Seismic Hybrid Testing using FMI-based Co-Simulation

Cláudio Gomes[1] Giuseppe Abbiati[2] Peter Gorm Larsen[1]

[1]Department of Electrical and Computer Engineering, Aarhus University, Denmark,
{claudio.gomes,pgl}@ece.au.dk
[1]Department of Civil and Architectural Engineering, Aarhus University, Denmark, abbiati@cae.au.dk

Abstract

Hybrid testing is an experimental technique extensively utilized in earthquake engineering to study the seismic response of structures. It requires coupling physical and numerical models in a closed feedback loop. Although this methodology is mature, a commonly accepted standard for orchestrating simulations and experiments is still missing. As a result, setting up a hybrid testing campaign still requires substantial system integration effort, which is often not affordable. In this paper, we propose the Functional Mockup Interface as a possible standard for orchestrating hybrid testing and discuss the limitations in enabling such support.

Keywords: earthquake engineering, hybrid testing, functional mockup interface, co-simulation, model exchange, master algorithm

1 Introduction

Modern civil engineering structures are designed to damage in a controlled manner when exposed to extreme seismic events. In principle, load-resisting systems are such that secondary structural elements behave as fuses that: i) cut-off loading introduced in primary structural elements; and ii) dissipate energy during seismic motion. As a result, the equivalent dynamic amplification factor of the structure is reduced and primary structural elements are preserved from damage. It comes with no surprise that understanding the dynamic response of structures in the inelastic regime is of central importance in earthquake engineering (Chopra 2012).

In order to develop construction technologies and design codes for seismic-resistant structures, a great deal of effort has been allocated to enable cost-effective experimentation in the last fifty years. Hybrid testing (HT), originally introduced in Japan, has rapidly replaced expensive shake table testing in response to this need (Nakashima 2020). HT is performed using hybrid physical-numerical models instead of purely physical models. A reference structural prototype is partitioned into a Physical Substructure (PS) and a Numerical Substructure (NS). The PS is tested in the laboratory by means of servo-controlled actuators whereas the NS is instantiated in a numerical simulation environment. A simulation algorithm solves the coupled equations of motion of the hybrid model, which is subjected to a realistic loading history, and updates the boundary conditions of PS and NS *on the fly* (Pan, Wang and Nakashima 2016). The PS is the focus of the experimental campaign since it comprises those structural components or sub-assemblies which lack of predictive numerical models. Conversely, the NS comprises all those parts that can be reliably replaced by numerical models (e.g., masses or components that experience a linear response regime).

The main advantage compared to shake table testing is that HT can be performed in *pseudodynamic* regime, that is, with an extended time scale, when the PS has a rate-independent restoring force (i.e., acts like a idealized spring). This assumption holds with a reasonable approximation for steel, concrete, and masonry structures. Noteworthy, in pseudodynamic HT, both inertia and damping forces of the PS are simulated numerically. Pseudodynamic HT enables full-scale experimentation with small hydraulic power. Typical testing time scales are $50-200$ times slower than wall-clock time. Real-time HT indicates the limit case of 1:1 time scale. If not specified, HT is assumed to be conducted in the pseudodynamic regime, which covers the vast majority of application cases. The paper of McCrum and Williams (2016) provides the most up-to-date review of the seismic HT methodology.

Seismic HT developed into a self-standing research area with relatively low cross-fertilization with other fields of civil engineering. HT methodologies developed in offshore (Sauder et al. 2016), fire (Sauca et al. 2021) and geotechnical engineering (Idinyang et al. 2019) are quite similar in form and maturity. However, for all cases, a standardized approach to coupling simulation and testing environments is still missing. As a result, setting up HT requires a lot of system integration effort, which is often not affordable.

Co-simulation, which is a technique to simulate a coupled system via combination of multiple black-box simulation units, shall provide a solution to this issue. The survey Gomes, Thule, Broman et al. (2018) gives a broad overview of co-simulation, spanning heterogeneous application domains like multi-body dynamics (Kübler and Schiehlen 2000) and large-scale circuit simulation (Lelarasmee, Ruehli and A. L. Sangiovanni-Vincentelli 1982; Newton and Alberto L. Sangiovanni-Vincentelli 1983). Specifically, simulation units, often developed and exported independently from each other in different Modelling & Simulation (M&S) tools, are coupled using an or-

chestration algorithm. The definition of simulation unit is not constrained to software artefacts, and therefore physical subsystems can be connected to virtual ones. From this perspective, HT is clearly a specific instance of co-simulation.

Contribution. To the best of our knowledge, no standard has emerged to enable seismic HT. As such, in this paper, we propose and evaluate the use of the FMI standard, version 2.1, to the co-simulation of a representative HS experimental setup, illustrated in Figure 1. We adapt the numerical algorithm proposed in Giuseppe Abbiati, La Salandra et al. (2018) to the FMI interface, with variants for Model Exchange and Co-simulation.

Structure. In order to demonstrate the use of FMI-based Co-simulation for seismic HT, an application example is described in Section 2. The specific co-simulation layout for the presented example is discussed in Section 3. Conclusions and future perspectives are given in Section 4.

2 Background and Case Study

This section provides background information on HT along with the definition of the experimental setup, which is used to demonstrate the application of FMI-based co-simulation. Such a case study aims at representing the class of structural systems typically investigated via seismic HT. Both hybrid model and simulation algorithm are accurately described.

2.1 Hybrid Model

The selected case study consists of a split-mass single-degree-of-freedom (S-DoF) system where both NS and PS are linear. Prototype structure, hybrid model and experimental setup, originally presented in Martin Hovmand, Giuseppe Abbiati and Vabbersgaard Andersen (2021), are illustrated in Figure 1. As can be appreciated, the PS consists of a cantilever beam with a tip mass whereas the NS is a S-DoF spring-mass-dashpot oscillator. An electric linear actuator controls the tip displacement of the cantilever beam, which corresponds the single DoF of the hybrid model. A force transducer measures the corresponding restoring force. The actuator is characterized by 300 mm stroke and 10 kN force capacity; the latter coincides with the force transducer admissible load. Both actuator controller and force transducer are connected to an industrial PC via EtherCAT. The industrial PC hosts a Python environment from which one can set the actuator position and read the corresponding restoring force using the control system API. The HT setup is installed at the Dynamisk LAB of Aarhus University. Since the NS of the selected hybrid model is defined by an analytical model, in our implementation, the same Python environment hosts both substructure models, simulation algorithm and the interface to the control system. In order to host more complex NSs, simulation algorithms are usually interfaced to an external simulation environment, typically based on the FE method. To this end, a number of middleware tools have

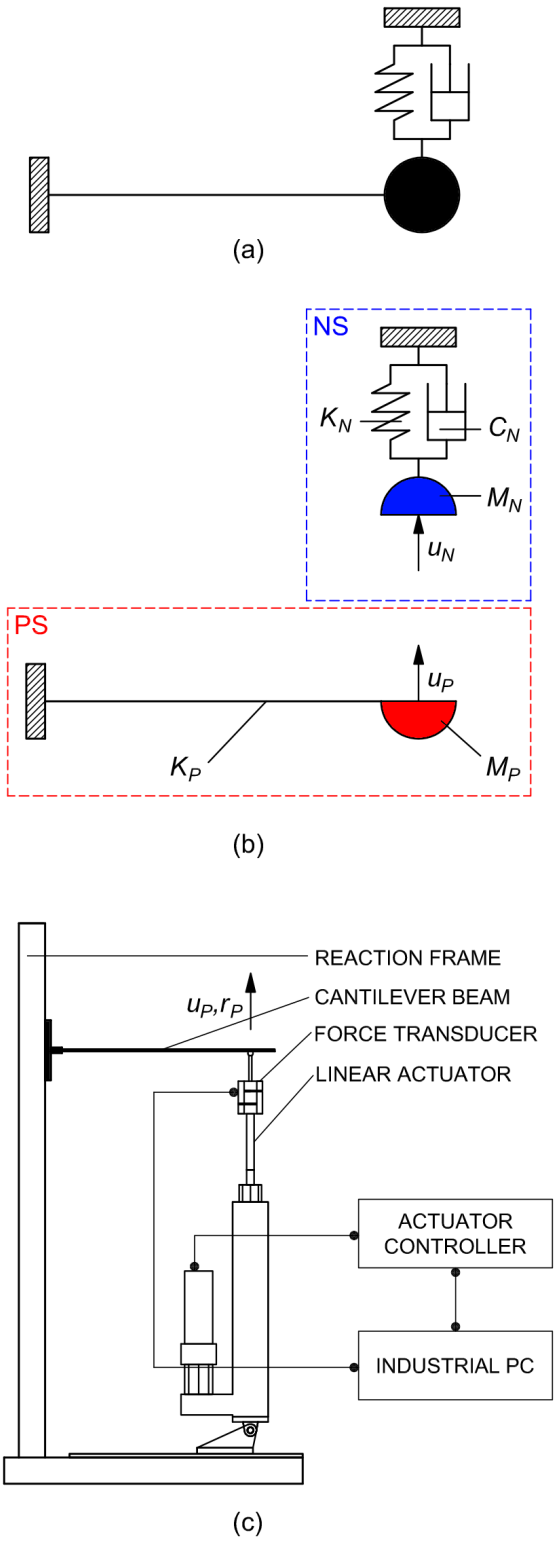

Figure 1. Reference case study adapted from M. Hovmand, G. Abbiati and Andersen (2021): a) prototype structure; b) hybrid model; c) experimental setup installed at the Dynamisk LAB of Aarhus University.

been developed. Among all, OpenFresco, developed at the University of California, Berkeley, is commonly used (McKenna, Scott and Gregory L Fenves 2010). OpenFresco can interface with industrial PCs and FE frameworks. OpenFresco is typically used in combination with OpenSees (Schellenberg, Mahin and Gregory L. Fenves 2007), which is a FE software specifically developed for analyzing steel and concrete structures subjected to earthquake loading. HT with geographically distributed PS and NS using OpenSees/OpenFresco has been demonstrated in Stojadinovic, Mosqueda and Mahin (2006). However, linking new simulation environments requires coding customized interfaces. Our contribution aims at eliminating this effort by promoting the use of FMI interface for HT.

2.2 Simulation Algorithm

In line with the philosophy of FMI-based Co-simulation, which treats the coupled simulation as a combination of independent simulation units, we selected a simulation algorithm based on partitioned time integration originally proposed in Giuseppe Abbiati, La Salandra et al. (2018) and lately adapted to hybrid fire testing in Giuseppe Abbiati, Covi et al. (2020). In a partitioned setting, Lagrange multipliers enforce compatibility conditions among substructures (i.e., simulation units) and the simulation time step is solved with a two-stage algorithm. First, substructure responses are evaluated as if they were uncoupled. Then, Lagrange multipliers are computed by solving a linearized interface problem. The original idea of using partitioned time integration to perform HT was proposed by Pegon and Magonette (2002). The first large-scale seismic testing campaign based on the scheme is described in Giuseppe Abbiati, Oreste S. Bursi et al. (2015) and Oreste S. Bursi et al. (2017).

The partitioned time integration scheme proposed in Giuseppe Abbiati, La Salandra et al. (2018) is presented to compute the seismic response of a generic hybrid model with one PS and one NS. The following system of differential-algebraic equations describes the motion of the hybrid model,

$$\begin{cases} \begin{cases} \mathbf{M}^N\ddot{\mathbf{u}}^N + \mathbf{C}^N\dot{\mathbf{u}}^N + \mathbf{r}^N\left(\mathbf{u}^N\right) = \mathbf{f}^N + \mathbf{L}^{N^T}\boldsymbol{\Lambda}^N \\ \mathbf{L}^N\dot{\mathbf{u}}^N + \bar{\mathbf{L}}^N\dot{\mathbf{u}}^g = \mathbf{0} \end{cases} \\ \begin{cases} \mathbf{M}^P\ddot{\mathbf{u}}^P + \mathbf{C}^P\dot{\mathbf{u}}^P + \mathbf{r}^P\left(\mathbf{u}^P\right) = \mathbf{f}^P + \mathbf{L}^{P^T}\boldsymbol{\Lambda}^P \\ \mathbf{L}^P\dot{\mathbf{u}}^P + \bar{\mathbf{L}}^P\dot{\mathbf{u}}^g = \mathbf{0} \end{cases} \\ \bar{\mathbf{L}}^{N^T}\boldsymbol{\Lambda}^N + \bar{\mathbf{L}}^{P^T}\boldsymbol{\Lambda}^P = \mathbf{0} \end{cases} \tag{1}$$

In detail, $\mathbf{u}^{(\bullet)}$, $\dot{\mathbf{u}}^{(\bullet)}$ and $\ddot{\mathbf{u}}^{(\bullet)}$ are displacement, velocity and acceleration vectors, respectively. The terms $\mathbf{r}^{(\bullet)}$ are rate-independent restoring force vectors. $\mathbf{M}^{(\bullet)}$ and $\mathbf{C}^{(\bullet)}$ are mass and damping matrices, respectively. Seismic loading is defined as $\mathbf{f}^{(\bullet)} = \mathbf{M}^{(\bullet)}\mathbf{t}^{(\bullet)}a(t)$ where $\mathbf{t}^{(\bullet)}$ are Boolean vectors that project seismic acceleration history $a(t)$ on system DoFs. Matrices $\mathbf{L}^{(\bullet)}$ and $\bar{\mathbf{L}}^{(\bullet)}$ localize the shared DoFs on each substructure DoF vector and the vector of generalized interface velocities $\dot{\mathbf{u}}_g$, respectively. The latter gathers all DoFs shared among substructures. The entries of $\mathbf{L}^{(\bullet)}$ are 0 and 1 whereas the entries of $\bar{\mathbf{L}}^{(\bullet)}$ are 0 and -1. According to (1), Lagrange multiplier vectors $\boldsymbol{\Lambda}^{(\bullet)}$ enforce velocity compatibility with $\dot{\mathbf{u}}_g$. It is important to stress that all Lagrange multiplier vectors form a set of self-balanced forces to ensure interface equilibrium *a priori*. The solution of (1) enforces kinematic compatibility *a posteriori*. The HT algorithm is presented to integrate (1) from time t_k to t_{k+1} with a time step Δt.

1. Solve the NS *free* problem at t_{k+1},

$$\begin{cases} \tilde{\mathbf{u}}_{k+1}^{N,free} = \mathbf{u}_k^N + \dot{\mathbf{u}}_k^N\Delta t + \left(\frac{1}{2} - \beta^N\right)\Delta t^2\ddot{\mathbf{u}}_k^N \\ \tilde{\dot{\mathbf{u}}}_{k+1}^{N,free} = \dot{\mathbf{u}}_k^N + \left(1-\gamma^N\right)\Delta t\ddot{\mathbf{u}}_k^N \end{cases} \tag{2}$$

$$\ddot{\mathbf{u}}_{k+1}^{N,free} = \mathbf{D}^{N^{-1}}\left[\mathbf{f}_{k+1}^N - \mathbf{C}^N\tilde{\dot{\mathbf{u}}}_{k+1}^{N,free} - \mathbf{r}_{k+1}^N\left(\tilde{\mathbf{u}}_{k+1}^{N,free}\right)\right] \tag{3}$$

$$\begin{cases} \mathbf{u}_{k+1}^{N,free} = \tilde{\mathbf{u}}_{k+1}^{N,free} + \ddot{\mathbf{u}}_{k+1}^{N,free}\beta^N\Delta t^2 \\ \dot{\mathbf{u}}_{k+1}^{N,free} = \tilde{\dot{\mathbf{u}}}_{k+1}^{N,free} + \ddot{\mathbf{u}}_{k+1}^{N,free}\gamma^N\Delta t \end{cases} \tag{4}$$

with,

$$\mathbf{D}^N = \mathbf{M}^N + \mathbf{C}^N\gamma^N\Delta t + \mathbf{K}^N\beta^N\Delta t^2 \tag{5}$$

where γ^N and β^N are the parameters of the Newmark scheme for the NS (Bathe 1982) and $\mathbf{K}^N$ is the stiffness matrix. In Equation (3), the displacement predictor $\tilde{\mathbf{u}}_{k+1}^{N,free}$ is sent to an external FE software that computes the corresponding restoring force $\mathbf{r}_{k+1}^N$.

2. Solve the PS *free* problem at t_{k+1},

$$\begin{cases} \tilde{\mathbf{u}}_{k+1}^{P,free} = \mathbf{u}_k^P + \dot{\mathbf{u}}_k^P\Delta t + \left(\frac{1}{2} - \beta^P\right)\Delta t^2\ddot{\mathbf{u}}_k^P \\ \tilde{\dot{\mathbf{u}}}_{k+1}^{P,free} = \dot{\mathbf{u}}_k^P + \left(1-\gamma^P\right)\Delta t\ddot{\mathbf{u}}_k^P \end{cases} \tag{6}$$

$$\ddot{\mathbf{u}}_{k+1}^{P,free} = \mathbf{D}^{P^{-1}}\left[\mathbf{f}_{k+1}^P - \mathbf{C}^P\tilde{\dot{\mathbf{u}}}_{k+1}^{P,free} - \mathbf{r}_{k+1}^P\left(\tilde{\mathbf{u}}_{k+1}^{P,free}\right)\right] \tag{7}$$

$$\begin{cases} \mathbf{u}_{k+1}^{P,free} = \tilde{\mathbf{u}}_{k+1}^{P,free} + \ddot{\mathbf{u}}_{k+1}^{P,free}\beta^P\Delta t^2 \\ \dot{\mathbf{u}}_{k+1}^{P,free} = \tilde{\dot{\mathbf{u}}}_{k+1}^{P,free} + \ddot{\mathbf{u}}_{k+1}^{P,free}\gamma^P\Delta t \end{cases} \tag{8}$$

with,

$$\mathbf{D}^P = \mathbf{M}^P + \mathbf{C}^P\gamma^P\Delta t + \mathbf{K}^P\beta^P\Delta t^2 \tag{9}$$

where γ^P and β^P are the parameters of the Newmark scheme for the PS (Bathe 1982). In Equation (7), the displacement predictor $\tilde{\mathbf{u}}_{k+1}^{P,free}$ is imposed to the PS by means of servo-controlled actuators and the corresponding restoring force vector $\mathbf{r}_{k+1}^P$ is measured using force transducers. Noteworthy displacement control errors affects the measured restoring

force $\mathbf{r}^{P,free}_{k+1,mes}$ and may bias the emulated system response. Accordingly, control and measurement errors are compensated as suggested by Oreste S Bursi O. S. and Shing (1996),

$$\mathbf{r}^{P,free}_{k+1} = \mathbf{r}^{P,free}_{k+1,mes} + \mathbf{K}^P(\mathbf{u}^{P,free}_{k+1} - \mathbf{u}^{P,free}_{k+1,mes}) \quad (10)$$

where $\mathbf{u}^{P,free}_{k+1,mes}$ and $\mathbf{r}^{P,free}_{k+1,mes}$ are measured displacement and restoring force vectors. The stiffness matrix of the PS $\mathbf{K}^P$ is estimated before HT based on low-amplitude cyclic tests.

3. Solve the linearized interface problem a t_{k+1},

$$\begin{bmatrix} \mathbf{\Lambda}^N_{k+1} \\ \mathbf{\Lambda}^P_{k+1} \\ \dot{\mathbf{u}}^g_{k+1} \end{bmatrix} = -\mathbf{G}^{-1} \begin{bmatrix} \mathbf{L}^N \dot{\mathbf{u}}^{N,free}_{k+1} \\ \mathbf{L}^P \dot{\mathbf{u}}^{P,free}_{k+1} \\ \mathbf{0} \end{bmatrix} \quad (11)$$

with,

$$\mathbf{G} = \begin{bmatrix} \mathbf{L}^N \mathbf{D}^{N^{-1}} \mathbf{L}^{N^T} \gamma^N \Delta t & \mathbf{0} & \bar{\mathbf{L}}^N \\ \mathbf{0} & \mathbf{L}^P \mathbf{D}^{P^{-1}} \mathbf{L}^{P^T} \gamma^P \Delta t & \bar{\mathbf{L}}^P \\ \bar{\mathbf{L}}^{N^T} & \bar{\mathbf{L}}^{P^T} & \mathbf{0} \end{bmatrix} \quad (12)$$

4. Calculate *link* kinematic quantities at t_{k+1},

$$\begin{cases} \ddot{\mathbf{u}}^{N,link}_{k+1} = \mathbf{D}^{N^{-1}} \mathbf{L}^{N^T} \mathbf{\Lambda}^N_{k+1} \\ \dot{\mathbf{u}}^{N,link}_{k+1} = \mathbf{D}^{N^{-1}} \mathbf{L}^{N^T} \mathbf{\Lambda}^N_{k+1} \gamma^N \Delta t \\ \mathbf{u}^{N,link}_{k+1} = \mathbf{D}^{N^{-1}} \mathbf{L}^{N^T} \mathbf{\Lambda}^N_{k+1} \beta^N \Delta t^2 \end{cases} \quad (13)$$

$$\begin{cases} \ddot{\mathbf{u}}^{P,link}_{k+1} = \mathbf{D}^{P^{-1}} \mathbf{L}^{P^T} \mathbf{\Lambda}^P_{k+1} \\ \dot{\mathbf{u}}^{P,link}_{k+1} = \mathbf{D}^{P^{-1}} \mathbf{L}^{P^T} \mathbf{\Lambda}^P_{k+1} \gamma^P \Delta t \\ \mathbf{u}^{P,link}_{k+1} = \mathbf{D}^{P^{-1}} \mathbf{L}^{P^T} \mathbf{\Lambda}^P_{k+1} \beta^P \Delta t^2 \end{cases} \quad (14)$$

5. Calculate *coupled* kinematic quantities at t_{k+1},

$$\begin{cases} \ddot{\mathbf{u}}^N_{k+1} = \ddot{\mathbf{u}}^{N,free}_{k+1} + \ddot{\mathbf{u}}^{N,link}_{k+1} \\ \dot{\mathbf{u}}^N_{k+1} = \dot{\mathbf{u}}^{N,free}_{k+1} + \dot{\mathbf{u}}^{N,link}_{k+1} \\ \mathbf{u}^N_{k+1} = \mathbf{u}^{N,free}_{k+1} + \mathbf{u}^{N,link}_{k+1} \end{cases} \quad (15)$$

$$\begin{cases} \ddot{\mathbf{u}}^P_{k+1} = \ddot{\mathbf{u}}^{P,free}_{k+1} + \ddot{\mathbf{u}}^{P,link}_{k+1} \\ \dot{\mathbf{u}}^P_{k+1} = \dot{\mathbf{u}}^{P,free}_{k+1} + \dot{\mathbf{u}}^{P,link}_{k+1} \\ \mathbf{u}^P_{k+1} = \mathbf{u}^{P,free}_{k+1} + \mathbf{u}^{P,link}_{k+1} \end{cases} \quad (16)$$

The main advantage of partitioned time integration is that the PS *free* response can be solved with an *explicit* scheme, which does not require estimating the PS stiffness $\mathbf{K}^P$, while the NS *free* response is solved with an *implicit* scheme. Compatibility of interface velocities stated in (1) ensures that the coupled simulation is stable as long as each time integration scheme is stable as uncoupled (Gravouil and Combescure 2001). Accordingly, in our implementation, the central difference scheme is utilized on the PS ($\gamma^P = \frac{1}{2}, \beta^P = 0$) whereas the mid-point rule on the NS ($\gamma^N = \frac{1}{2}, \beta^N = \frac{1}{4}$) (Bathe 1982). Usually, the PS is characterized by very few DoFs ($1-10$) and relatively low eigenfrequencies ($0-20$ Hz) so, typically, $\Delta t = 10$ msec. Considering a time scale $\lambda = 100$, the wall-clock time required to solve one time integration step is $\Delta T = \Delta t \lambda = 1$ sec, which is sufficiently large to accommodate actuation and filtering delays. In our example, $\mathbf{M}^N$ and $\mathbf{K}^N$ have constant scalar values and the NS *free* response is solved without Newton-Raphson iterations, following the operator-splitting approach (Oreste S Bursi O. S. and Shing 1996). Since, HT is performed with an extended time scale, $\mathbf{M}^P$ is estimated analytically. Following the procedure suggested in Molina et al. (2011), we set the PS damping matrix $\mathbf{C}^P$ to zero.

We stress that the Steklov-Poincaré operator $\mathbf{G}$ defined in (12) is computed and inverted once before HT. As a result, the solution of the interface problem (Step 3) and calculation of *link* kinematic quantities (Step 4) require only of a few matrix multiplications.

3 FMI-based Implementation

The FMI 2.1 defines two main interfaces: the Co-simulation (CS), and the Model Exchange (ME). In the FMI nomenclature, a simulation unit is called the Functional Mockup Unit (FMU), and it may implement one or two of the main interfaces. The FMU is a zip file containing: binaries and source code (optional) implementing the API functions; miscellaneous resources; and an XML file, describing the variables, model structure, and other data.

In this work, HT is numerically simulated. Therefore, both NS and PS FMUs are implemented as ODEs. In this regard, the difference between an the ME and the CS interfaces lies in the fact that the former enables the FMU to expose the ODE derivative function, whereas the CS always represents the time discretized sub-model, enabling the FMU to export the sequence of ODE states, as the simulation progresses in time. From the point of view of the orchestration algorithm, defined below, a FMU ME still needs to be coupled to a numerical solver, which will be responsible for querying the derivatives and updating the states of the FMU, whereas a FMU CS comes with its own numerical solver. The differences are illustrated in Figures 2 and 3.

A simulation scenario is a description of the FMUs and their inter-connections. In order to make the following discussion clearer, we define the following:

Importer – Denotes the application that loads the FMUs;

Orchestrator – Denotes the algorithm, executing on the Importer, that interacts with the FMUs, inspecting their outputs, setting their inputs, etc, according to the simulation scenario.

Coupling – Denotes the strategy that the orchestrator uses, in order to ensure that physical laws are respected.

While it is common to mix the Orchestrator and Coupling together as one concept, as we will show later, the

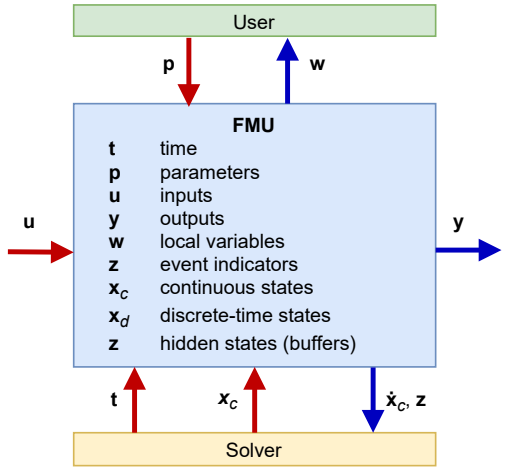

Figure 2. Schematic view of data flow between user, the solver of the importer and the FMU for Model Exchange. Based on (FMI v. 3.0 2021).

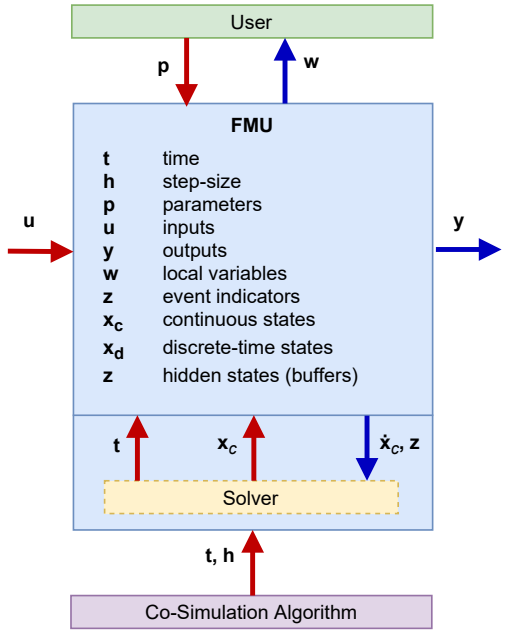

Figure 3. Schematic view of data flow between user, the solver of the importer and the FMU for Co-Simulation. Based on (FMI v. 3.0 2021).

distinction is useful because the Orchestrator is generic and can be applied to many different simulation scenarios (e.g., consider the Jacobi and Gauss-Seidel schemes (Kübler and Schiehlen 2000; Bastian et al. 2011)), but the Coupling is often specifically designed for a particular simulation scenario (e.g., the coupling can be implemented as a semantic adaptation (Gomes, Meyers et al. 2018), or automatically generated from hints (Gomes, Oakes et al. 2019; Oakes et al. 2021)).

3.1 FMI Co-simulation Interface

Figure 4 shows two possible simulation scenarios that implement the setup described in Figure 1 in a co-simulation with the coupling algorithm introduced in Section 2.2. In the figure, variants A and B differ only in the data that is communicated between the Coupling and the FMUs. In variant A, vectors $\dot{\mathbf{u}}^{\mathbf{N}}$ and $\dot{\mathbf{u}}^{\mathbf{P}}$ are copied, whereas in variant B, the (typically) smaller vectors $\mathbf{L}^{N}\dot{\mathbf{u}}^{\mathbf{N}}$ and $\mathbf{L}^{P}\dot{\mathbf{u}}^{\mathbf{P}}$ are passed. Each variant requires therefore, a slightly different FMU implementation. In the figure, variant B only shows the differences with respect to variant A for simplicity.

In both variants, the *NSFMU* and *PSFMU* communicate with the FE software and Test Setup, respectively. These FMUs have no input-to-output algebraic dependencies, and all of their computations can happen inside the `fmi2DoStep` function, making them well suited to be implemented as co-simulation FMUs.

It is important to remark that the application of the FMI CS interface implies that the FMUs are compatible with the coupling algorithm. In particular, recall that the coupling algorithm requires a correction to the states of the FMUs, in Equations (15) and (16). As such, the FMUs must support a mechanism to perform this correction. The authors of (Brembeck et al. 2014) also report on the same difficulties. The implementation of *Coupling* can too be done in multiple ways, described next.

Coupling as Custom Orchestrator. The most common approach is to implement, or automatically generate, the coupling algorithm directly as part of the orchestration algorithm. This works well when the user has full control over the orchestration algorithm, or when the co-simulation framework, or modelling environment, being used, provides the ability for customization (e.g., using a co-simulation library like PyFMI (Andersson, Åkesson and Führer 2016), FMPy[1], among others).

Coupling as FMU. For situations where the user has little programming expertize, or no control over the co-simulation framework, it is easier to implement the coupling as an FMU, which is then loaded during the co-simulation, as any other FMU. Frameworks such as the one described in (Gomes, Meyers et al. 2018), where the user is offered a domain specific language to describe common algebraic expressions, which is then used to automatically generate a new FMU that wraps the old ones

[1] `https://github.com/CATIA-Systems/FMPy`

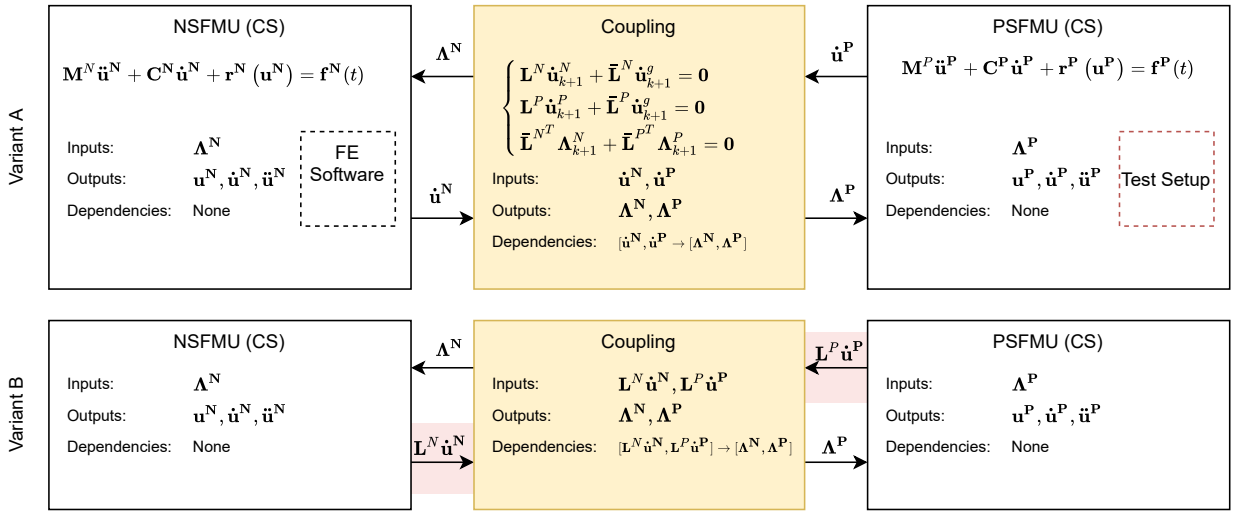

Figure 4. Illustration of two possible simulation scenarios that implement the setup described in Figure 1 in a co-simulation with the coupling algorithm introduced in Section 2.2. The dashed boxes represent the fact that NSFMU (respectively PSFMU) communicate with a FE Software (resp. the Test Setup), when the `fmi2DoStep` function is invoked. Variant B only shows the differences (highlighted in red) with respect to variant A. Each variant requires a different FMU implementation, with variant B allowing for less communication.

with those extra expressions, can be used for this. In addition, there are libraries that facilitate the implementation of FMUs, such as Moka (Aslan, Durak and Taylan 2015) and PyFMU (Legaard et al. 2020) (the later allows the user to program an FMU in Python).

However, for the scenario described in Figure 4, special attention must be payed to the algebraic dependencies: since FMI 2.1 supports no feedthrough, the computation of *Coupling* must be carried out in the `fmi2DoStep` function. This means that the `fmi2DoStep` of the *Coupling* FMU must be invoked after having provided the new inputs, originated from the outputs of the already stepped *NSFMU* and *PSFMU*. The consequence is that the co-simulation framework must accept some kind of description of the ordering in which the FMUs shall be stepped. The FMI standard 2.1 offers no mechanism to provide this information, even though it plays an important role in reducing co-simulation errors (Gomes, Thule, Lúcio et al. 2020; Oakes et al. 2021; Gomes, Oakes et al. 2019; Gomes, Lucio and Vangheluwe 2019).

Consistent Initialization. The implemented coupling algorithm, needs to be initialized with the jacobian of *NSFMU* and *PSFMU*. Since we had control over the implementation of these FMUs, we choose to simply expose these as outputs. In other cases, the `fmi2GetDirectionalDerivative` can be used to obtain this information.

3.2 FMI Model Exchange Interface

When implementing the scenarios described in Figure 4 using the FMI ME interface, the main difference lies in the ability to easily perform the state corrections described by Equations (15) and (16). This is illustrated in Figure 5.

The advantage is that the FMU is independent of the coupling algorithm, making this approach ideal for when the users have no control over the implementation of the FMUs (e.g., as in the case of FMI export features in M&S tools). The downside is the added complexity of the orchestration algorithm and the FMU exporter, which must also implement a greater number of FMI functions (compared to FMI CS) and ensure that the internal state is organized in a single vector.

4 Results and Conclusion

This paper describes multiple ways in which the FMI 2.1 for Model Exchange (ME) and Co-Simulation (CS) interfaces, can be used for the implementation of hybrid testing. To the best of our knowledge, there has not yet been any application of the FMI to the field of seismic hybrid testing and no standard has emerged to enable this. Having such standard would greatly facilitate the coupling of heterogeneous codes and facilitate the exchange of numerical models.

The approaches are discussed within the context of their implementation in an hybrid testing setup, installed at the Dynamisk LAB of Aarhus University. A snapshot of the numerical results can be seen in Figure 6, and a video recording of the experiment is available online at `https://youtu.be/-VkrQJaUo1o`.

The conclusion is that the both FMI ME and CS interfaces are well suited for this task, provided that the co-simulation framework supports minor customization of the ordering in which the FMUs are stepped in simulated time.

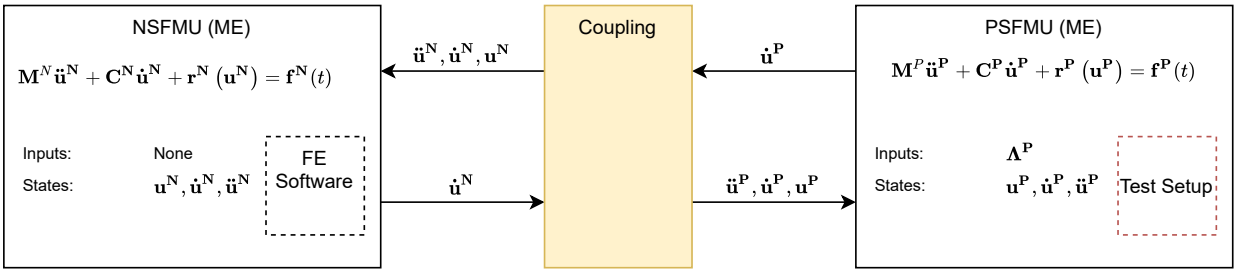

Figure 5. Illustration of simulation scenario implementing the setup of Figure 1 using the FMI Model Exchange interface. The coupling implementation is the same as in Figure 4. The dashed boxes represent the fact that NSFMU (respectively PSFMU) communicate with a FE Software (resp. the Test Setup), when the `fmi2GetReal` function is invoked.

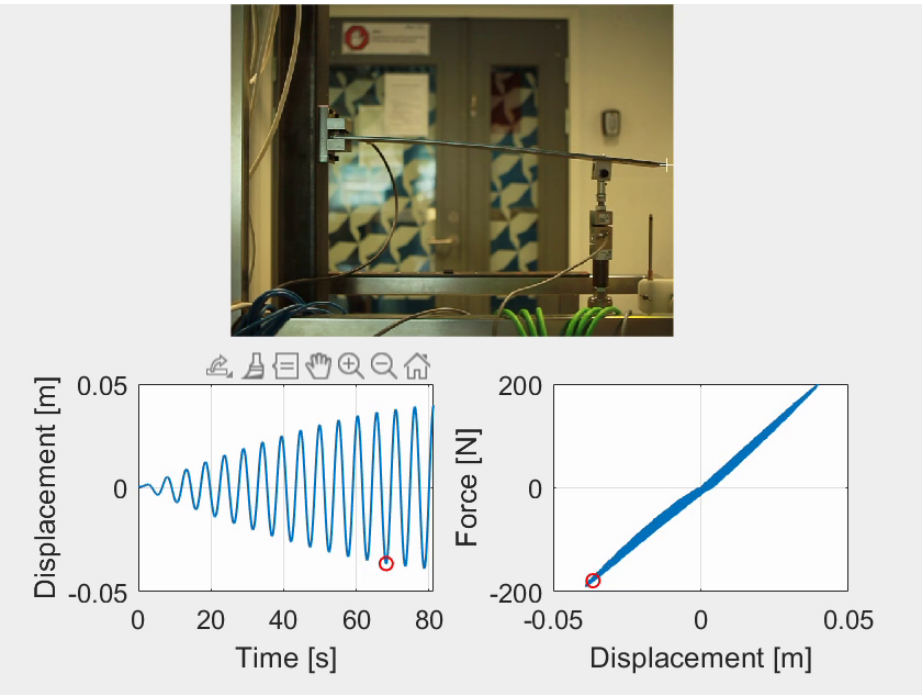

Figure 6. Numerical results using the FMI Co-simulation Interface. The results with Model Exchange are similar. The full video can be seen online at `https://youtu.be/-VkrQJaUo1o`.

Acknowledgements

We are grateful for the support from the Poul Due Jensen Foundation of the Centre for Digital Twin Technology at Aarhus University.

References

Abbiati, Giuseppe, Oreste S. Bursi et al. (2015-10). "Hybrid simulation of a multi-span RC viaduct with plain bars and sliding bearings". en. In: *Earthquake Engineering & Structural Dynamics* 44.13. ZSCC: NoCitationData[s0], pp. 2221–2240. ISSN: 00988847. DOI: 10.1002/eqe.2580. URL: http://doi.wiley.com/10.1002/eqe.2580 (visited on 2020-03-04).

Abbiati, Giuseppe, Patrick Covi et al. (2020-09). "A Real-Time Hybrid Fire Simulation Method Based on Dynamic Relaxation and Partitioned Time Integration". en. In: *Journal of Engineering Mechanics* 146.9, p. 04020104. ISSN: 0733-9399, 1943-7889. DOI: 10.1061/(ASCE)EM.1943-7889.0001826.

Abbiati, Giuseppe, Vincenzo La Salandra et al. (2018-02). "A composite experimental dynamic substructuring method based on partitioned algorithms and localized Lagrange multipliers". en. In: *Mechanical Systems and Signal Processing* 100, pp. 85–112. ISSN: 08883270. DOI: 10.1016/j.ymssp.2017.07.020. URL: https://linkinghub.elsevier.com/retrieve/pii/S0888327017303849 (visited on 2020-03-05).

Andersson, Christian, Johan Åkesson and Claus Führer (2016). *Pyfmi: A Python Package for Simulation of Coupled Dynamic Models with the Functional Mock-up Interface*. Centre for Mathematical Sciences, Lund University Lund.

Aslan, Memduha, Umut Durak and Koray Taylan (2015-07). "MOKA: An Object-Oriented Framework for FMI Co-Simulation". In: *Conference on Summer Computer Simulation*. Chicago, Illinois: Society for Computer Simulation International San Diego, CA, USA, pp. 1–8.

Bastian, Jens et al. (2011-06). "Master for Co-Simulation Using FMI". In: *8th International Modelica Conference*. Dresden, Germany: Linköping University Electronic Press, Linköpings universitet, pp. 115–120. DOI: 10.3384/ecp11063115.

Bathe, Klaus-JOrgen (1982). "Finite Element Procedures for Solids and Structures LinearAnalysis". In: *Finite Element Procedures*, pp. 148–214.

Brembeck, Jonathan et al. (2014-03). "Nonlinear State Estimation with an Extended FMI 2.0 Co-Simulation Interface". In: *10th International Modelica Conference*. Lund, Sweden: Linköping University Electronic Press; Linköpings universitet, pp. 53–62. DOI: 10.3384/ecp1409653.

Bursi O. S., Oreste S and Pui-Shum B. Shing (1996). "Evaluation of Some Implicit Time-Stepping Algorithms for Pseudodynamic Tests". en. In: *Earthquake Engineering & Structural Dynamics* 25, pp. 333–355.

Bursi, Oreste S. et al. (2017-11). "Nonlinear heterogeneous dynamic substructuring and partitioned FETI time integration for the development of low-discrepancy simulation models". en. In: *International Journal for Numerical Methods in Engineering* 112.9, pp. 1253–1291. ISSN: 00295981. DOI: 10.1002/nme.5556. URL: http://doi.wiley.com/10.1002/nme.5556 (visited on 2020-03-04).

Chopra, Anil K. (2012). *Dynamics of structures: theory and applications to earthquake engineering*. en. 4th ed. ZSCC: 0000002. Upper Saddle River, N.J: Prentice Hall. ISBN: 978-0-13-285803-8.

FMI v. 3.0 (2021). *Functional Mock-up Interface for Model Exchange, Co-Simulation, and Scheduled Execution*. https://fmi-standard.org/.

Gomes, Cláudio, Levi Lucio and Hans Vangheluwe (2019). "Semantics of Co-Simulation Algorithms with Simulator Contracts". In: *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. Munich, Germany: IEEE, pp. 784–789. DOI: 10.1109/MODELS-C.2019.00124.

Gomes, Cláudio, Bart Meyers et al. (2018). "Semantic Adaptation for FMI Co-Simulation with Hierarchical Simulators". In: *SIMULATION* 95.3, pp. 1–29. DOI: 10.1177/0037549718759775.

Gomes, Cláudio, Bentley James Oakes et al. (2019). "HintCO - Hint-Based Configuration of Co-Simulations". In: *International Conference on Simulation and Modeling Methodologies, Technologies and Applications*. Prague, Czech Republic, pp. 57–68. ISBN: 978-989-758-381-0. DOI: 10.5220/0007830000570068.

Gomes, Cláudio, Casper Thule, David Broman et al. (2018). "Co-Simulation: A Survey". In: *ACM Computing Surveys* 51.3, 49:1–49:33. DOI: 10.1145/3179993.

Gomes, Cláudio, Casper Thule, Levi Lúcio et al. (2020). "Generation of Co-Simulation Algorithms Subject to Simulator Contracts". en. In: *Software Engineering and Formal Methods*. Ed. by Javier Camara and Martin Steffen. Vol. 12226. Lecture Notes in Computer Science. Oslo, Norway: Springer International Publishing, pp. 34–49. ISBN: 978-3-030-57505-2 978-3-030-57506-9. DOI: 10.1007/978-3-030-57506-9_4.

Gravouil, A. and A Combescure (2001). "Multi-time-step explicit–implicit method for non-linear structural dynamics". In: *Int. J. Numer. Methods* 50 (1), pp. 199–225.

Hovmand, M., G. Abbiati and L. V. Andersen (2021). "Real-Time Hybrid Simulation with Nonlinear Numerical Substructures Based on State-Space Modeling". In: *17th World Conference on Earthquake Engineering*. Sendai, Japan, submitted.

Hovmand, Martin, Giuseppe Abbiati and Lars Vabbersgaard Andersen (2021). "Real-time hybrid simulation with nonlinear numerical substructures based on state-space modeling". In: *17 World Conference of Earthquake Engineering (17WCEE)*. Sendai, Japan.

Idinyang, Solomon et al. (2019-07). "Real-time data coupling for hybrid testing in a geotechnical centrifuge". en. In: *International Journal of Physical Modelling in Geotechnics* 19.4, pp. 208–220. ISSN: 1346-213X, 2042-6550. DOI: 10.1680/jphmg.17.00063. URL: https://www.icevirtuallibrary.com/doi/10.1680/jphmg.17.00063 (visited on 2020-03-07).

Kübler, R. and W. Schiehlen (2000). "Two Methods of Simulator Coupling". In: *Mathematical and Computer Modelling of Dynamical Systems* 6.2, pp. 93–113. ISSN: 1387-3954. DOI: 10.1076/1387-3954(200006)6:2;1-M;FT093.

Legaard, Christian Møldrup et al. (2020). "Rapid Prototyping of Self-Adaptive-Systems Using Python Functional Mockup Units". In: *2020 Summer Simulation Conference*. SummerSim '20. Virtual event: ACM New York, NY, USA, to appear.

Lelarasmee, E., Albert E. Ruehli and A. L. Sangiovanni-Vincentelli (1982). "The Waveform Relaxation Method for Time-Domain Analysis of Large Scale Integrated Circuits". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. Vol. 1, pp. 131–145. ISBN: 0278-00701. DOI: 10.1109/TCAD.1982.1270004.

McCrum, D.P. and M.S. Williams (2016-07). "An overview of seismic hybrid testing of engineering structures". en. In: *Engineering Structures* 118, pp. 240–261. ISSN: 01410296. DOI: 10.1016/j.engstruct.2016.03.039. URL: https://linkinghub.elsevier.com/retrieve/pii/S0141029616300748 (visited on 2020-03-05).

McKenna, Frank, Michael H Scott and Gregory L Fenves (2010). "Nonlinear Finite-Element Analysis Software Architecture Using Object Composition". en. In: *Journal of Computing in Civil Engineering* 24.1, p. 13.

Molina, Francisco J. et al. (2011-07). "Monitoring Damping in Pseudo-Dynamic Tests". en. In: *Journal of Earthquake Engineering* 15.6, pp. 877–900. ISSN: 1363-2469, 1559-808X. DOI: 10.1080/13632469.2010.544373. URL: http://www.tandfonline.com/doi/full/10.1080/13632469.2010.544373 (visited on 2020-03-05).

Nakashima, Masayoshi (2020-04). "Hybrid simulation: An early history". en. In: *Earthquake Engineering & Structural Dynamics* 49.10, pp. 949–962. ISSN: 00988847. DOI: 10.1002/eqe.3274. URL: http://doi.wiley.com/10.1002/eqe.3274 (visited on 2020-04-28).

Newton, Arthur Richard and Alberto L. Sangiovanni-Vincentelli (1983-09). "Relaxation-Based Electrical Simulation". In: *SIAM Journal on Scientific and Statistical Computing* 4.3, pp. 485–524. ISSN: 0196-5204. DOI: 10.1137/0904036.

Oakes, Bentley James et al. (2021). "Hint-Based Configuration of Co-Simulations with Algebraic Loops". en. In: *Simulation and Modeling Methodologies, Technologies and Applications*. Vol. 1260. Cham: Springer International Publishing, pp. 1–28. ISBN: 978-3-030-55866-6 978-3-030-55867-3. DOI: 10.1007/978-3-030-55867-3_1.

Pan, Peng, Tao Wang and Masayoshi Nakashima (2016). *Development of online hybrid testing: theory and applications to structural engineering*. en. Oxford [England] ; Waltham, MA: Elsevier / Butterworth Heinemann. ISBN: 978-0-12-803378-4.

Pegon, Pierre and Georges Magonette (2002). *Continuous PsD testing with nonlinear substructuring: presentation of a parallel inter-field procedure*. Tech. rep. I.02.167. Ispra, Italy: European Laboratory for Structural Assessment, Institute for the Protection and the Security of the Citizen, Joint Research Centre.

Sauca, A. et al. (2021-05). “Experimental validation of a hybrid fire testing framework based on dynamic relaxation”. en. In: *Fire Safety Journal* 121, p. 103315. ISSN: 03797112. DOI: 10.1016/j.firesaf.2021.103315. URL: https://linkinghub.elsevier.com/retrieve/pii/S0379711221000552 (visited on 2021-04-09).

Sauder, Thomas et al. (2016-06). “Real-Time Hybrid Model Testing of a Braceless Semi-Submersible Wind Turbine: Part I — The Hybrid Approach”. en. In: *Volume 6: Ocean Space Utilization; Ocean Renewable Energy*. Busan, South Korea: American Society of Mechanical Engineers, V006T09A039. ISBN: 978-0-7918-4997-2. DOI: 10.1115/OMAE2016-54435. URL: https://asmedigitalcollection.asme.org/OMAE/proceedings/OMAE2016/49972/Busan,%20South%20Korea/281288 (visited on 2020-05-01).

Schellenberg, Andreas, Stephen A. Mahin and Gregory L. Fenves (2007-10). “A Software Framework for Hybrid Simulation of Large Structural Systems”. en. In: *Structural Engineering Research Frontiers*. Long Beach, California, United States: American Society of Civil Engineers, pp. 1–16. ISBN: 978-0-7844-0944-2. DOI: 10.1061/40944(249)3. (Visited on 2020-03-04).

Stojadinovic, Bozidar, Gilberto Mosqueda and Stephen A. Mahin (2006-01). “Event-Driven Control System for Geographically Distributed Hybrid Simulation”. en. In: *Journal of Structural Engineering* 132.1, pp. 68–77. ISSN: 0733-9445, 1943-541X. DOI: 10.1061/(ASCE)0733-9445(2006)132:1(68). (Visited on 2020-03-04).