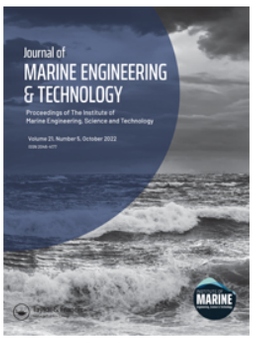

Journal of Marine Engineering & Technology

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/tmar20

A distributed object-oriented simulator framework for marine power plants with weak power grids

Stian Skjong & Eilif Pedersen

To cite this article: Stian Skjong & Eilif Pedersen (2022): A distributed object-oriented simulator framework for marine power plants with weak power grids, Journal of Marine Engineering & Technology, DOI: 10.1080/20464177.2022.2120171

To link to this article: https://doi.org/10.1080/20464177.2022.2120171

Published online: 12 Sep 2022.

JOURNAL OF MARINE ENGINEERING & TECHNOLOGY
https://doi.org/10.1080/20464177.2022.2120171

A distributed object-oriented simulator framework for marine power plants with weak power grids

Stian Skjong [a] and Eilif Pedersen [b]

[a]Department of Fisheries and New Biomarine Industry, SINTEF Ocean, Ålesund, Norway; [b]Department of Marine Technology, Norwegian University of Science and Technology, Trondheim, Norway

ABSTRACT

In this work, we discuss and demonstrate how multi-engine marine power plants with weak power grids efficiently can be set up and simulated in a distributed co-simulation framework. To facilitate configuration switching such as starting and stopping, connecting and disconnecting arbitrary gensets online, the generator models are modelled as hybrid causality component models. This implementation enables seamless and energy conservative model switching. Also, the proposed simulator framework is scalable such that the number of gensets in the power plant can be set by a single parameter, which automatically scales the power management system and the tailored simulator master algorithm accordingly. To control the number of active gensets being connected to the power grid while running the simulation, a simple mixed integer linear programming formulation is proposed. A simulation case study including a marine power plant configuration with four equal-sized gensets is conducted in the end to demonstrate the features of the proposed simulator framework, which also can be applied to, e.g. a small wind farm, or an isolated number of islands with interconnected power generators.

ARTICLE HISTORY
Received 29 July 2020
Accepted 27 August 2022

KEYWORDS
Marine power plant; Hybrid generator models; Co-simulation; Power management system; Mixed integer linear programming; Power plant control

1. Introduction

The maritime industry is facing both environmental and performance challenges these days. To respond to these challenges novel, possibly more efficient technologies and system solutions are introduced at a high rate. The challenge to evaluate and optimize these more complex systems in real operations have introduced new requirements for the software tools for this type of integrated system analysis (Bouman et al. 2017). Such tools should help the vessel designers to reduce environmental footprints by enabling rapid evaluation of vessel concepts based on given operational profiles. Such simulation studies are also influenced by stochastic environmental loads, being one of the most important design considerations.

Virtual prototyping, using multi-disciplinary integrated systems modelling and simulation, has the potential to reduce development cost and optimize total system performance. However, this requires availability of proper simulation models, computer software enabling fast assembly of models into large complete system simulators and not at least competence and training using such simulators efficiently (Skjong 2017; Skjong et al. 2017b). Virtual prototyping has been widely used in the automotive (Abel et al. 2012; Krammer et al. 2015; Winter et al. 2015) and aerospace industry (Sogandares 2002) for decades. Now, the maritime industry is turning towards the same approach even though the maritime industry is more geared towards tailor made products based on customer requirements. Hence, virtual prototyping might be even more interesting for this industry.

During the last years, several initiatives towards virtual prototyping for the maritime industry has been reported. The *ViProMa* project (Sadjina et al. 2018) investigated the use of *distributed co-simulations* (Gomes et al. 2017) in virtual prototyping of maritime system and operations using the *Functional Mock-Up Interface* (FMI) co-simulation standard (Blochwitz et al. 2011). This standard facilitates connecting distributed sub-simulators (FMUs), each being solved independent of each other and exchanges data according to a predefined connection scheme only at given discrete communication points. The FMI standard also enables connection of hardware in the simulator for hardware in the loop (HIL) testing (Skjong and Pedersen 2017b) and the use of multi-disciplinary black-box simulator models. The *Open Simulator Platform* (OSP) initiative (DNV GL 2020) has taken the approach even further inviting major competitors in the maritime industry to work towards a common open simulator platform, also for virtual prototyping purposes, based on results from the ViProMa project.

Here, the main focus is given to virtual prototyping of weak power plants, the hearts of all modern marine vessels. A simulator of a complex marine power plant is crucial in order to optimize the vessel's performance in different operations and conditions. The main issue with such simulators, representing weak power plants, is that they usually either have specific implementations which do not enable event-based operations such as starting and stopping of generators, or that they are not able to simulate in real-time because of small time constants from the weak power grid, which are included to enable event-based operations (Skjong and Pedersen 2017a). Moreover, connecting such simulators as submodules in larger simulations for full-system analysis purposes, would not be a generic and trivial task.

Each of the component models that constitute a power plant are thoroughly documented in the literature, often with either focus on control applications (Machowski et al. 2008) or studies of dynamical properties and responses also involving the vessel (Bø et al. 2015; Yum et al. 2016). In Sahm (1979) a synchronous generator model is modelled in the $(d, q, 0)$-reference frame, using the bond graph modelling theory (et al. 2006), which is further studied in Pedersen

CONTACT Stian Skjong stian.skjong@sintef.no

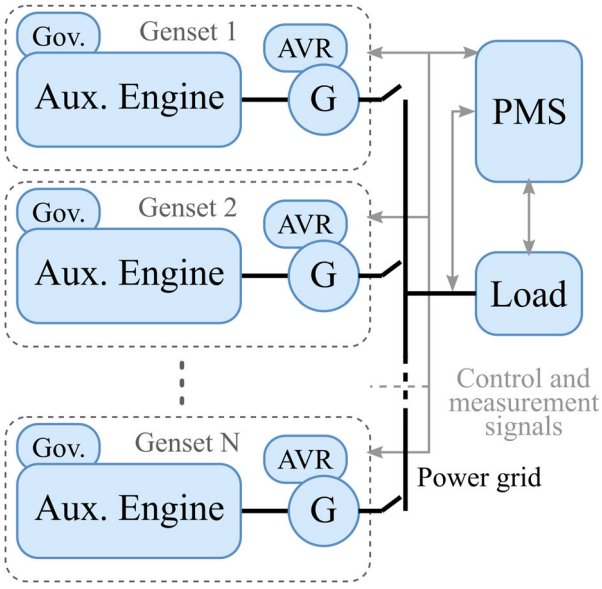

Figure 1. A marine power plant with auxiliary engines, governors (Gov), generators (G), automatic voltage regulators (AVR) a PMS and a power grid load.

(2009) and Pedersen and Pedersen (2012) with focus on marine applications. An extensive review of internal combustion engines is given in Heywood (1988). Turbo charged diesel engines in particular are discussed in Yum and Pedersen (2016). Automatic control of gensets in power plants are also discussed in the literature, ranging from control of synchronous motors (Louis 2013) and automatic voltage regulation of synchronous generators (Funabiki et al. 1991; Marwali and Keyhani 2004) to active and reactive power sharing control (Han et al. 2017) also involving the power management system (PMS). The PMS is in general quite complex (Steghöfer et al. 2013) and usually involves functionalities such as fast load reduction schemes to avoid blackouts, load sharing strategies among active producers (Marwali and Keyhani 2004), generator scheduling (Skjong et al. 2017a), generator synchronization control (Skjong and Pedersen 2017a) and surveillance, to mention a few.

This article presents a generic, simple scalable and modular object-oriented simulator framework for marine power plants with weak power grids, as shown in Figure 1. The power plant integrates all power systems onboard a modern vessel, from control- and propulsion allocation systems (Skjong and Pedersen 2017c) to power consumers such as deck machinery and propulsion systems. This modular framework closes the gap between real-time solvable simulations and event-based power plant operations, and is a significant contribution towards rapid prototyping of new machinery configurations in new-builds, where *time-to-market* is of importance. This article builds on the results presented in Skjong and Pedersen (2017a), where a real-time capable marine power plant model with weak power grid is presented, which also enables event-based operations such as starting and stopping of arbitrary gensets. This, through the use of *hybrid causality models*.

Hybrid causality models are a subset of hybrid models (Goebel and Sanfelice 2012), a type of switched models (Edström 1999) that can propagate in both continuous and discrete time. The hybrid causality models have the property of changing their interface causality online during a simulation, as will be discussed in more detail in Section 2.1. The use of hybrid causality generator models in the power plant simulator is crucial for maintaining computational speed since small time constants in the power grid can be neglected without affecting event-based operation capabilities. However, by excluding the power grid capacitive effects one of the power producers or consumers must provide the power grid voltages, mathematically speaking. In the simulator framework presented in this article, one of the active generators (gensets) provide the power grid voltage, and the rest of the active gensets and the power grid load give currents in feedback. To facilitate transient power plant operations, such as connecting/disconnecting arbitrary gensets to/from the power grid, the generator models must have the property of switching between outputting voltages and currents to the power grid, which is why the generators are here implemented as hybrid causality models.

Here, in comparison to the work presented in Skjong and Pedersen (2017a), the main contributions are the addition to the framework that makes it simply scalable with respect to the number of generators, and the generic generator scheduling algorithm. The framework is also here designed with respect to co-simulation applications, which enables full-system analyses since other systems such as the vessel's hull and various deck machinery, possibly modelled in different modelling and simulation software, can be included in the study when using a common co-simulation standard. This, without reducing the computational time since each component is solved in parallel in between the co-simulation communication time-steps. A case study with a *proof-of-concept* implementation of the proposed framework in the Python programming language is given in the end of this article. It should be noted that the implementation itself is not considered a contribution in this article since it lacks parallelization of submodule calculations, does not support externally imported simulation modules in a simple manner. It is only intended for testing the proposed framework, the proposed generator scheduling algorithm, and to be used as a pesudo-code for improved future implementations.

Even though the main focus is given to the simulator framework, the review and discussion of the generic and modular components are also considered valuable. Also note that less focus is given to control systems, except for the proposed generator scheduling algorithm. Hence, only simple PID-based control laws are considered here. However, because of the modularity of the proposed framework, the control systems can simply be replaced by more sophisticated ones. This also makes the proposed simulator framework a foundation for further research on power plant control systems.

2. Modelling of power plant components

This section concerns modelling of the main components of a typical marine power plant, namely the generator model, the auxiliary engine model and the power grid load model. A short introduction to hybrid causality models are given in the following, before presenting and discussing the mathematical models.

2.1. Hybrid causality models

A hybrid causality model can alter between its inputs and outputs during a simulation without introducing algebraic loops. To best illustrate this, consider a simple mass-damper-spring system influenced by an external force, e.g.

$$\dot{x} = v \tag{1a}$$

$$\dot{v} = \frac{1}{m}(F - bv - kx) \tag{1b}$$

where x and v are the position and the velocity of the mass, respectively, m is the mass, F is the external force given as an input to the system, b is the damping coefficient, k is the spring stiffness and the output from the model is chosen to be the velocity of the mass. This set of ordinary differential equations constitutes the *integral causality*

model representing the system. If the inputs and outputs are altered, the model of the mass-damper-spring system is changed to

$$\dot{x} = v \tag{2a}$$
$$F = m\dot{v} + bv + kx \tag{2b}$$

Now, the velocity of the mass is given as a model input and the force is given in feedback. Here, the ordinary differential equation and the differential algebraic equation constitute the *differential causality model* of the system. However, as can be seen in (2b) the input needs to be differentiated to calculate the output force F. This can be achieved by using a low-pass filter with differential effect where its transfer function is given as

$$\frac{y}{u} = \frac{s}{Ts+1} \tag{3}$$

where y and u is the transfer function output and input, respectively, and where T is the low-pass filter time constant. Hence, by setting $u = v$ in the filter, a filtered acceleration is given as output. By combining (2a) and (3), the system can be rewritten in the time plane as

$$\dot{x} = v \tag{4a}$$
$$\dot{\xi} = -\frac{1}{T}(\xi - v) \tag{4b}$$
$$F = -\frac{m}{T}(\xi - v) + bv + kx \tag{4c}$$

and contains only ordinary differential equations, the same amount as in (1a), without differential algebraic loops in the model interface. Hence, a hybrid causality model of the mass-damper-spring system can switch between (1a) and (4a) online during a simulation to alter the model interface. However, note that it is important that the filter time constant T is chosen properly to avoid filtering out important dynamics, or introducing large phase lags, and that the initial conditions are chosen to be power conserving to avoid introducing wrongful energy to the system. Details regarding this is considered out of scope here, but are thoroughly documented in Skjong (2017) and Skjong and Pedersen (2016). The same approach for reformulating differential causalities using the filter in (3) is to be applied to the generator model to enable transient power plant operations, and is discussed in the following.

2.2. Synchronous generator models

The integral causality model for a synchronous generator can according to Skjong and Pedersen (2017a) be expressed in the $(d, q, 0)$-reference frame as

$$\dot{\boldsymbol{\psi}} = -\omega_m \boldsymbol{D}\boldsymbol{\psi} - \boldsymbol{R}\boldsymbol{i} + \boldsymbol{E}\boldsymbol{u}_{dq} + \boldsymbol{b}u_f \tag{5a}$$
$$\boldsymbol{i} = \boldsymbol{L}^{-1}\boldsymbol{\psi} \tag{5b}$$

where ω_m is the engine speed, $\boldsymbol{\psi} = [\psi_d, \psi_q, \psi_f, \psi_D, \psi_Q]^\top$ is the magnetic fluxes for d, q, the field and the damping in d and q, respectively, $\boldsymbol{i} = [i_d, i_q, i_f, i_D, i_Q]^\top$ is the current vector, $\boldsymbol{u}_{dq} = [u_d, u_q]^\top$ is the voltage vector containing the voltages for d and q, u_f is the generator field voltage that controls the generator rms voltage and

$$\boldsymbol{D} = \begin{bmatrix} 0 & -n_p & 0 & 0 & 0 \\ n_p & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{E} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$
$$\boldsymbol{R} = \begin{bmatrix} R_d & 0 & 0 & 0 & 0 \\ 0 & R_q & 0 & 0 & 0 \\ 0 & 0 & R_f & 0 & 0 \\ 0 & 0 & 0 & R_D & 0 \\ 0 & 0 & 0 & 0 & R_Q \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \tag{6}$$
$$\boldsymbol{L} = \begin{bmatrix} L_d & 0 & L_{df} & L_{dD} & 0 \\ 0 & L_q & 0 & 0 & L_{qQ} \\ L_{df} & 0 & L_f & L_{fD} & 0 \\ L_{dD} & 0 & L_{fD} & L_D & 0 \\ 0 & L_{qQ} & 0 & 0 & L_Q \end{bmatrix}$$

where $\boldsymbol{R}$ is the internal resistance matrix, $\boldsymbol{L}$ is the inductance matrix and n_p is the number of pole pairs in the generator. The model outputs are the currents i_d and i_q.

Correspondingly, the equations for the generator model that outputs the voltages u_d and u_q can be expressed as

$$\boldsymbol{u}_{dq} = \dot{\boldsymbol{\psi}}_{dq} + \omega_m \boldsymbol{D}_{dq}\boldsymbol{\psi}_{dq} + \boldsymbol{R}_{dq}\boldsymbol{i}_{dq} \tag{7a}$$
$$\dot{\boldsymbol{\psi}}_{fDQ} = -\boldsymbol{R}_{fDQ}\boldsymbol{i}_{fDQ} + \boldsymbol{b}_{fDQ}u_f \tag{7b}$$

where $\boldsymbol{u}_{dq} = [u_d, u_q]^\top$, $\boldsymbol{\psi}_{dq} = [\psi_d, \psi_q]^\top$, $\boldsymbol{i}_{dq} = [i_d, i_q]^\top$, $\boldsymbol{\psi}_{fDQ} = [\psi_f, \psi_D, \psi_Q]^\top$, $\boldsymbol{i}_{fDQ} = [i_f, i_D, i_Q]^\top$ and

$$\boldsymbol{R}_{dq} = \begin{bmatrix} R_d & 0 \\ 0 & R_q \end{bmatrix}, \quad \boldsymbol{D}_{dq} = \begin{bmatrix} 0 & -n_p \\ n_p & 0 \end{bmatrix}$$
$$\boldsymbol{R}_{fDQ} = \begin{bmatrix} R_f & 0 & 0 \\ 0 & R_D & 0 \\ 0 & 0 & R_Q \end{bmatrix}, \quad \boldsymbol{b}_{fDQ} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{8}$$

It is then possible to rearrange (5b) such that

$$\begin{bmatrix} \boldsymbol{\psi}_{dq} \\ \boldsymbol{i}_{fDQ} \end{bmatrix} = \boldsymbol{Z} \begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \tag{9}$$

where $\boldsymbol{Z}$ is given as in (11).

This differential causality model of the generator contains only three states while the integral causality model contains five. However, based on the transfer function-based differentiation discussed in Section 2.1, the differential algebraic equations in (7a) can be reformulated to differential equations. By defining

$$\boldsymbol{\psi}_{dq} = \boldsymbol{Z}_{dq} \begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \tag{10}$$

where $\boldsymbol{Z}_{dq} \in \mathcal{R}^{2\times 5}$ contains the two first rows in $\boldsymbol{Z}$ given as

$$\boldsymbol{Z} = \begin{bmatrix} Z_{11} & 0 & \frac{L_{dD}L_{fD} - L_D L_{df}}{L_{fD}^2 - L_D L_f} \\ 0 & \frac{L_{qQ}^2 + L_Q L_q}{L_Q} & 0 \\ \frac{L_{dD}L_{fD} + L_D L_{df}}{L_{fD}^2 - L_D L_f} & 0 & -\frac{L_D}{L_{fD}^2 - L_D L_f} \\ \frac{L_{fD}L_{df} + L_{dD}L_f}{L_{fD}^2 - L_D L_f} & 0 & \frac{L_{fD}}{L_{fD}^2 - L_D L_f} \\ 0 & -\frac{L_{qQ}}{L_Q} & 0 \end{bmatrix} \begin{bmatrix} \frac{L_{fD}L_{df} - L_{dD}L_f}{L_{fD}^2 - L_D L_f} & 0 \\ 0 & \frac{L_{qQ}}{L_Q} \\ \frac{L_{fD}}{L_{fD}^2 - L_D L_f} & 0 \\ -\frac{L_f}{L_{fD}^2 - L_D L_f} & 0 \\ 0 & \frac{1}{L_Q} \end{bmatrix} \tag{11}$$

where

$$Z_{11} = \frac{L_f L_{dD}^2 - 2L_{dD}L_{fD}L_{df} + L_d L_{fD}^2 + L_D L_{df}^2 - L_D L_d L_f}{L_{fD}^2 - L_D L_f} \tag{12}$$

and that $\dot{\hat{\boldsymbol{\psi}}}_{dq}$ is the derivative of $\boldsymbol{\psi}_{dq}$ obtained from the transfer function.

This means that $\boldsymbol{\psi}_{dq}$ can be calculated from (9), differentiated and inserted into (7a) to obtain $\boldsymbol{u}_{dq}$. The current vector $\boldsymbol{i}_{fDQ}$ is found by first obtaining $\dot{\boldsymbol{\psi}}_{fDQ}$ from integrating (7b) and inserting it into (9). Hence, the reformulated differential causality model can be expressed as

$$\dot{\boldsymbol{\psi}}_{fDQ} = \boldsymbol{b}_{fDQ}u_f - \boldsymbol{R}_{fDQ}\boldsymbol{i}_{fDQ} \tag{13a}$$

$$\dot{\boldsymbol{\xi}}_{dq} = -\frac{1}{T}\boldsymbol{\xi}_{dq} + \frac{1}{T}\boldsymbol{Z}_{dq}\begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \tag{13b}$$

$$\boldsymbol{i}_{fDQ} = \boldsymbol{Z}_{fDQ}\begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \tag{13c}$$

$$\boldsymbol{u}_{dq} = -\frac{1}{T}\boldsymbol{\xi}_{dq} + \boldsymbol{R}_{dq}\boldsymbol{i}_{dq} + \left(\frac{1}{T}\boldsymbol{I}_{2\times 2} + \omega_m \boldsymbol{D}_{dq}\right)\boldsymbol{Z}_{dq}\begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \tag{13d}$$

where $\boldsymbol{Z}_{dq} \in \mathcal{R}^{2\times 5}$ and $\boldsymbol{Z}_{fDQ} \in \mathcal{R}^{3\times 5}$ contains the two first rows and the three last rows in $\boldsymbol{Z}$ given in (11), respectively, and $\boldsymbol{I}_{2\times 2}$ is the identity matrix of size 2. Note that $\hat{\boldsymbol{\psi}}_{dq} = \boldsymbol{\xi}_{dq}$ in (13a). The number of states are now restored to five for the reformulated differential causality model.

To enable smooth switching between the two causality orientations online during a simulation, proper initial conditions must be derived. When switching from the integral causality model to the reformulated differential causality model, the power conserving initial conditions for the simulation time t_0 for the states in the reformulated differential causality model are given as

$$\begin{aligned} \boldsymbol{\psi}_{fDQ}^{rD}(t_0) &= \boldsymbol{\psi}_{fDQ}^{I}(t_0) \\ \boldsymbol{\xi}_{dq}(t_0) &= -T\boldsymbol{u}_{dq}^{I}(t_0) + T\boldsymbol{R}_{dq}\boldsymbol{i}_{dq}^{I}(t_0) \\ &\quad + \left(\boldsymbol{I}_{2\times 2} + T\omega_m(t_0)\boldsymbol{D}_{dq}\right)\boldsymbol{Z}_{dq}\begin{bmatrix} \boldsymbol{i}_{dq}^{I}(t_0) \\ \boldsymbol{\psi}_{fDQ}^{I}(t_0) \end{bmatrix} \end{aligned} \tag{14}$$

Note that the superscripts *rD* and *I* are abbreviations for the variables in the reformulated differential causality model and the integral causality model, respectively, to separate equal variable names. Correspondingly, the power conserving initial conditions when switching from the reformulated differential causality model to the integral causality model at simulation time t_0 are given as

$$\begin{aligned} \boldsymbol{\psi}_{fDQ}^{I}(t_0) &= \boldsymbol{\psi}_{fDQ}^{rD}(t_0) \\ \boldsymbol{\psi}_{dq}^{I}(t_0) &= \boldsymbol{Z}_{dq}\begin{bmatrix} \boldsymbol{i}_{dq}^{rD}(t_0) \\ \boldsymbol{\psi}_{fDQ}^{rD}(t_0) \end{bmatrix} \end{aligned} \tag{15}$$

2.3. Auxiliary engine models

The auxiliary engine models are based on simple equations given in Heywood (1988). The effective engine power can be expressed as

$$P_e = \dot{m}_f h_n \eta = T_m \omega_m \tag{16}$$

where $\dot{m}_f$ is the fuel flow rate, h_n is the lower heating value of the fuel, η is the effective thermal efficiency and T_m is the torque. By rearranging the equation, the mean torque generated by the combustion process can be expressed as

$$T_m = \frac{\dot{m}_f h_n \eta}{\omega_m} \tag{17}$$

where the fuel flow rate is given as

$$\dot{m}_f = m_{\text{inj}}\frac{\omega_m}{2\pi k} \tag{18}$$

Note that m_{inj} is the amount of fuel injected per cycle and k is a parameter distinguishing two-stroke engines from four-stroke engines, $k = 1$ for two-stroke and $k = 2$ for four-stroke. The thermal efficiency can be expressed as

$$\eta = \frac{1}{b_e(P_e)h_n} \tag{19}$$

where $b_e(P_e)$ is the specific fuel consumption as a function of effective engine power, and can be measured for a specific engine given the engine speed. By assuming a four-stroke engine, the torque can be expressed as

$$T_m = \frac{m_{\text{inj}}}{4\pi b_e(P_e)} \tag{20}$$

The set of differential equations representing the auxiliary engine can then be expressed as

$$\dot{\theta}_m = \omega_m \tag{21a}$$

$$\dot{\omega}_m = \frac{1}{J_m + J_G}\left(T_m - b_f\omega_m - b_b\omega_m^n - T_e\right) \tag{21b}$$

where θ_m is the engine angle, J_m is the inertia of the engine, J_G is the inertia of the generator, T_e is the electromagnetic torque given as

$$T_e = (\psi_d i_q - \psi_q i_d)n_p, \tag{22}$$

b_f is a friction parameter and b_b is the braking effect when the engine is choked. Note that $b_b \approx 0$ when $m_{\text{inj}} \neq 0$.

When multiple gensets are connected to the power grid, the phase difference between them are crucial for active load sharing, as will be discussed in more detail in section 3.1. Here, the phase between the generators are obtained by adding a transformation on the generator model voltages and currents such that the power grid voltages and currents are given as

$$\begin{aligned} \boldsymbol{u}_{\mathrm{PG}} &= \boldsymbol{S}(\phi_i)\boldsymbol{u}_{Gi} \\ \boldsymbol{i}_{\mathrm{PG}} &= \boldsymbol{S}(\phi_i)\boldsymbol{i}_{Gi} \end{aligned} \tag{23}$$

where $\boldsymbol{u}_{Gi}$ and $\boldsymbol{i}_{Gi}$ are the voltage and current vectors for generator i, respectively, and where

$$\boldsymbol{S}(\phi_i) = \begin{bmatrix} \cos(\phi_i) & -\sin(\phi_i) \\ \sin(\phi_i) & \cos(\phi_i) \end{bmatrix} \tag{24}$$

is the phase transformation and ϕ_i is the phase difference between the leading generator, the generator with voltage output causality, and generator i, and is be expressed as

$$\phi_i = \int_0^t (\omega_l - \omega_{mi})\,\mathrm{d}t \tag{25}$$

where ω_l and ω_{mi} are the speeds of the leading generator and generator i, respectively. Note that ϕ_i is usually normalized to $\pm\pi$. More details regarding these phase transformations are given in Skjong (2017), Skjong and Pedersen (2017a) and will not be given more attention here.

2.4. Power grid load and open circuit load

The power grid load represents the power consumers connected to the power grid. In a marine power plant, these consumers range from thrusters and deck machinery to hotel loads and auxiliary systems needed to operate the vessel. Here, it is assumed that the active and reactive power consumption can be set and based on the power grid voltage, and where currents are given in feedback. The active power and reactive power grid load in the $(d, q, 0)$-reference frame are here expressed as

$$P = \boldsymbol{u}_{d,q}^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \boldsymbol{i}_{d,q} \tag{26a}$$

$$Q = \boldsymbol{u}_{d,q}^T \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \boldsymbol{i}_{d,q}, \tag{26b}$$

respectively. Note that reactive power is related to impedance in the (a, b, c)-reference frame. By solving (26a) and (26b) with respect to $\boldsymbol{i}_{d,q}$ the current given in feedback is given as

$$\boldsymbol{i}_{d,q} = \frac{1}{||\boldsymbol{u}_{d,q}||_2^2} \begin{bmatrix} Pu_d + Qu_q \\ Pu_q - Qu_d \end{bmatrix} \tag{27}$$

where

$$\|\boldsymbol{u}_{d,q}\|_2^2 = u_d^2 + u_q^2 \tag{28}$$

is the square of the $\mathcal{L}_2$-norm. Care must be taken to avoid dividing by zero at the start of the simulation. Also, low-pass filters are used to filter the input voltages to avoid algebraic loops. Note that the set-points for P and Q are also low-pass filtered to add dynamics to the load demands to make them more realistic. When more sophisticated models for power plant loading are used, such as a thruster model or deck machinery, the low-pass filter can be neglected.

Whenever a generator model is running but not connected to the power grid, it is loaded with an open circuit load. This load has a high ohmic resistance, which in reality is approaching infinity. Since an infinite open circuit resistance is not possible to realize in mathematics without producing 'NaN'-values, the resistance is set to a finite number. Hence, the open circuit load currents are calculated as

$$i_j = \frac{u_j}{R_{\mathrm{oc}}}, \quad \forall\, j \in d, q \tag{29}$$

where R_{oc} is the open circuit load resistance.

3. Control systems

The power plant control systems include governors, automatic voltage regulators and a PMS. Note that also a simulator controller is needed, e.g. for controlling the causality of the hybrid generators. This will be incorporated in the PMS, as discussed later. These control systems can vary quite much in both functionality and implementation, and details are usually considered business secrets. A short presentation of these systems is given below.

3.1. Engine governor

The speed of an auxiliary engine is regulated by governors, controlling the amount of injected fuel m_{inj}. In general, the governors are part of a large and sophisticated engine control unit (ECU). Both the ECU itself and active power sharing is here considered out of scope. When a generator is connected to the power grid, the control objective for the governor is to match the engine speed to the power grid frequency, f_{PG}, resulting in a reference speed expressed as

$$\omega_{\mathrm{ref}} = \frac{2\pi f_{\mathrm{PG}}}{n_p} \tag{30}$$

This reference speed is here given as set-point to a PI controller with integrator anti-windup functionalities, constituting the governor. Note that when a generator is running in standby as a spinning reserve, it usually have a lower speed reference than when being active. However, when a generator is synchronizing to the power grid it must also match the phase angle to the leading generator, as discussed in Section 2.3. Hence, the addition to the control error for synchronizing generator i, being fed to the governor by the PMS, is here given as

$$e_{Si} = K_S(u_{dl} - u_{di}) + I_S \int_0^t (u_{dl} - u_{di})\,\mathrm{d}t \tag{31}$$

where K_S is a proportional synchronization gain, I_S is an integral synchronization gain, and u_{dl} and u_{di} are the d-voltage components from the leading generator and generator i, respectively. Note that controlling the difference in d-voltages to zero is the same as controlling the difference in phase angles to zero (Skjong and Pedersen 2017a). When generator i is synchronized to the power grid, active load sharing is initiated by the PMS. This is also an addition to the control objective for the governor, provided by the PMS, and is here for generator i given as

$$e_{Pi} = K_P\left(\sigma\frac{P}{P_{\mathrm{cap}}} - \frac{P_i}{P_{\mathrm{cap},i}}\right) \tag{32}$$

where K_P is a proportional gain and $\sigma \in \{0, 1\}$ is a control variable. σ equals 1 if the generator is connected to the power grid but 0 if it is either active but to be de-synchronized, in standby as a spinning reserve or not running. P is the total active power grid load, P_{cap} is the current planned maximal capacity of active power, $P_{\mathrm{cap},i}$ is the maximal capacity of active power for generator i and P_i is the

current active power produced by generator i. Note that if a generator is to be de-synchronized from the power grid, it should not be included in P_{cap} and that only equal sharing active load among the active generators is considered in (32).

3.2. Automatic voltage regulator

The generator rms voltages are controlled by automatic voltage regulators (AVRs) through the generator field voltages, using PID controllers with anti-windup. An AVR has also a second control objective when a generator is active and connected to the power grid, namely to share the reactive power grid load among the active generators. As for the governor, the AVR receives an additional control error from the PMS to obtain this, and the resulting control error is here expressed as

$$e_{Qi} = K_Q \left(\frac{\sigma}{N_A} - \frac{Q_i}{Q} \right) + I_Q \int_0^t \left(\frac{\sigma}{N_A} - \frac{Q_i}{Q} \right) \mathrm{d}t \tag{33}$$

where K_Q is a proportional gain and $\sigma \in \{0, 1\}$ is a control variable being 1 if the generator is active and 0 if it is active but to be de-synchronized, in standby as a spinning reserve or not running. N_A is the number of current scheduled active generators, Q_i is the reactive power for generator i, Q is the reactive power for the power grid load and I_Q is an integral gain. Note that N_A does not include generators that are active but is to be de-synchronized and that only equal sharing of reactive power is considered in (33).

As discussed, both the governor and the AVR are closely cooperating with the PMS which feeds them with reference set-points and load sharing control objectives. This will be illustrated in Section 4. In the following, a short presentation of the PMS is given with main focus on the operation of the entire power plant.

3.3. Power management system

The PMS is a supervisory control algorithm, containing lots of logic functions, and interacts with the lower level controllers, such as the governor and the AVR, for controlling the entire power plant. Even though it contains lots of supervisory control functions, only generator scheduling, synchronization and load sharing will be treated here. The connection between the PMS, the governor and the AVR have been briefly discussed in Sections 3.1 and 3.2, but is here presented with focus on the PMS.

In addition to calculate control errors for the governors and the AVRs for synchronization and load sharing purposes, the PMS also schedules the activity of the generators – which generator and the number of generators to run in active mode, producing power to the power grid, if a generator is to be synchronized or de-synchronized to the power grid, and which generator and the number of generators running in standby as spinning reserves. Here, this is achieved using a Mixed Integer Linear Programming (MILP) formulation of the scheduling problem.

The MILP formulation is a simple linear optimization formulation, here implemented as a separate algorithm and solved as a subroutine in the PMS, for determining whether or not to start or shut down a generator based on power measurements, the current power plant configuration and system constraints. It also determines which generator to start and stop based on the amount of active running hours, the amount of time each generator in the power plant is being connected to the power grid, trying to level the amount for all generators in the power plant for service and maintenance reasons. For safety reasons, the algorithm may also make sure that there is always one generator running in standby to speed up the process of connecting one more generator to the power grid if needed, if possible. This, at the same time as increasing the power plant efficiency by reducing the active capacity, such that the active generators are not running with too low load. Here, the MILP-based algorithm is given as

$$\min_{\boldsymbol{x_s}, \boldsymbol{x_a}} \frac{\boldsymbol{c}^\top \boldsymbol{x}_a}{\max(\boldsymbol{c})} + \frac{\alpha \boldsymbol{h}^\top}{\max(\boldsymbol{h})} (\beta \boldsymbol{x}_s + \boldsymbol{x}_a) \tag{34a}$$

subject to

$$0 \leq \boldsymbol{x}_s(i) + \boldsymbol{x}_a(i) \leq 1, \quad \forall\, i \in N \tag{34b}$$

$$\eta P \leq \sum_{i=1}^{N} \boldsymbol{c}(i) \boldsymbol{x}_a(i) \tag{34c}$$

$$1 - \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}_a(i) \leq \sum_{i=1}^{N} \boldsymbol{x}_s(i) \tag{34d}$$

$$0 \leq \sum_{i_1}^{N} \boldsymbol{y}_a(i)[\boldsymbol{y}_a(i) - \boldsymbol{x}_a(i)] + [1 - \boldsymbol{y}_a(i)]\boldsymbol{x}_a(i) \leq 1 \tag{34e}$$

$$0 \leq \sum_{i_1}^{N} \boldsymbol{y}_s(i)[\boldsymbol{y}_s(i) - \boldsymbol{x}_s(i)] + [1 - \boldsymbol{y}_s(i)]\boldsymbol{x}_s(i) \leq 2 \tag{34f}$$

$$\boldsymbol{x}_s, \boldsymbol{x}_a \in \{0, 1\} \tag{34g}$$

where $\boldsymbol{x_s}, \boldsymbol{x_a} \in [0, 1]$ are decision variable vectors with sizes equal to the number of generators in the power plant, being either 0 or 1, telling whether or not a generator is in standby or active, respectively. The vectors $\boldsymbol{c}$ and $\boldsymbol{h}$ have the same sizes as $\boldsymbol{x_s}$ and $\boldsymbol{x_a}$, but contains the maximal capacity and the running hours for each genset, respectively. α is a weighting gain between the capacity cost and the active running hours cost, and $0 < \beta < 1$ is a constant added to ensure that the generator with most running hour is set in idle if all generators are active and one is to be disconnected from the power grid. $\eta \geq 1$ is a safety factor for ensuring that there is enough available power in the power grid and should be determined based on stochastic power plant loading considerations. However, this is considered out of scope here. P is the measured active power grid load, which will have stochastic variations in a marine power plant. The vectors $\boldsymbol{y_s}, \boldsymbol{y_a} \in [0, 1]$ are measurement vectors with sizes equal to the number of generators in the power plant, containing information about if a generator is in standby mode ($\boldsymbol{y}_s$) or active ($\boldsymbol{y}_a$).

The constraints in the MILP-based algorithm are given in (34b)–(34g), where the first constraint makes sure that a generator can either be in standby, active or not running at all. The second constraint ensures that there is always enough power available in the power grid, as long as the total power plant capacity is not overrun. This should never happen since it causes a blackout. Hence, fast load reduction functionalities in the PMS should be considered to make sure this never happens, but is considered out of scope here. The third constraint in the MILP-based algorithm makes sure that there is always one generator in standby, as long as not all generators are active while the fourth constraint restricts the MILP-based algorithm to allow only one change in $\boldsymbol{x_a}$ in comparison to the measured configuration $\boldsymbol{y_a}$ at a time. Likewise, the fifth constraint allows the algorithm to make in total two changes in $\boldsymbol{x_s}$ in comparison to $\boldsymbol{y_s}$ at a time. The last constraint is a binary constraint, stating that $\boldsymbol{x}_s$ and $\boldsymbol{x}_a$ are vectors consisting only of binary values. Also note that $\boldsymbol{y}_s$ and $\boldsymbol{y}_a$ have this property, but since these are measurement vectors they are given as input to the problem formulation and considered constant between each run.

The output from the MILP-based algorithm is the decision variable vectors $\boldsymbol{x_s}$ and $\boldsymbol{x_a}$ which are fed to another PMS function that use a combination of logic functions and measurements, to change

the composition of running generators. Note that the MILP-based algorithm is only initiated when the PMS is not working on synchronizing or de-synchronizing a generator to or from the power grid.

When the results from running the MILP-based algorithm suggest to connect or disconnect a generator from the power grid, a synchronization or de-synchronization algorithm is initiated by the PMS, respectively. In the synchronization algorithm, the reference speed for the auxiliary engine is updated to match the power grid frequency, as given in (30), in addition to feeding the governor with the synchronization control error given in (31). At this point, the generator is running with an open circuit load, as given in (29), having voltages as output. Based on phase measurements, frequency measurements and voltage measurements from the leading generator, the active generator with voltage outputs, the synchronization algorithm closes the circuit breaker, connecting generator i to the power grid, when

$$|\phi_i| \leq \phi_{\max} \tag{35a}$$

$$\left|\frac{\mathrm{d}\phi}{\mathrm{d}t}\right| \leq \dot{\phi}_{\max} \tag{35b}$$

$$|V_{\mathrm{rms},l} - V_{\mathrm{rms},i}| \leq \Delta V_{\mathrm{rms,max}} \tag{35c}$$

where ϕ_i is as given in (25), $\phi_{\max}$ is the maximal allowed phase angle, $\dot{\phi}_{\max}$ is the maximal allowed phase angle rate, $V_{\mathrm{rms},l}$ and $V_{\mathrm{rms},i}$ is the rms voltages for the leading generator and generator i, respectively, and $\Delta V_{\mathrm{rms,max}}$ is the maximal allowed difference in rms voltage. However, instead of using the phase and the corresponding phase rate as given in (35a) and (35b), respectively, to determine when to connect the synchronizing generator, a comparison of d-voltages and engine speeds are used here and the new criteria can be expressed as

$$|u_{d,l} - u_{d,i}| \leq \Delta u_{d,\max} \tag{36a}$$

$$|\omega_{m,l} - \omega_{m,i}| \leq \Delta\omega_{m,\max} \tag{36b}$$

where $\Delta u_{d,\max}$ is the maximal allowed absolute difference in d-voltages and $\Delta\omega_{m,\max}$ is the maximal allowed absolute difference in engine speeds. Note that when a generator is connected to the power grid, the PMS also closes the respective circuit breaker, changes the causality for the generator such that it has currents as outputs, disengages the synchronization control and starts the load sharing functionalities.

When a generator is to be de-synchronized from the power grid, the generator must transfer its loads to the other active generators, which is done by setting $\sigma = 0$ and updating P_{cap} and N_A in (32) and (33), respectively. When the load is low enough, e.g.

$$P_i \leq P_{\max} \tag{37}$$

where $P_{\max}$ is the maximal allowed active power for de-synchronizing and disconnecting a genset, the PMS opens the circuit breaker and changes the generator causality to output voltages to the open circuit load. Note that a similar criterion can be added for the reactive power for disconnecting a genset from the power grid as well. Also note that if the generator that is to be de-synchronized is the leading generator, the lead is transferred to another generator before disconnecting it from the power grid. This is also an additional functionality given the PMS, but note that this is a functionality only needed in the simulator since we are using hybrid causality generator models. Also note that the PMS also initiates start, stop and standby activities for the generators, but this is trivial operations when they are not connected to the power grid and only involves changing set-points for the lower level controllers.

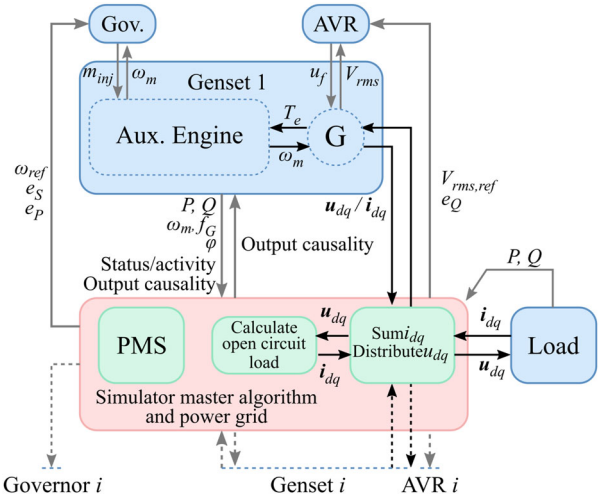

Figure 2. Sketch illustrating the connections between the components in the simulator. Note that the blue boxed represent co-simulation slaves, the red box the simulation master algorithm including the weak power grid and the green boxed the calculation routines that do not perform independent time steps.

The connections between the PMS and the lower level controllers will be further illustrated in Section 4.1 with an object-oriented simulator framework in the main scope.

4. Simulator framework

The main objective in this section is to construct a scalable simulator framework where the number of generators is set with a single parameter. This is achieved by implementing the models and the control systems as class objects in the simulator framework and by distributing the total system in a co-simulation. Then, arrays with length equal to the number of generators in the power plant can be used to hold the class object instances, where each class with ordinary differential equations has its own local numerical integration routine. Note that such class object instances are hereafter referred to as slaves in the co-simulation. This system distribution also facilitates a generic simulation master algorithm containing generic logic functions for stepping each slave and exchanging data, independent of the number of generators in the power plant by the use of for-loops.

In the following, the presentation of the object-oriented marine power plant simulator framework is split into subsystems, connections between them and the simulation master algorithm.

4.1. Subsystems and connections

In the marine power plant simulator all gensets, governors, AVRs and the power grid load are considered separate co-simulation slaves, as illustrated in Figure 2. Note that these co-simulation slaves are represented by blue boxes in the figure, the simulation algorithm is represented by a red box and calculation routines that do not perform independent time steps are represented by green boxes. The co-simulation slaves in the simulator have specific and generic interfaces, and the master algorithm interfaces each slave with predefined functions (Blochwitz et al. 2011), but only the six most important functions, facilitating the presentation of the simulation master algorithm in Section 4.2, are presented here and listed with a short description in Table 1.

The slave inputs and outputs, in general, are slave-specific predefined arrays with fixed lengths and compositions where the structure known by the simulation master algorithm. These input/output (I/O) connections between the system components in the power plant are

Table 1. Generic slave interface functions.

Function	Short functional description
Instantiate()	Instantiates a slave in the simulation.
Initialize()	Initializes the slave instance with user- and simulator. Takes user-specific data, local time-step size, start time etc. as input.
DoStep()	Steps the slave to the time $t + T_d$, where T_d is the communication time step size, with the local time step size Δt.
GetData()	Sends model outputs to simulation master algorithm.
SetData()	Gets model inputs from simulation master algorithm.
Terminate()	Terminates the slave instance and cleans up memory.

Algorithm 1 Pseudocode for the object-oriented marine power plant simulator master algorithm.

```
1:  procedure RunSim(N_G, t_start, t_stop, T_d, user-specifics)
2:      Slaves=InstantiateSlaves(N_G)
3:      InitializeSlaves(Slaves, t_start, Δt, user-specifics)
4:      t = t_start
5:      while t ≤ t_stop do
6:          DoStep(Slaves, t)
7:          data=GetData(Slaves)
8:          StoreData(data, t)
9:          currents=CalcOpenCircuitLoads(data, N_G)
10:         if t ≤ t_startup then
11:             cmd=RunStartUpProcedure()
12:         else
13:             cmd=RunPMS(data, N_G)
14:             load-scenario=LoadScenario(data, t)
15:             SetData(Slaves, data, cmd, currents, load-scenario)
            end
16:         t = t + T_d
        end
17:     TerminateSlaves(Slaves)
18:     PostProcessing()
    end
```

also shown in Figure 2 – the black arrows represent power connections and grey arrows represent control signals, commands and measurements. Note that *Output causality* refers to the causality of the power connections between the generators and the power grid, and that *Status/activity* refers to the active generator state – if it is running in standby, is synchronizing to the power grid or de-synchronizing from the power grid, connected to the power grid or is not running at all. Also note that the electric power connections between generator 1 and the power grid in the figure are labelled $\boldsymbol{u}_{dq}/\boldsymbol{i}_{dq}$, meaning that the I/O configuration can either be voltage-input/current-output or vice versa.

The simulator master algorithm itself also represents the weak power grid in the power plant including the circuit breakers, in addition to control the entire co-simulation, as will be elaborated in the following.

4.2. Simulator master algorithm and PMS

Since the dynamics in both the circuit breakers and the weak power grid in the power plant are assumed negligible in this simulator, their calculations and actions, such as summing currents, distributing voltages and closing circuit breakers, are considered time-invariant. Hence, they can be performed directly in the simulation master algorithm, as illustrated in Figure 2.

The structure and the functions in the simulator master algorithm are presented through the pseudocode in Algorithm 1 where the overall simulator master algorithm is referred to as RunSim and takes the number of gensets in the power plant, N_G, as input in addition to the simulation start time t_{start}, stop time t_{stop}, the global co-simulation communication time-step size T_d and possibly other user-specific parameters considered configurable.

In Algorithm 1, the simulation master first instantiates all slaves in the simulation and stores them in separate arrays, e.g. each genset are stored in the array *Gensets*, each AVR in the array *AVRs*, etc. Note that in Algorithm 1 the array *Slaves* are used to save space, but refers to all the arrays holding all instances of specific class objects, such as all the gensets. After instantiating the slaves, it runs a separate slave initialization routine for setting initial values, simulation parameters, user-specific parameters and initial conditions. Right before starting the main simulation loop also the global propagating time variable is initialized.

In the simulation loop the master algorithm commands all slaves to perform a global time step, T_d, by calling the *DoStep()*-function in each slave, and when all the slaves have finished performing their local time steps such that their local propagating time variables match the global simulation time t, slave output data are obtained by calling the *GetData()*-functions before storing the data. Note that storing the data is here defined as a master algorithm functionality. After storing the data, simulator specific calculations are performed in the master algorithm, starting with calculating all open circuit loads for the gensets running – but disconnected from the power grid. Following, if the simulation time is less than t_{startup}, a function named *RunStartUp()* is called, which contains a start-up routine for the power plant. Typically, such a start-up routine involves starting one genset and connecting it to the power grid, connecting the load, and starting a second genset in standby as a spinning reserve for safety reasons. After this initial start-up period, the PMS is called (*RunPMS()*) and fed with relevant measurements from the slaves. Then, the load-scenario is obtained (*LoadScenario()*) – which is specified by the user before running the simulator and is a lookup-table with time-stamped values for P and Q. Furthermore, all relevant data are given as feedback to the slaves (*SetData()*). When the main simulation loop has finished all the slaves are terminated (*TerminateSlaves()*) before running an optional post-processing of the saved data (*PostProcessing()*).

In the following, a case study of a marine power plant simulator using the proposed framework is presented, implemented and tested.

5. Case study

To test the proposed object-oriented marine power plant model, a case study including four gensets and a changing power grid load with noise are to be conducted. In this case study, a sample implementation of the proposed simulator framework has been implemented in the Python programming language, where each co-simulation slave is implemented as a class such that the number of gensets – including governors and AVRs, are scaled with a single parameter, as discussed in Section 4. To solve the optimization problem in the MILP-based algorithm the Python package *pulp* in combination with the *GLPK* solver for linear and mixed integer programming (Makhorin 2012) is used. Note that the reason for implementing it in the Python programming language is purely based on practicalities. The presented framework can be implemented in many modelling and simulation software. Nevertheless, the proposed framework has its strengths when it comes to co-simulations due to the explicit, but dynamic, component connection formulation, excluding algebraic loops between system components.

5.1. Simulation set-up

In this case study, the four gensets are assumed equally sized, each, having a capacity of producing 2010 kW at a rms voltage of 690 V with a frequency of 60 Hz. Table 2 lists the main parameters in the simulator. Note that the same model parameters used in Skjong (2017, Table 4.1, p. 99) are used here.

Table 2. Main simulator parameters.

Parameter	Value
Number of gensets	4 equal gensets
Rated power	2010 kW pr. genset
n_p, number of generator pole pairs	5-
Power grid frequency	60 Hz (720 rpm)
Standby engine speed	600 rpm
Power grid rms voltage	690 V
Open circuit load, R_{oc}	100 MΩ
Field voltage capacity	±100 V
$t_{startup}$	30 s
Start-up active power plant load	1 MW, with noise
Start-up reactive power plant load	1 Mvar, with noise
α, cost function weight in MILP	0.1-
β, cost function weight in MILP	0.9-
Available power safety factor in MILP, η	1.2-
MILP run frequency	1 Hz
$\Delta u_{d,max}$, synch. criterion in PMS	0.1 V
$\Delta\omega_{m,max}$, synch. criterion in PMS	0.5 rpm
$\Delta V_{rms,max}$, synch. criterion in PMS	0.5 V
P_{max}, de-synch. criterion in PMS	5.0 W

Table 3. Mean active power grid loading scenarios.

Time slot	Mean active power
[30 s, 50 s)	1.0 MW
[50 s, 100 s)	1.8 MW
[100 s, 150 s)	2.7 MW
[150 s, 200 s)	3.6 MW
[200 s, 250 s)	5.5 MW
[250 s, 300 s)	3.6 MW
[300 s, 350 s)	2.7 MW
[350 s, 400 s)	1.8 MW
[400 s, ∞ s)	1.0 MW

In the start of the simulation, a power plant start-up procedure which starts up and connects genset 1 to the power grid and puts genset 2 in standby is implemented and the power grid load is activated when the first generator is synchronized and connected to the power grid, having a mean active and a reactive power grid load of 1 MW and 1 Mvar, respectively, also being influenced by noise. This noise represents realistic loading conditions due to environmental forces such as waves and wind acting on the marine vessel, and is included for model- and control system robustness testing purposes, having a maximal magnitude of 50 kW (kvar) with a frequency of 1 Hz. When $t \geq t_{startup}$ the PMS algorithm takes over for the start-up procedure and a power grid loading scenario, set by the user prior to the simulation, is initiated. This load-scenario changes the mean active power grid load during the simulation and is summarized in Table 3.

Each slave in the co-simulation is solved by the Runge–Kutta 2 integration method where the local solver time-step size is set to $\Delta t = 0.0001$ s, $T_d = 0.0002$ s and the total length of the simulation is set to 500 s.

5.2. Simulation results

The upper left-most plot in Figure 3 shows the rms voltage for the gensets in the power plant and a selected magnified region between $[50\,s,\ 100\,s]$ is shown in the upper right-hand most plot when genset 2 is being synchronized and connected to the power grid. Following, the active power grid load and the power produced by each genset are shown in the left-hand most plot in the second row while the plot to the right shows a magnified region of the active power grid load in the same area as for the magnified voltage plot, also showing that the active load sharing algorithm in the PMS has been activated, reducing the active power for genset 1. The reactive powers are shown in the third row in the figure, including a magnified region of the reactive powers in the right-most plot, also illustrating the activation of the reactive load sharing algorithm in the PMS. The plot in the fourth row in the figure shows the commanded field voltages for the gensets from the AVRs and the last plot in the figure shows the causality output from each genset, being either current or voltage. Note that each running gensets not being connected to the power grid has the voltage vector $\boldsymbol{u}_{dq}$ as output and the current vector $\boldsymbol{i}_{dq}$ as input.

The first plot on the first row in the figure shows that when a genset is running, either in standby, being synchronized, de-synchronized or connected to the power grid, it has a rms voltage set-point of 690 V. The magnified rms voltage plot also shows that when the second genset is connected to the power grid, the voltage fluctuations are slightly reduced since both gensets share the loading, having a larger inertia in the system. This can also be seen in the reactive power plot where the noise in reactive power from each genset is reduced with increasing number of gensets connected to the power grid. The first plot in the second row in the figure clearly shows the changing active power grid load, which is ramped up to 5.5 MW and back again to its initial value of 1 MW, in accordance with the information given in Table 3. The plot also shows that the load is shared equally between all active gensets and when a new genset is synchronized and connected to the power grid the loading of each active generator decreases. Oppositely, when a genset is being de-synchronized and disconnected from the power grid, both the active- and the reactive loading of that genset is being transferred to the other active gensets before being disconnected from the power grid. The same goes for the commanded field voltages from the AVRs shown in the plot in the fourth row in the figure. To get a better view of the corresponding phases and frequencies, the phase voltage u_a and the corresponding current i_a are shown in Figure 4 for different situations.

The left-most column of plots shown in Figure 4 shows the u_a phase voltages, the corresponding currents, i_a, and the frequencies, respectively, for generator 1 and generator 2 right after the AVR for genset 2 has been activated. As can be seen in the upper plot, the voltage is being increased, along with the frequency as shown in the lower plot. The middle plot shows the currents, and indicates that there is no current corresponding to genset 2. The middle column of plots in the figure shows the same quantities right after genset 2 has been connected to the grid. The upper plot shows that the phase voltages are overlapping whereas the middle plot shows that the current from genset 2 is being increased, due to the power sharing, at the same time as the current from genset 1 is being decreased. This is further illustrated in the last plot, showing that the frequencies are mirrored around the base frequency of the grid due to the active power sharing. The last column shows the same quantities after the power has been shared between the two active gensets, and now both the phase voltages, the corresponding currents and the frequencies for the two active gensets are overlapping.

To understand the commands from the MILP-based algorithm, it is easier to look at the auxiliary engine speeds driving the generators, which is shown in the first plot in Figure 5. The PMS and the MILP-based algorithm are activated at $t = 30$ s in the simulation when genset 1 is active, having voltage output causality, and genset 2 is put in standby, also having voltage output causality but a speed of 600 rpm as shown in the plot. When the active power grid load is increased from 1.0 MW to 1.8 MW, starting at $t = 50$ s, genset 2 which is running in standby is commanded to start the synchronization process with the power grid by the MILP-based algorithm while genset 3 is started and put in standby. When the synchronization criteria are fulfilled, genset 2 is connected to the power grid

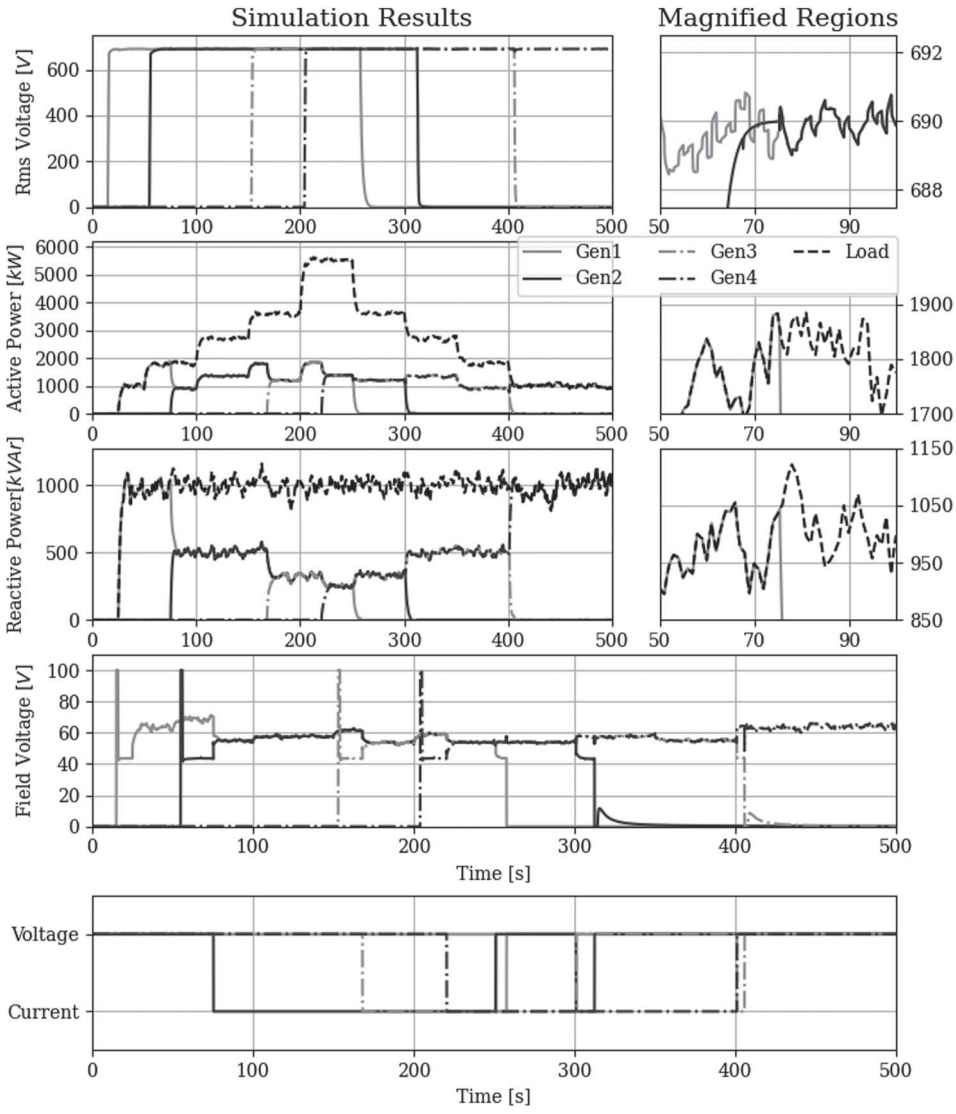

Figure 3. Voltages, active powers, reactive powers, field voltages and generator output causalities from the simulation.

and changes its causality to output currents and to get voltages in feedback. Following, when the active power grid load is increased to 3.6 MW genset 3 is commanded to start the synchronization process with the power grid, as genset 2, while genset 4 is put in standby, and when the active power grid load is increased to its maximal mean value of 5.5 MW in the simulation also the last genset is commanded to start the synchronization process with the power grid and connected when the connection criteria are fulfilled, also changing its causality to output currents. At this point, all gensets are active and there is no genset running in standby. Note that the time each genset uses to synchronize and connect to the power grid is listed in Table 4.

At $t = 250$ s the active power grid load is reduced back to 3.6 MW and genset 1 is commanded by the MILP-based algorithm to start the de-synchronization process, transferring its active and reactive load to the other active gensets. Also, the PMS algorithm tells genset 2 to take the lead, providing the power grid voltages. When the active and reactive power grid loads are transferred to the other active gensets, genset 1 is disconnected from the power grid and put in standby. The same goes for genset 2 and 3 when the active power grid load is further decreased – the next active generator in line takes the lead, and the genset with the lowest runtime is put in standby while the others are shut down. Note that the MILP-based algorithm takes out the active genset with the largest runtime from the power grid when the load is reduced. In this case, genset 2 has the lowest runtime and is kept in standby for the rest of the simulation, as shown in the simulation results.

Table 4. Synchronization time for the gensets in the power plant.

Genset	Approx. simulation time	Synchronization time
2	75 s	20.41 s
3	168 s	15.10 s
4	220 s	16.50 s

The right-most plot in the first row in the figure shows a magnified region of the genset speeds for genset 1 and 2 when the synchronization- and load sharing procedures are activated. When genset 2 is being synchronized it has a slightly lower speed than genset 1 to match the phase, and when the synchronization criteria are fulfilled the speed is ramped back up to match the power grid frequency. Afterwards, the load sharing process is activated and the two gensets get opposite speed peaks with absolute power grid frequency magnitudes of about 0.17 Hz due to the active load sharing

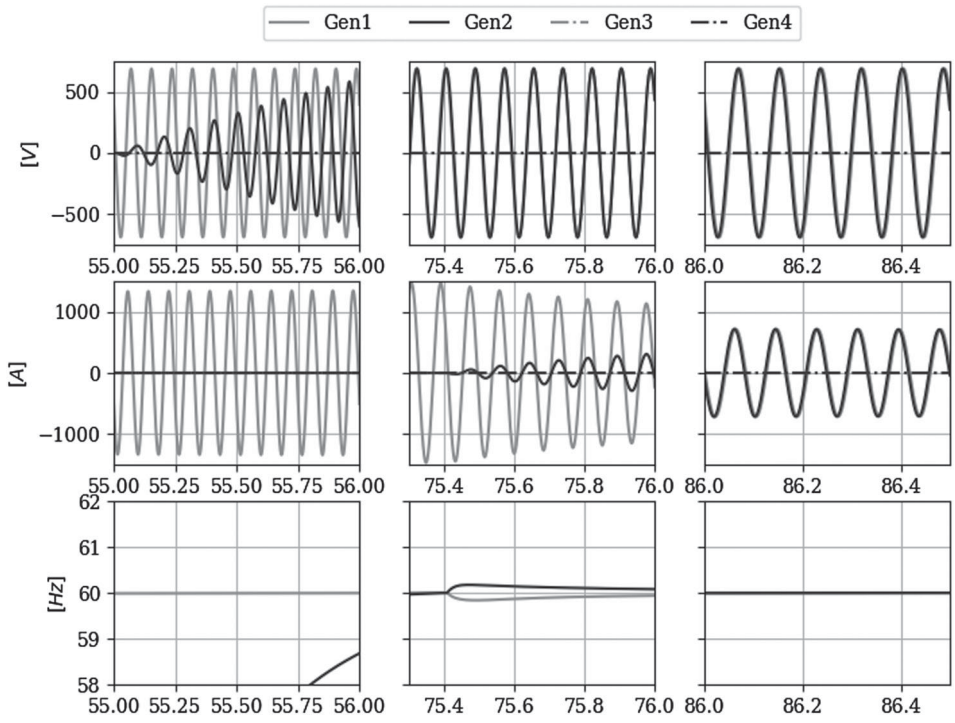

Figure 4. The phase voltage u_a and the corresponding current, i_a, given in the *a*, *b*, *c* reference frame.

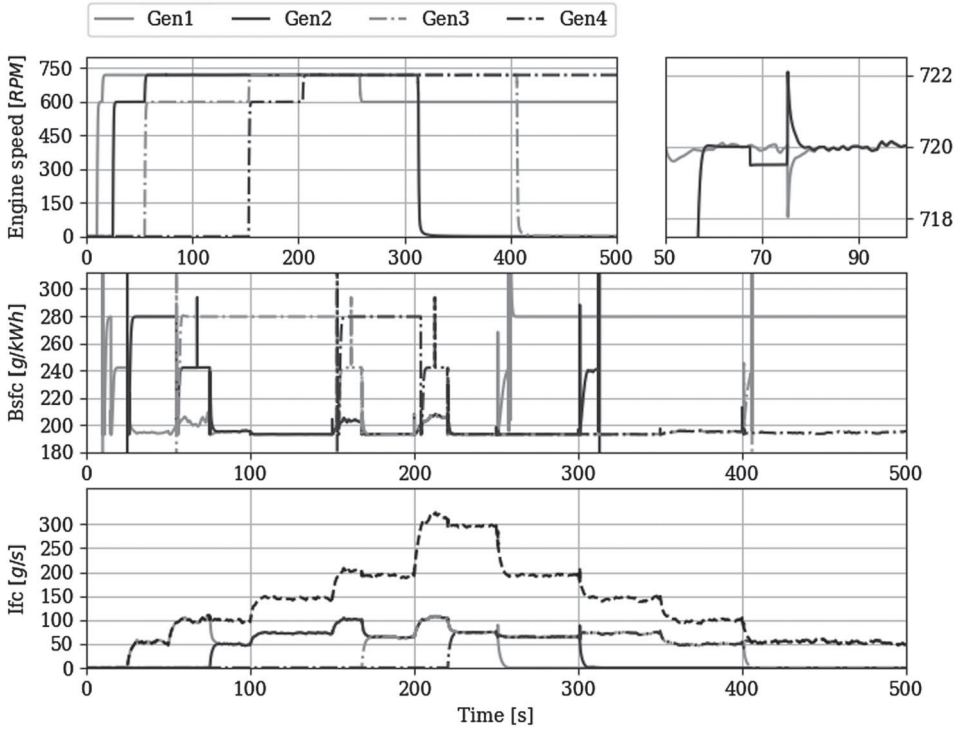

Figure 5. Auxiliary engine speeds, brake specific fuel consumptions and instantaneous fuel consumptions from the simulation.

control process initiated by the two governors controlling the auxiliary engines driving the generators. The plot in the second row in the figure shows the brake specific fuel consumption for each genset in the power plant, which is calculated from $b_e(P_e)$ in (19). As the results illustrate, a genset running in standby has a higher brake specific fuel consumption than when being connected to the power grid in production mode. However, a genset running in standby has a lower instantaneous fuel consumption when running in standby than when being connected to the power grid in production mode, as illustrated in the last plot in the figure. Note that the last plot also shows the total instantaneous fuel consumption for the entire power plant represented by the black dashed line.

6. Conclusion

This article set out to present an object-oriented, generic and modular simulator framework involving distributed co-simulation technology for simulating marine power plants with weak power grids. The generators in the power plant simulator are modelled as hybrid causality models in order to enable altering model interfaces – having either voltages as outputs and currents as inputs, or vice versa. This is crucial since the power grid is considered weak, having negligible capacitive effects, resulting in the fact that one of the active generators must provide the voltages and where the others contribute with currents, mathematically speaking, if event-based operations such as starting and stopping arbitrary gensets are aimed for. For generator scheduling an MILP-based algorithm is proposed which starts and stops arbitrary gensets based on a predefined cost functions and required overhead in power production.

The gensets, the power grid load, the AVRs and the governors are here implemented as separate object-oriented models such that the size of the power plant, the number of gensets, can be set by a single parameter, also automatically scaling the PMS. A case study involving four equally sized gensets was conducted to test the proposed simulator framework. For simplification reasons, the case study was implemented in Python and simulated using only one thread. This means that all calculations are done in series, which slows down the real-time capabilities. However, the simulator were able to run in real-time when the code was compiled using Cython.

In future work, each power plant component should be implemented as a stand-alone co-simulation FMU[1], such that the power plant model can be assembled in a full-system simulator of a marine vessel for various applications, enabling a larger scope of study by facilitating additional system components, such as the vessel's hull, more sophisticated power consumers such as deck machinery and propulsion systems, and stochastic environmental conditions such as waves, wind and current. Moreover, the PMS and the power grid calculations can be moved from the master algorithm to a separate FMU, or a *function unit* (Sadjina et al. 2018), which makes the use of standard co-simulation master algorithm possible. However, details regarding this is considered out of scope here. Nevertheless, the results from the case study illustrated that the proposed framework seems to be stable and suited for its purpose.

The proposed simulator framework also provides a good foundation for further research related to marine power plants, small wind farms or an isolated number of islands with power generators, for e.g. testing configurations with different genset sizes, development of better and more sophisticated control systems and PMS algorithms, in addition to being a virtual prototyping tool for development purposes. Also, it is possible to split the gensets into generators and auxiliary engine models for increasing the simulator modularity as well as replacing the power grid load with sub-simulators containing power grid consumers such as thrusters and propulsion systems for marine vessel applications, being themselves affected by stochastic load profiles, and where the PMS is extended to also include load reduction functionalities. However, this is considered out of scope in this article.

Note

1. The FMI standard is supported by over 100 modelling and simulation software and tools.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

The authors would like to thank the partners in the *ViProMa* project, Research Counsel of Norway (Norges Forskningsråd) project number 225322, and the collaboration project *SEACo*, Research Counsel of Norway (Norges Forskningsråd) project number 326710, for financial support.

Notes on contributors

Stian Skjong received the M.Sc. degree in 2014 and the PhD degree in 2017, both in marine technology from the Norwegian University of Science and Technology (NTNU). He has been working as a Senior Research Scientist in SINTEF Ocean since 2018. His research interests include modeling, simulation and control of marine systems, nonlinear control, model based control designs, co-simulation strategies for virtual prototyping of marine systems and operations, and simulation software development. He has been working with co-simulations, both modelling, numerical and dynamical stability of models and simulators, co-simulation algorithms and interface standardization since 2014, which is also the topics of his Ph.D. thesis.

Eilif Pedersen received the M.Sc. degree in Marine Engineering from the Norwegian Institute of Technology, Norway, in 1983. He has been with the Norwegian Marine Technology Research Institute, as a Senior Research Engineer until 1999, where he joined the NTNU as an Associate Professor. He has held multiple positions, such as a Vice Dean of Education with the Faculty of Engineering Science and Technology, the Head of Master Programs in marine technology, the Leader with the Research Group of Marine Systems, and the Head of Machinery Laboratory at the Department of Marine Technology. His areas of expertise are in the field of modeling methodology and simulation of dynamic multidisciplinary and mechatronic systems focusing on machinery system dynamics, internal combustion engines, vibrations, thermal- and hydraulic machines, fuel cell system dynamics, and hybrid power plants for marine applications.

ORCID

Stian Skjong http://orcid.org/0000-0002-1953-9589
Eilif Pedersen http://orcid.org/0000-0001-9778-6422

References

Abel A, Blochwitz T, Eichberger A, Hamann P, Rein U. 2012. Functional mockup interface in mechatronic gearshift simulation for commercial vehicles. 9th International Modelica Conference; Munich, Germany. p. 775–780.

Blochwitz T, Otter M, Arnold M, Bausch C, Clauss C, Elmqvist H, Junghanns A, Mauss J, Monteiro M, Neidhold T, et al. 2011. The functional mockup interface for tool independent exchange of simulation models. Proceedings of the 8th International Modelica Conference. Dresden, Germany.

Bø TI, Johansen TA, Dahl AR, Miyazaki MR, Pedersen E, Rokseth B, Skjetne R, Sørensen AJ, Thorat L, Utne IB. 2015. Real-time marine vessel and power plant simulation. ASME 2015 34th International Conference on Ocean, Offshore and Arctic Engineering. American Society of Mechanical Engineers. St. John's, NL, Canada. p. V001T01A007–V001T01A007.

Bouman EA, Lindstad E, Rialland AI, Strømman AH. 2017. State-of-the-art technologies, measures, and potential for reducing GHG emissions from shipping – A review. Transp Res D Transp Environ. 52(Part A):408–421.

DNV. 2020. www.opensimulationplatform.com. Visited 09.03.2022. (webpage)

Edström K. 1999. Switched bond graphs: simulation and analysis [PhD thesis]. Linköping University, Automatic Control.

Funabiki S, Hitsumoto A, Yamakawa Y, Ito T. 1991. Automatic voltage regulation of synchronous generator with pole assignment self-tuning regulator. Proceedings of the International Conference on Industrial Electronics, Control and Instrumentation. Kobe, Japan. 1991. IECON '91; Vol. 3. p. 1807–1811.

Goebel R, Sanfelice RG. 2012. Hybrid dynamical systems: modeling, stability, and robustness. Princeton University Press.

Gomes C, Thule C, Broman D, Larsen PG, Vangheluwe H. 2017. Co-simulation: state of the art. *CoRR*, abs/1702.0.

Han Y, Li H, Shen P, Coelho EAA, Guerrero JM. 2017. Review of active and reactive power sharing strategies in hierarchical controlled microgrids. IEEE Trans Power Electron. 32(3):2427–2451.

Heywood JB. 1988. Internal combustion engine fundamentals. McGraw-Hill. (Automotive technology series).

Karnopp DC, Margolis DL, Rosenberg RC. 2006. System dynamics: modeling and simulation of mechatronic systems. New York (NY): John Wiley & Sons, Inc.

Krammer M, Martin H, Radmilovic Z, Erker S, Karner M. 2015. Standard compliant co-simulation models for verification of automotive embedded systems. Forum on Specification and Design Languages; Barcelona, Spain. Vol. 2015-Octob.

Louis JP. 2013. Control of synchronous motors. ISTE. Wiley.
Machowski J, Bialek JW, Bumby JR. 2008. Power system dynamics: stability and control. 2nd ed.Chichester, UK: Wiley.
Makhorin AMAI. 2012. GLPK – GNU Project – Free Software Foundation (FSF). (https://www.gnu.org/software/glpk/). Visited 09.03.2022.
Marwali MN, Keyhani A. 2004. Control of distributed generation systems-Part I: voltages and currents control. IEEE Trans Power Electron. 19(6):1541–1550.
Pedersen TA. 2009. Bond graph modeling of marine power systems [PhD thesis]. Norwegian University of Science and Technology, Department of Marine Technology.
Pedersen TA, Pedersen E. 2012. Bond graph modelling of marine power systems. Math Comput Model Dyn Syst. 18(2):153–173.
Sadjina SS, Kyllingstad L, Rindarøy M, Skjong S, Æsøy V, Pedersen E. 2018. Distributed co-simulation of maritime systems and operations. J Offshore Mech Arct Eng. 141(1). https://doi.org/10.1115/1.4040473.
Sahm D. 1979. A two-axis, bond graph model of the dynamics of synchronous electrical machines. J Franklin Inst. 308(3):205–218.
Skjong E, Johansen TA, Molinas M, Sørensen AJ. 2017a. Approaches to economic energy management in diesel–electric marine vessels. IEEE Trans Transp Electr. 3(1):22–35.
Skjong S. 2017. Modeling and simulation of maritime systems and operations for virtual prototyping using co-simulations [PhD thesis]. Trondheim: Norwegian University of Science and Technology.
Skjong S, Pedersen E. 2016. The theory of bond graphs in distributed systems and simulations. 2016 International Conference on Bond Graph Modeling and Simulation (ICBGM 2016); Montreal, Canada. Society for Modeling & Simulation International.
Skjong S, Pedersen E. 2017a. A real-time simulator framework for marine power plants with weak power grids. Mechatronics. 47:24–36.
Skjong S, Pedersen E. 2017b. Co-simulation of a marine offshore vessel in DP-operations including Hardware-In-the-Loop (HIL). ASME 2017 36th International Conference on Ocean, Offshore and Arctic Engineering (OMAE2017); Trondheim, Norway.
Skjong S, Pedersen E. 2017c. Non-angular MPC-based thrust allocation algorithm for marine vessels – A study of optimal thruster commands. IEEE Trans Transp Electr. 3(3):792–807.
Skjong S, Rindarøy M, Kyllingstad L, Æsøy V, Pedersen E. 2017b. Virtual prototyping of maritime systems and operations: applications of distributed co-simulations. J Mar Sci Technol. 1–19.
Sogandares FM. 2002. Stone axes and warhammers: a decade of distributed simulation in aviation research. Proceedings – 16th Workshop on Parallel and Distributed Simulation, PADS; Washington, D.C. 2002.
Steghöfer J-P, Anders G, Siefert F, Reif W. 2013. A system of systems approach to the evolutionary transformation of power management systems. In: Horbach M, editor. INFORMATIK 2013 – Informatik angepasst an Mensch, Organisation und Umwelt. Bonn: Gesellschaft für Informatik e.V.; p. 1500–1515.
Winter M, Taube J, Froeschl J, Herzog H-G. 2015. From simulation to testbench using the FMI-standard. 2015 IEEE Vehicle Power and Propulsion Conference; Montreal, QC, Canada. VPPC 2015 – Proceedings.
Yum KK, Pedersen E. 2016. Architecture of model libraries for modelling turbocharged diesel engines. Math Comput Model Dyn Syst. 22(6):584–612.
Yum KK, Skjong S, Tasker B, Pedersen E, Steen S. 2016. Simulation of a hybrid marine propulsion system in waves. Proceedings of the 28th CIMAC World Congress. Helsinki: CIMAC.