10) with minimal effort.

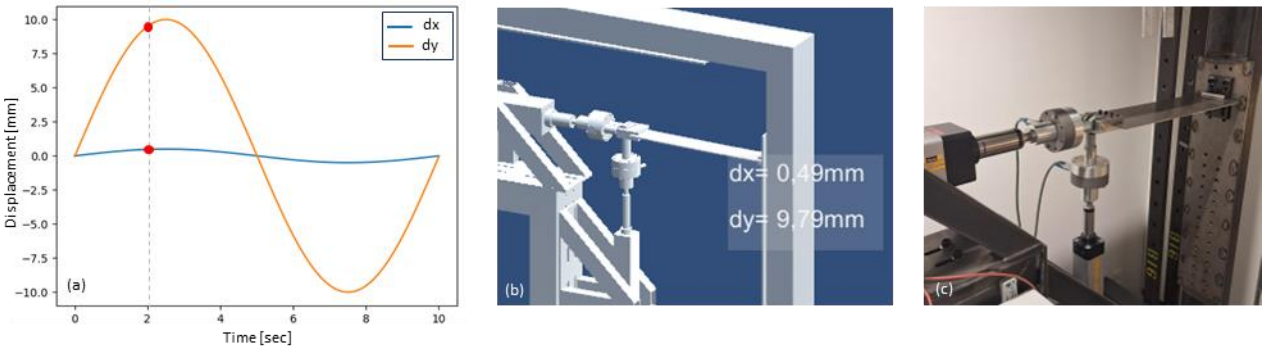


Figure 10 (a) Displacement input for dx and dy, and zoomed-in deformed shape (b) in the 3D visualization in digital space, and (c) physical space.

3. Conclusions

DIGIT-BENCH DT's development status has been presented using a 2-DoFs test bench as a demonstrator. FMI-based Co-Simulation is used to deploy the DT as a collection of FMUs. A virtual testing application combined with 3D visualization is demonstrated using an existing 2-DoFs test bench. Virtual testing is used to simulate an experiment before its execution by combining the simulation models of DUT and the test bench.

The main takeaway from the development of DIGIT-BENCH DT so far is summarized as follows:

- Despite being supported by 170+ simulation tools, the coverage of FMI-based co-simulation is still limited, especially if one wishes to adopt an open-source technology stack. As an example, the authors could not find an open-source FEA tool that supports exporting FMUs suitable for the example.

- Software that features FMU export might not allow for any possible I/O configuration (e.g., input: displacement, output: force input: force, output: displacement). In fact, different I/O configurations might require radically different equation solvers within the FMU. Thus, our recommendation to the potential DT developer is to carefully account for these features when choosing the pool of the simulation tools.
- The sampling rate of the data exchange produced by the DT is very specific to the application. A DT can potentially produce significant amounts of data, which might not be convenient to store entirely. Therefore, crafting an effective and efficient data management strategy is as important as crafting the digital twin models.

Acknowledgements

The funding of the DIGIT-BENCH project by the Energy Technology Development and Demonstration Program (EUDP) (No. 640222-497272) is kindly acknowledged.

References

[1] Schkoda, Bulgakov, Addepalli, and Haque (2013). "System Level Dynamic Modeling Framework Being Developed at Clemson Unviersity's Wind Turbine Drivetrain Testing Facility." In *Proceedings of the ASME 2013 Dynamic Systems and Control Conferences DSCC2013*. Palo Alto, California, USA: ASME.
[2] Siddiqui, M. O., A. R. Nejad, and E. Pedersen (2024). "Virtual Model Development of the Load Application System of a Wind Turbine Nacelle Test Bench for Hybrid Test Applications." *Journal of Dynamic Systems, Measurement, and Control* 146(2): 021002.
[3] Fuller A, Fan Z, Day C, and Barlow C (2020) *Digital Twin: Enabling Technologies, Challenges, and Open Research* IEEE Access 8: 108952–71.
[4] Jassmann, U. (2018). "Hardware-in-the-Loop Wind Turbine System Test Benches and Their Usage for Controller Validation", PhD Thesis
[5] Wagg, DJ, Worden K, Barthorpe R J, and Gardner P (2020) *Digital Twins: State-of-the-Art and Future Directions for Modeling and Simulation in Engineering Dynamics Applications*. ASCE-ASME J Risk and Uncert in Engrg Sys Part B Mech Engrg 6(3): 030901.
[6] (January 2024) Functional mock-up interface standard [online]. Available: https://fmi-standard.org/
[7] (January 2024) Modelica association [online]. Available: https://modelica.org/
[8] Legaard, C. M., D. Tola, T. Schranz, H. D. Macedo, and P. G. Larsen. "A Universal Mechanism for Implementing Functional Mock-up Units," to appear. SIMULTECH 2021. Virtual Event, 2021
[9] Gomes, C., C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe (2018). "Co-simulation: A Survey." *ACM Computing Surveys* 51(3):1–33.
[10] (January 2024) Matlab/Simulink [online]. Available: https://www.mathworks.com/products/simulink.html
[11] Thule, Casper, Kenneth Lausdahl, Cláudio Gomes, Gerd Meisl, and Peter Gorm Larsen (2019). "Maestro: The INTO-CPS Co-simulation Framework." *Simulation Modelling Practice and Theory* 92: 45–61.
[12] (January 2024) FMPy [online]. Available: https://fmpy.readthedocs.io/en/latest/
[13] Taghirad, Hamid. *Parallel Robots Mechanics and Control*, 2013.
[14] (January 2024) SymPy [online]. Available: https://www.sympy.org/en/index.html
[15] (January 2024) Unity [online]. Available: https://unity.com/
[16] (January 2024) ProBuilder [online]. Available: https://docs.unity3d.com/Packages/com.unity.probuilder@6.0/manual/index.html
[17] (January 2024) InfluxDB [online]. Available: https://www.influxdata.com/