DELIVERABLE D7.1

QBlade Interfacing Framework

Robert Behrens de Luna (Technische Universität Berlin)

Funded by the European Union

Document track information

Project information	
Project Title	Developing the Next Generation of Environmentally-Friendly Floating Wind Farms with Innovative Technologies and Sustainable Solutions
Starting date	01.01.2024
Duration	48 months
Programme	HORIZON EUROPE
Call Identifier	HORIZON-CL5-2023-D3-01
Grant Agreement No	101136091

Deliverable information	
Deliverable number	7.1
Work package number	7
Deliverable Title	QBlade Interfacing Framework
Lead beneficiary	Technische Universität Berlin
Author	Robert Behrens de Luna
Due date	31.12.2024
Actual submission date	17.12.2024
Type of deliverable	Other
Dissemination Level	Public

Funded by the European Union

Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor EISMEA can be held responsible for them.

Revision table

Version	Contributors	Date	Description
V0.1	Behrens de Luna, TUB	11/12/2024	First draft
V1.0	Claudio Lugni, CNR Behrens de Luna, TUB	17/12/2024 17/12/2024	Revision Final edits

List of acronyms

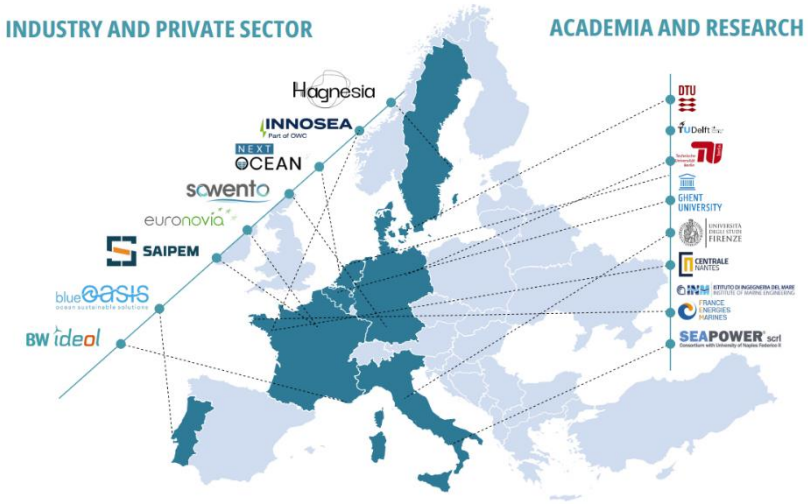
Acronym	Full name
AEP	Annual Energy Production
BEM	Blade Element Momentum
DEL	Damage Equivalent Load
DLC	Design Load Case
DOF	Degree Of Freedom
FPM	Fully Populated Matrix
FOWT	Floating Offshore Wind Turbine
HAMS	Hydrodynamic Analysis of Marine Structures
LLFVW	Lifting-Line Free Vortex Wake
NREL	National Renewable Energy Laboratory
Pc	Pitch control
WEIS	Wind Energy and Integrated Servo-Controls
WISDEM	Wind-plant Integrated System Design and Engineering Model
WP	Work Package
WSL2	Windows Subsystem for Linux 2
ROSCO	Reference OpenSource Controller

ABOUT FLOATFARM

The FLOATFARM project is a Research and Innovation Action funded by the European Union under the Horizon Europe program. This project is closely linked to the FLOATECH project (2020-2023) and aims to bring the technologies developed within FLOATECH to the next level of technological readiness, complementing them with a significant number of new concepts, innovations and methods.

FLOATFARM aims to significantly advance the maturity and competitiveness of floating offshore wind (FOW) technology by increasing energy production, achieving significant cost reductions within the design and implementation phases, improving offshore wind value chain and supporting EU companies in this growing sector. Additionally, FLOATFARM aims to decrease negative environmental impacts on marine life and to enhance the public acceptability of FOW, thereby accelerating the EU energy transition.

The FLOATFARM consortium is coordinated by the Technische Universität Berlin and is made up of 17 partners spread across 8 countries, each contributing to technology and scientific excellence in the wind energy sector.



The approach of FLOATFARM can be broken down into three actions:

1) **Turbine Technology**: Development of innovative technologies and methods for improvements on an individual FOW turbine level,

2) **Farm Technology**: Development, investigation and demonstration of technologies that are applicable to an array of turbines within a FOW farm,

3) **Environmental & Socioeconomic Impacts**: Model development, data collection and scenario analysis of environmental, economic and sociological impacts of FOW farms.

Table of contents

1. Introduction

1.1. Context within FLOATFARM

This document presents the interface between the aero-hydro-servo-elastic simulation tool QBlade and the holistic optimization framework called WEIS (Wind Energy integrated Servo-Controls), henceforth called QBtoWEIS, that was developed in Task 7.1 of WP7 of FLOATFARM. This integration provides the basis for several optimization and design tasks within Work Packages (WPs) 1-6 and is therefore essential for successful achievement of FLOATFARM objectives.

This document should be read as a guide to the public release of QBtoWEIS, which is the actual deliverable.

1.2. Document Structure

The document is divided into three sections. First, the motivation for QBtoWEIS and its capabilities are discussed. Second, a guide to installing and running the model is provided. Lastly, results of two test case optimization problems are demonstrated.

2. QBtoWEIS

2.1. Overview and Motivation

Floating Offshore Wind Turbines (FOWTs) are inherently interdisciplinary systems that are excited by a multitude of non-linear external inputs. Figure 1 provides and overview of the main sub-systems that comprise a FOWT and the environment in which they operate. Turbines currently analyzed within the science community possess rotor diameters exceeding 250m and have a rated power of 20 or more megawatt. These large structures experience vastly different aerodynamic conditions depending on the azimuthal position of the blade. At the same time, floating structures are exposed to linear and non-linear wave excitation that potentially cause resonance. The aerodynamic and wave forces cause the floating substructure to move within its rigid degrees of freedom (DOFs), most prominently in surging and pitching motions, causing the turbine to be exposed to negative aerodynamic damping and, at times, interact with its own wake. Due to the rotational speed of the rotor, gyroscopic effects play an increasing role when the turbine undergoes a pitching motion. The continuous motion of the turbine adds to the load requirements that are posed to a FOWT, compared to their fixed-bottom or onshore counterparts (e.g. on the intersection between floating sub-structure and the tower). To keep the system stable, to

maximize annual energy production (AEP), to react to grid events or to secure the system during extreme weather events, a controller manipulates rotational speed and blade pitch during operation.

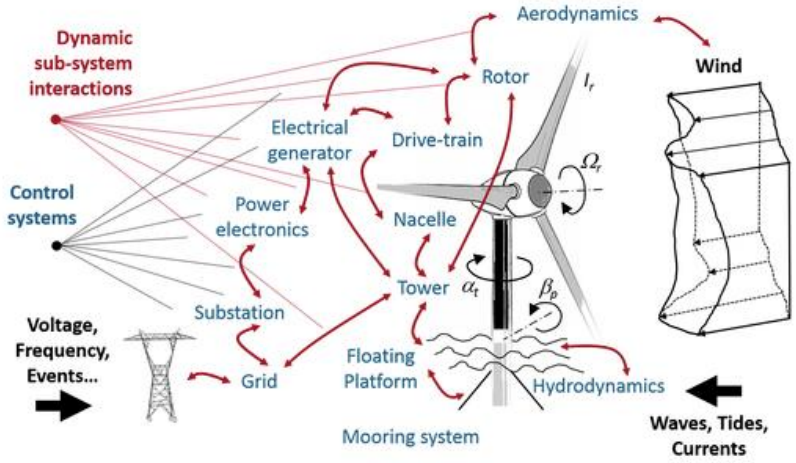


Figure 1 – Sub-systems of a FOWT and environmental excitation. Figure taken from [1].

This complex interaction of sub-systems poses a challenge to designers of each sub-component as any design decision taken might have an unintended consequence on overall system behavior in such dynamic and tightly coupled system.

To cope with this challenge, the design approach of FOWT might have to evolve from a sequential one to a Control Co-Design (CCD) approach. In a sequential approach the mechanical (sub-)system is typically designed initially and the controller design begins once the mechanical system is considered to be in a finished iteration. Within the design of the mechanical sub-systems, each group has to design their component so that pre-defined constraints and objectives are met. By nature, these constraints and objectives have to be conservative to prevent a highly iterative process that can cost time [1].

CCD offers the potential to design the mechanical floating offshore wind energy system at the same time as the controller and thus finding solutions that could not found in a sequential approach [3].

2.2. WEIS

The Wind Energy and Integrated Servo-Controls toolset is a framework developed at the National Renewable Energy Lab (NREL) to design and optimize floating offshore wind turbine structures.

WEIS consists of many mostly NREL developed open-source tools that are connected in WEIS using the NASA developed open-source python library OpenMDAO [4]. Figure 2

shows the tools that make up WEIS and demonstrates the direction of the workflow in a typical optimization or design task.

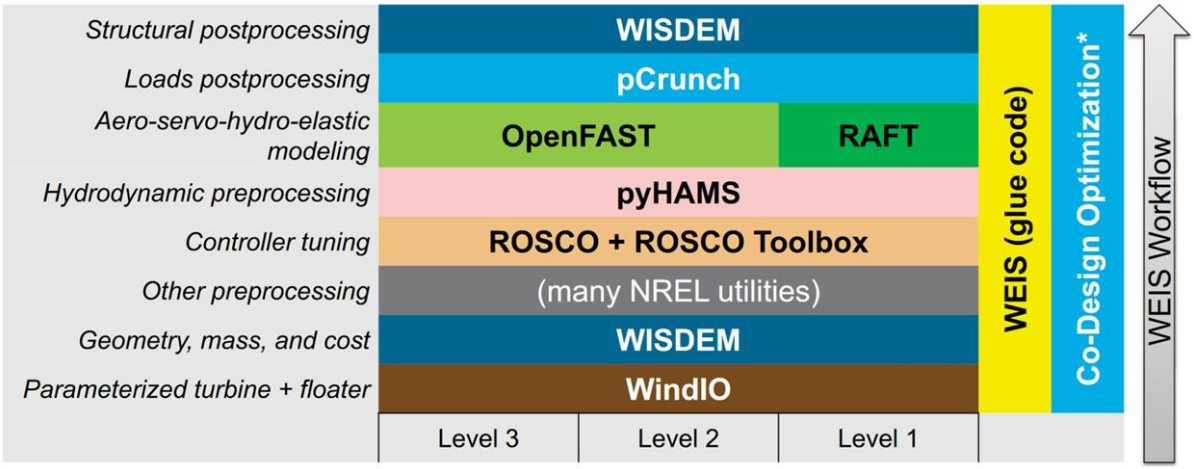


Figure 2 – Components that are integrated in WEIS, taken from [5].

In the present document, the toolchain is briefly described. For a detailed description of the individual tools and the work flow, the reader is referred to publication [5].

WindIO: Parametric definition of mostly inputs and outputs of a FOWT design [6].

WISDEM: Wind-plant Integrated System Design and Engineering Model (WISDEM) is an optimization framework in its own right. WISDEM relies on steady state tools and includes techno-economic modules to derive masses, costs and structural beam properties. Within WEIS, WISDEM is utilized mostly to evaluate costs and convert the WindIO definition into a OpenFAST/RAFT/QBlade model.

ROSCO: Reference OpenSource Controller (ROSCO) is an open-source reference controller that comes with the ROSCO toolbox. The toolbox allows for a generic process to enable CCD.

pyHAMS: python wrapper for Hydrodynamic Analysis of Marine Structures (HAMS) [8], which is an open-source software to analyze wave-structure interaction in the frequency domain. Added mass, damping coefficients and excitation forces are evaluated for 3D structures in order to include radiation and diffraction effects in time domain simulations.

OpenFAST: Nonlinear aero-servo-hydro-elastic simulation framework for wind turbine response in time domain. The typical simulation duration is in the magnitude of minutes.

RAFT: Response Amplitudes of Floating Turbines (RAFT) is a linear, frequency domain representation of a FOWT [10]. The typical simulation duration is in the magnitude of seconds.

pCrunch: IO and post processing interface for OpenFAST results [11]. Used within WEIS to analyze load case results and derive parameters usually used as constraints or merit figures (such as damage equivalent loads (DELs), AEP, etc.)

2.3. QBlade in the WEIS framework

The objective of the coupling that is described in this document is to expand the stack of tools that WEIS combines by the simulation framework QBlade and thus provide a design engineer with another option next to OpenFAST and RAFT to carry out aero-servo-hydro-elastic simulations.

QBlade [12] is an aero-servo-hydro-elastic simulation framework with similar capabilities as OpenFAST. During the Horizon2020 project FLOATECH (https://www.floatech-project.com/), QBlade was expanded by a hydrodynamic module called QBlade-Ocean [13] and subsequently validated in [14]. Accordingly, wind turbine response in time domain can be analyzed leveraging the various fidelity levels with regards to wake aerodynamics and structural modeling that QBlade offers.

In particular, QBlade includes a Lifting-Line Free Vortex Wake method (LLFVW) which increases the fidelity level of wake aerodynamics compared to the conventional Blade Element Momentum (BEM) method. The LLFVW method is highly optimized for work station use and thus may enable optimization of selected load cases with this higher fidelity method. Previous studies ([15,16]) have assessed that the LLFVW method estimates lower DELs compared to the BEM method. Including this method may therefore lead to more slender designs, ultimately reducing the material costs.

Furthermore, QBlade's coupling to the multi-physics engine Chrono [17] adds a fully non-linear structural model to WEIS with the option to use Euler-Bernoulli, Timoshenko or Timoshenko-FPM (fully populated matrix) beam elements. Especially the latter beam option represents a crucial increase of fidelity to capture coupled dynamics in the structural model (such as blade-twist coupling). The choice of the beam model has hardly an influence on computational speed making it feasible to include the FPM model in optimization problems.

In order to enable a designer to access QBlade from within WEIS, an OpenMDAO component has been created for QBlade and all required inputs from WISDEM, pyHAMS, ROSCO, etc. have been mapped to QBlade specific inputs. This enables a user to derive an accurate QBlade model from the windIO definition. Subsequently, QBlade simulations for user defined design load cases (DLCs) are generated and called from a QBlade wrapper within WEIS. Once the simulation terminates, relevant time series are written out and mapped to channels that can be interpreted by the post-processing tool pCrunch. pCrunch

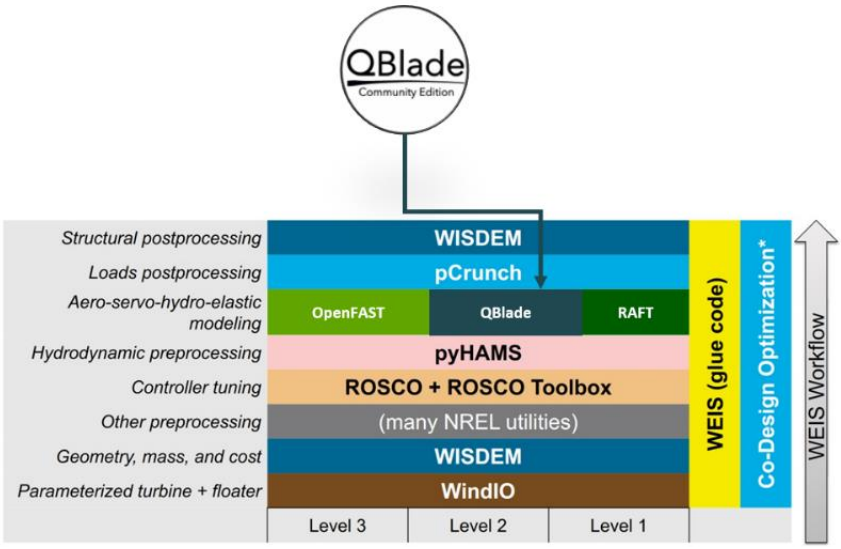


Figure 3 – QBlade in the WEIS Framework, taken from [5] and modified.

derives parameters such as AEP, DELs, max platform pitch angle, etc. These are utilized by the optimizer as constraints, design variables or merit figures.

3. Access, Installation and Basic User Guide

3.1. Access

QBtoWEIS is released as a fork to the WEIS repository maintained by NREL. This was chosen as this allows to maintain the repository efficiently when other developers submit updates to the main WEIS repository. Furthermore, it provides high visibility to the repository.

The fork can be accessed at this public link:

https://github.com/rbehrensdeluna/QBtoWEIS.git

3.2. Installation

QBtoWEIS has been developed and tested on Windows Subsystem for Linux (WSL2). While cross-platform compatibility has been considered throughout development, installation on a Windows platform is not recommended at this point. This is one area of focus to develop the code further after the initial release in December 2024.

The installation process is in line with the one of the core WEIS code. The following commands have been tested on various machines starting with a baseline WSL distribution and Ubuntu-22.04. Installation with Anaconda 64-bit (https://www.anaconda.com/) in a newly created self-contained environment is recommended.

1. Clone the repository and create a virtual environment install the software:

```
conda config --add channels conda-forge
git clone https://github.com/rbehrensdeluna/QBtoWEIS.git
cd QBtoWEIS
```

2. Create a virtual environment and install the software:

```
conda env create --name qbweis-env -f environment.yml
conda activate qbweis-env
conda install -y petsc4py mpi4py
pip install --no-deps -e . -v
conda install -c conda-forge pyoptsparse
```

3. Download and configure QBladeCE executables under WSL2:

Download QBladeCE from https://qblade.org/downloads/ and place it in the WSL2 directory. Then install some libraries that are required to run QBlade:

```
sudo apt-get update -y
sudo apt-get install -y libqt5opengl5 libqt5xml5 libquadmath0 libglu1-
mesa
```

Then, make QBladeCE executable:

```
chmod +x run_qblade.sh
chmod +x QBladeCE_x.y.z # x.y.z should be replaced by the actual version
number, e.g. 2.0.7.8
```

NOTE: QBtoWEIS requires QBladeCE/QBladeEE version 2.0.7.8 or newer to enable for multi-processing capabilities (number_of_workers > 1)

Congratulations, you now have installed QBtoWEIS on your local machine!

To test if everything is working, navigate to the "qb_examples" folder and run the example

```
cd qb_examples
python weis_driver_oc3.py
```

This test case simply runs the NREL5MW wind turbine on the oc3 floating sub structure at rated conditions without starting an optimization.

Note: To make sure that the QBlade wrapper can access the shared object and set the environment to run QBlade, please enter the paths to where QBlade is located on your system in the *modeling_options.yaml* file of the provided example:

```
General:
    qblade_configuration:
        flag: True
        path2qb_libs: <abs_path_to_QBlade_Libraries>
        # should end like: <…>/QBladeCE_2.0.7.8/Libraries
        path2qb_dll: <abs_path_to_QBlade_shared_object> should end like
        # should end like: <…>/QBladeCE_2.0.7.8/libQBladeCE_2.0.7.8.so.1.0.0
```

3.3. User Guide

When a user plans to use QBlade in an optimization problem or design task within WEIS, the workflow follows the same logic as if a level 1 (RAFT) or level 2 or 3 (OpenFAST) optimization with WEIS was to be run.

In particular, WEIS is accessed by a user through three **.yaml* type files and one driver python script that calls each of the *yaml* files and runs WEIS. In the following a short overview is provided

windIO.yaml*:*
This file contains the parametric definition of a FOWT including information such as, airfoils, blade geometry, internal blade layup, floater geometry, materials, wall thicknesses, etc.

There are several wind turbine geometries available to the research community:

- NREL5MW-Spar OC3 & -Semi OC4
- IEA 3.4-130 RWT
- IEA-15MW-250-RWT Monopile & VolturnUS-S
- IEA-22MW-280-RWT Monopile & VolturnUS-S

modeling_options.yaml*:*
The modeling options file lets the user define code specific simulation settings. Here, the user may select between the different aero-servo-hydro-elastic modeling tools (OpenFAST, RAFT, QBlade), and enter inputs relevant to, e.g. wake modeling, time step size, if a controller is included, the load case, etc. This file allows for an exhaustive list of entries as almost any QBlade parameter may be set. To keep the mapping intuitive, most entries are set using the QBlade keywords that match the text-based format of QBlade turbine and simulation (*.bld/*.sim/*.trb), see [18] for more information on the text-based QBlade format.

A detailed overview of available inputs, detailed descriptions, entry-types and bounds can be found in the modeling_schema.yaml (*QBtoWEIS/weis/inputs/modeling_schema.yaml*). Here, all QBlade relevant inputs are found in in the "Level4" object and "qblade_configuration" which is nested in the properties object.

analysis_options.yaml*:*
The analysis options file determines in which context simulations are run. Either, chosen load cases are simply run for analysis purposes, an optimization problem is set up or a design of experiments built. In case any of the latter two options is chosen, design variables, constraints and, if required, a merit figure can be chosen. Furthermore, settings concerning the optimization algorithm such as maximum iterations or convergence criteria can be defined.

weis_driver.py:
Not necessarily required, this python script points to the three abovementioned files and invokes the "run_weis()" method and passes the three files as arguments to the function.

4. Demonstration Test Cases

The test cases presented below are intended to demonstrate the functional coupling between QBlade and WEIS and to show that the optimizer finds solutions to problems that are easy to interpret and may seem trivial for validation purposes.

4.1. IEA15MW-Monopile

In this example, a fixed bottom offshore wind turbine is supposed to be optimized with the objective to reduce the Damage Equivalent Loads acting on the tower base connection between monopile and tower.

We simulate steady conditions at wind speeds of {4, 6, 8, 10, 12, 15, 20}[m/s]. The simulations include the ROSCO controller. Thus, the optimization problem is formulated as:

Minimize:
- twr base del

By Varying:
- tower diameter
- tower wall thickness
- monopile diameter
- monopile wall thickness

Subject to:
- stress constraint
- global buckling
- shell buckling
- diam. to thick. ratio
- taper
- frequency
- …

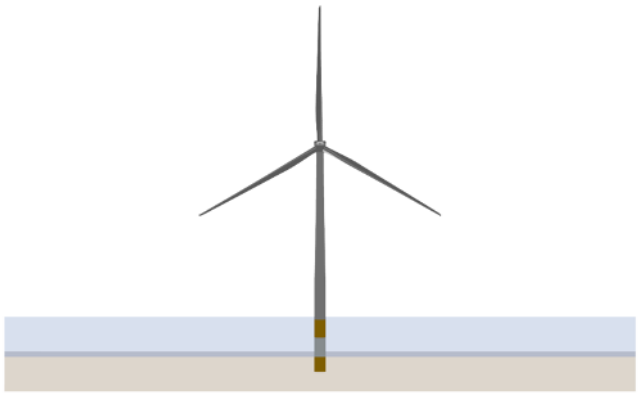


Figure 4 – IEA15MW turbine, rendered in the QBlade GUI

Since there are no constraints on structure mass nor cost, one would expect the optimizer to increase the wall thicknesses and diameters of tower and monopile in order to minimize stress in the system and thus reduce the tower base DEL.

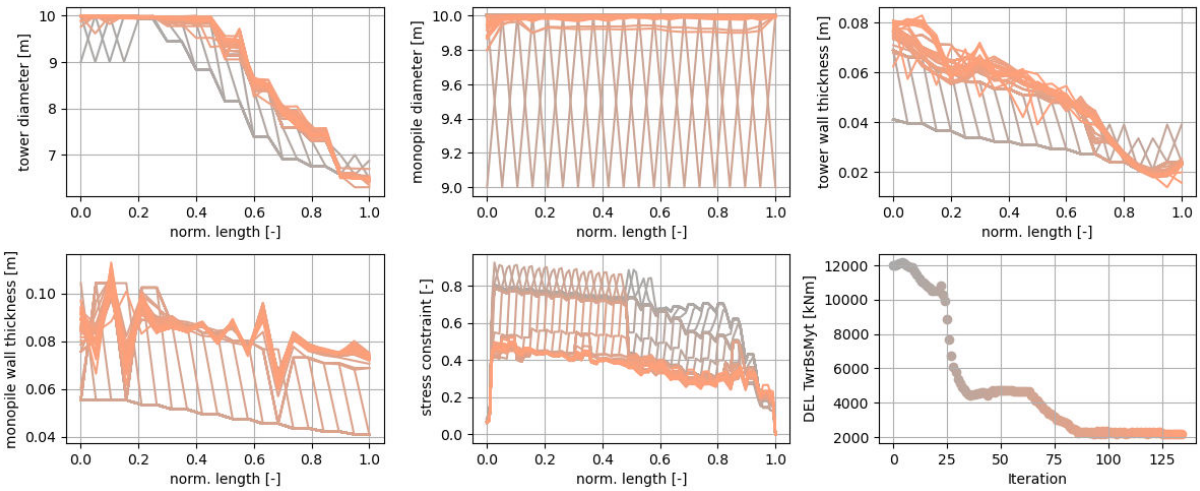


Figure 5 - Optimization progression to reduce the tower base fore-aft DEL of on the IEA15MW wind turbine on a monopile platform. Shown are the control points accessible to the optimizer over the normed length of the tower or monopile as well as the merit figure per iteration.

Figure 5 shows the progression of the optimization. In all sub-figures except for *DEL TwrBsMyt*, the various control points are displayed over the normalized length of the tower or monopile. To visualize the progression of the optimization, a color-code is added, where gray represents the baseline configuration and orange the final configuration.

As was expected, the tower diameter was increased within the bounds (upper bound 10m), while still complying with a taper constraint. The monopile diameter is maintained at 10m which presents an upper bound as well. For both the tower and the monopile, the wall thicknesses are increased considerably, leading to a reduction of the stress constraint[1]. The influence of the chosen design variables on the fore-aft tower base DEL is evident in the iteration tracker of the chosen merit figure *DEL TwrBsMyt*.

4.2. IEA22MW UMaine VolturnUS-S

The second test case focuses on the IEA22MW wind turbine mounted on the semi-submersible VolturnUS-S substructure with catenary mooring lines. This time, the objective is the reduction of the structural mass of the substructure, a common optimization merit figure to reduce material use and save cost.

[1] The stress constraint is defined as the maximum allowable von Mises stress relative to the material yield stress

With 11.5m/s wind speed, rated wind conditions (two turbulent seeds) with still water are considered and the ROSCO controller is utilized. The optimization problem is formulated accordingly:

Minimize:

platform mass

By Varying:

platform draft
column spacing
column diameter
pc natural frequency
pc damping ratio
...

Subject to:

pitch natural frequency
heave natural frequency
max. platform pitch

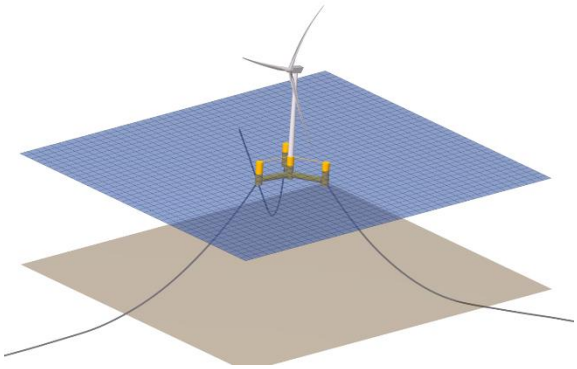


Figure 6– IEA22MM on UMaine VolturnUS-S platform, rendered in the QBlade GUI

In order to reduce platform mass, it seems obvious that the optimizer would attempt to reduce the dimensions of the platform.

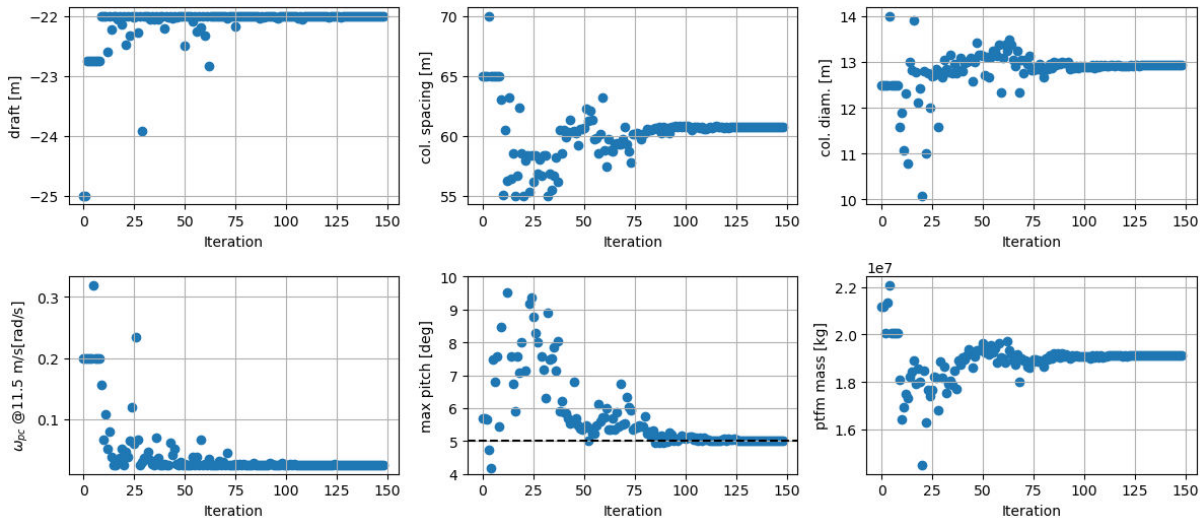


Figure 7 – Optimization progression to reduce the platform mass of the UMaine VolturnUS-S for a IEA22MW wind turbine. Shown are platform draft defined as the distance from sea level, the column spacing as the radial distance from the tower position to the outer columns as well as their column diameter. The second row shows the pitch control natural frequency design variable, the maximum floater pitching constraint and the merit figure, platform mass.

Analyzing the iteration tracker of the optimization in Fig. 7 one can see that platform draft is immediately reduced to the upper bound of -22m while the column spacing and diameter is initially reduced as well. As one would expect, this measure reduces platform mass greatly.

However, platform stability is strongly decreased as the maximum pitch angle during operation reaches values well above the set constraint of 5 degrees. To comply with this constraint, the optimizer gradually increases column spacing and diameter. Finally, the max pitch constraint is reached with a considerable improvement in platform mass with regards to the initial configuration. In parallel, the pitch control natural frequency and damping ratio (not shown here for brevity) are tuned, demonstrating a functional CCD example with the QBtoWEIS coupling.

5. Conclusion

In this deliverable, an overview of the work carried out in Task 7.1 within WP 7 of the FLOATFARM project is given. Thereby, an integration of the aero-servo-hydro-elastic wind turbine simulation tool QBlade into the design and optimization framework WEIS is introduced.

The motivation for such a tool lies in the multi-disciplinary complex problem that is posed to designers and engineers involved in the optimization and design process of FOWTs. WEIS already provides a holistic toolset to carry out multi-disciplinary design analysis and optimization tasks. This toolset is expanded by QBlade, adding various options and capabilities to the framework that include wake modeling and structural dynamics.

Two test cases were presented for the optimizer where the optimization problems were formulated in such a way that a certain result could be expected based on the given design space, constraints and merit figure. QBtoWEIS performed as expected, reducing the DEL on a fixed-bottom monopile offshore wind turbine and the structural mass of the floating substructure on a semi-submersible platform.

6. Literature

[1] Garcia-Sanz, M. (2019). Control Co-Design: An engineering game changer. *Advanced Control for Applications*, *1*(1). https://doi.org/10.1002/adc2.18

[2] IEAWindTask. *GitHub - IEAWindTask37/IEA-22-280-RWT: Repository for the IEA 22-MW offshore reference wind turbine developed by the IEA Wind Task 55 REFWIND*. GitHub. https://github.com/IEAWindTask37/IEA-22-280-RWT

[3] Abbas, N. J., Jasa, J., Zalkind, D. S., Wright, A., & Pao, L. (2023). Control co-design of a floating offshore wind turbine. *Applied Energy*, *353*, 122036. https://doi.org/10.1016/j.apenergy.2023.122036

[4] J. S. Gray, J. T. Hwang, J. R. R. A. Martins, K. T. Moore, and B. A. Naylor, "OpenMDAO: An Open-Source Framework for Multidisciplinary Design, Analysis, and Optimization," Structural and Multidisciplinary Optimization, 2019.

[5] Daniel Zalkind and Pietro Bortolotti 2024 J.Phys.:Conf.Ser. 2767 082020

[6] IEAWindTask. *GitHub - IEAWinDTask37/WiNdIO*. GitHub. https://github.com/IEAWindTask37/windIO

[7] ROSCO. *GitHub - NREL/ROSCO: a reference open source controller for wind turbines*. GitHub. https://github.com/NREL/ROSCO

[8] YingyiLiu. *GitHub - YingyiLiu/HAMS: An open-source computer program for the analysis of wave diffraction and radiation of three-dimensional floating or submerged structures*. GitHub. https://github.com/YingyiLiu/HAMS

[9] OpenFAST. *GitHub - OpenFAST/openfast: Main repository for the NREL-supported OpenFAST whole-turbine and FAST.Farm wind farm simulation codes.* GitHub. https://github.com/OpenFAST/openfast

[10] *RAFT*. *GitHub - WISDEM/RAFT: A frequency-domain dynamics model for floating wind turbines*. GitHub. https://github.com/WISDEM/RAFT

[11] *pCrunch*. *GitHub - NREL/pCrunch*. GitHub. https://github.com/NREL/pCrunch

[12] *QBlade Documentation — QBlade Documentation 2.0.7 documentation*. https://docs.qblade.org/

[13] Saverin, J., Perez-Becker, S., Behrens de Luna Robert, Marten, D., Gilloteaux, J., & Kurnia, R. (2021). D1.2 Higher order Hydroelastic module. In *Zenodo (CERN European Organization for Nuclear Research)*. https://doi.org/10.5281/zenodo.6958081

[14] Perez-Becker, S., Saverin, J., Behrens de Luna, R., Papi, F., Combreau, C., Ducasse, M.-L., Marten, D., and Bianchini, A.: Validation Report of QBlade-Ocean, Tech. rep., https://doi.org/10.5281/zenodo.7817605, 2022.

[15] Behrens de Luna, R., Perez-Becker, S., Saverin, J., Marten, D., Papi, F., Ducasse, M., Bonnefoy, F., Bianchini, A., & Paschereit, C. (2024). Quantifying the impact of modeling fidelity on different substructure concepts for floating offshore wind turbines – Part 1: Validation of the hydrodynamic module QBlade-Ocean. *Wind Energy Science*, *9*(3), 623–649. https://doi.org/10.5194/wes-9-623-2024

[16] Papi, F., Troise, G., Behrens de Luna, R., Saverin, J., Perez-Becker, S., Marten, D., Ducasse, M., & Bianchini, A. (2024). Quantifying the impact of modeling fidelity on different substructure concepts – Part 2: Code-to-code comparison in realistic environmental conditions. *Wind Energy Science*, *9*(4), 981–1004. https://doi.org/10.5194/wes-9-981-2024

[17] Tasora, A., Serban, R., Mazhar, H., Pazouki, A., Melanz, D., Fleischmann, J., Taylor, M., Sugiyama, H., and Negrut, D.: Chrono: An Open Source Multi-physics Dynamics Engine, in: Lecture Notes in Computer Science, pp. 19–49, Springer International Publishing, https://doi.org/10.1007/978-3-319-40361-8_2, 2016.

[18] *Turbine Definition ASCII File — QBlade Documentation 2.0.7 documentation*. https://docs.qblade.org/src/user/turbine/turbineexport.html

FLOATFARM

NEXT GENERATION FLOATING WIND FARMS