Fire Safety Journal 139 (2023) 103832

Contents lists available at ScienceDirect

Fire Safety Journal

journal homepage: www.elsevier.com/locate/firesaf

Hybrid fire testing using FMI-based co-simulation

G. Abbiati [a,*], E.E. Baş [b], C. Gomes [c], P.G. Larsen [c]

[a] University of Aarhus, Department of Civil and Architectural Engineering, Inge Lehmanns Gade, 10, 8000, Aarhus C, Denmark
[b] R&D Test Systems A/S, Sigma, 3, 8382 Hinnerup, Denmark
[c] University of Aarhus, Department of Electrical and Computer Engineering, Finlandsgade, 22, 8200, Aarhus C, Denmark

ARTICLE INFO

Keywords:
Hybrid fire testing
Functional mock-up interface standard
Functional mock-up unit
Co-simulation
Arduino

ABSTRACT

The failure mode of a structural component subjected to fire loading is sensitive to its mechanical boundary conditions. It follows that controlling boundary conditions is crucial to obtain a realistic loading scenario with a fire experiment. Hybrid fire testing has been developed for this purpose. Specifically, a structural component under test is loaded using actuators and exposed to fire loading. A coordination algorithm updates actuator setpoints on the fly so that the tested structural component experiences the sought mechanical boundary conditions, e.g., associated with a virtual assembly that is simulated numerically.

To enable interoperability of simulation tools and control systems utilized in hybrid testing, a number of middlewares have been proposed, some of which have been adapted to hybrid fire testing. Middlewares facilitate the implementation of hybrid fire testing within the same laboratory. However, the portability of hybrid models from one laboratory to another is critical when different middleware and simulation tools are adopted. A consequence of that is that round-robin verification cannot be easily deployed for hybrid fire testing.

In response to this limitation, this paper proposes to craft hybrid fire testing experiments using the Co-Simulation paradigm supported by the Functional Mock-up Interface standard. The methodology is demonstrated experimentally.

1. Introduction

1.1. Background and motivation

The failure mode of a structural component subjected to fire loading is sensitive to its Boundary Conditions (BCs). For example, thermally induced buckling of a restrained column exposed to fire might occur or not, depending on the stiffness of the connected structural system. However, the common situation in *standard fire testing* is that the experimenter has limited control of the BCs of the tested specimen. For example, a column can be easily tested with either fixed or pinned ends but not with any intermediate flexible restraint [1]. Hybrid Fire Testing (HFT) has been developed to overcome this limitation. HFT is carried out using a *hybrid model* that combines Physical and Numerical Substructures (PS and NS, respectively). The PS is enclosed in a furnace or exposed to a heat source, while servo-controlled actuators equipped with force transducers control mechanical BCs. Specifically, actuator position setpoints (or force setpoints) are adjusted to ensure force equilibrium and displacement compatibility with the NS, which is simulated using, for example, a Finite Element Analysis (FEA) software such as SAFIR [2]. Accordingly, the NS is crafted to provide some desired BCs that the PS shall experience while being exposed to fire loading. Mechanical and thermal actuation plus feedback transducers form the *transfer system* while the algorithm that updates actuator setpoints based on PS and NS responses *on the fly* is indicated as *coordination algorithm*. For a comprehensive state-of-art review of HFT, the reader is addressed to [3].

In order to reduce the effort associated with the coupling of experimental equipment and simulation software to enable hybrid testing, several *middlewares* have been developed. Remarkable work has been carried out in the domain of earthquake engineering, where hybrid testing was pioneered [4]. An incomplete list of middleware packages includes, in chronological order of appearance, *UI-SimCor* [5], *Online Hybrid Test System* [6], *ISEE* [7,8], *Open-Fresco* [9], *HybridFEM* [10], *CELESTINA* [11], *UT-SIM* [12], and JRC-ELSA framework [13]. The use of Open-Fresco as a middleware for HFT is demonstrated in [14,15] while the use of UT-SIM for the purpose of HFT is demonstrated in [16,17]. If one intends to enable the use of a new component (e.g., simulation software or control system) in HFT, a specific extension must be programmed for every supported middleware. Ideally, such effort shall be sustained once. However, this requires *standardizing* all interfaces of all software components involved in HFT. The challenge of improving the interoperability of simulation tools and control systems for the

* Corresponding author.
E-mail address: abbiati@cae.au.dk (G. Abbiati).

https://doi.org/10.1016/j.firesaf.2023.103832
Received 24 October 2022; Received in revised form 27 February 2023; Accepted 28 May 2023
Available online 31 May 2023

purpose of hybrid testing was raised for the first time by Huang & Kwon [12], who proposed a framework called UT-SIM. UT-SIM relies on TCP/UDP protocols to establish communication between PS, NS, and coordination algorithm. The data exchange format was formulated based on a pool of existing middlewares. UT-SIM has been applied to perform HFT of a stiff steel column subjected to axial load and enclosed into a furnace (PS and transfer system) coupled with a frame FEA model implemented in Abaqus [18] (NS) [16,17].

As identified by Huang & Kwon [12], interoperability of simulation tools is necessary to facilitate the implementation of HFT. However, to exchange hybrid models across laboratories, *portability* of simulation tools is as critical as their *interoperability*. A crucial motivation for that is the round-robin verification of HFT experiments. In experimental methodology, a round-robin verification consists of a pool of laboratories performing the same experiment independently and cross-validating their results. To enable round-robin verification campaigns, a pool of laboratories shall be able to deploy the same hybrid model regardless of its specific software infrastructure (specific release of simulation tools and middleware).

The Functional Mock-up Interface (FMI) standard, originally introduced in the automotive sector to craft complex vehicle *Co-Simulations* by coupling domain-specific simulation environments [19,20] is a viable solution for crafting portable hybrid models. Specifically, the FMI standard defines how to encapsulate simulation models into standardized containers to form Functional Mock-up Units (FMUs) [21], which are coupled in a Co-Simulation through standard interfaces. However, to the authors' knowledge, FMI-based Co-Simulation is not yet exploited to support HFT.

1.2. Scope

The FMI standard is organized under the roof of the *Modelica Association* [22]. The institutionalization of the FMI standard dates back to 2011 (onset of *Industry 4.0* revolution) when the *Modelica Association* [22] was funded as an outcome of the *Modelisar ITEA-2* project [23] led by the automotive industry. As of today, the FMI standard is officially supported by +170 simulation tools [24] and provides templates for supporting FMU import/export in several programming languages. In essence, the FMI standard defines *interface* and *container* specifications that an FMU must comply with to be linked with other FMUs within a Co-Simulation [19,21]. In practice, an FMU is a `*.ZIP` file that encapsulates either the binaries of a simulation model (e.g., an FEA model) or the Application Programming Interface (API) necessary to communicate with a physical system (e.g., a control system), and an `*.XML` file describing the FMU input/output variables.

The FMU API follows the FMI standard and includes an (i) `init()` method that initializes the FMU, a (ii) `doStep()` method that solves a single analysis step and (iii) `set()` and `get()` methods that read/write output/input variables. An FMU can be either generated using the FMU exporter of specific simulation software – when available – or programmed directly using existing software libraries such as *Moka* [25] and *UniFMU* [26]. In order to check that FMUs comply with the FMI standard it is possible to use *VDMCheck* [27].

Co-Simulation is executed by an *orchestrator*, which is a computer program that initializes all the FMUs and coordinates their execution and data exchange following the specific Co-Simulation *scenario*. Specifically, the Co-Simulation scenario describes the inter-connections between input and output variables of each FMU while the orchestrator software enforces these input–output relationships at each step of the Co-Simulation using a numerical solver for nonlinear algebraic equations (e.g., based on Jacobi or Gauss–Seidel algorithm) [28,29]. Gomes et al. [20,30] provides a broad overview of Co-Simulation.

Because of its inherent modularity and compartmental structure, FMI-based Co-Simulation is an efficient framework for coupling PS and NS of a hybrid model in HFT. However, some constraints associated with the FMI standard render some implementation choices preferable to others. To the authors' knowledge, a comprehensive account in this regard is still missing. It follows that the original contribution of this paper is a methodology for crafting FMUs and Co-Simulation scenario for the purpose of HFT.

The paper is organized as follows. Section 2 presents the Co-Simulation-based HFT framework. First, the HFT concept, along with a mathematical description of the hybrid model, is illustrated. Then, the methodology for crafting FMUs and Co-Simulation scenario for HFT with one PS and one NS and a generic coordination algorithm is demonstrated. Section 3 describes the experimental validation of the proposed methodology using a single-Degree-of-Freedom (single-DoF) hybrid model. FMUs are crafted for two different coordination algorithms taken from the state of the art [31,32]. The orchestrator software *Maestro* [33] developed by the research group of *Cyber–Physical Systems* of Aarhus University, Denmark, is utilized to conduct Co-Simulation. Section 4 summarizes the findings of this study.

2. Co-simulation-based HFT

2.1. Hybrid model equations

The mechanical response of a structural system exposed to fire loading is typically quasi-static. Therefore, the hybrid model response can be described by a static equilibrium equation with a reasonable degree of approximation. The following set of equations expresses the static equilibrium of a multiple-DoFs hybrid model and the compatibility of displacements at the interface of PS and NS,

$$\begin{cases} \mathbf{f}^P_{int}(\mathbf{u}^P, \theta^P(t)) = \mathbf{f}^P_{ext}(t) + \mathbf{L}^{P\,T}\boldsymbol{\Lambda} \\ \mathbf{f}^N_{int}(\mathbf{u}^N, \theta^N(t)) = \mathbf{f}^N_{ext}(t) + \mathbf{L}^{N\,T}\boldsymbol{\Lambda} \\ \mathbf{L}^P\mathbf{u}^P + \mathbf{L}^N\mathbf{u}^N = \mathbf{0} \end{cases} \tag{1}$$

where $\mathbf{f}^P_{int}(\mathbf{u}^P, \theta^P(t))$ and $\mathbf{f}^N_{int}(\mathbf{u}^N, \theta^N(t))$ are the internal restoring force vectors of the PS and the NS, subjected to the corresponding displacement $(\mathbf{u}^P, \mathbf{u}^N)$ and temperature fields $(\theta^P(t), \theta^N(t))$; $\mathbf{f}^P_{ext}(t)$ and $\mathbf{f}^N_{ext}(t)$ are the vectors of externally applied loading; $\boldsymbol{\Lambda}$, is the vector of interface forces that couple PS and NS; $\mathbf{L}^P$ and $\mathbf{L}^N$ are signed Boolean collocation matrices used to cast interface compatibility and equilibrium equation consistently [34]. It is important to remark that memory variables associated with the internal restoring forces, which account for memory effects such as hysteresis, are purposely omitted. The reason is that such variables are not exchanged among the various component of the HFT.

For the sake of simplicity and, as often done in HFT, both PS and NS equilibrium equations are condensed to the coupled DoFs. It follows that the signed Boolean matrices reduce to

$$\mathbf{L}^P = -\mathbf{L}^N = \mathbf{I} \tag{2}$$

and

$$\mathbf{u}^P = \mathbf{u}^N = \mathbf{u} \tag{3}$$

where $\mathbf{I}$ is the identity matrix. For this specific case, the equilibrium equation of the hybrid model reported in (1) becomes

$$\mathbf{f}^P_{int}(\mathbf{u}, \theta^P(t)) + \mathbf{f}^N_{int}(\mathbf{u}, \theta^N(t)) = \mathbf{f}^P_{ext}(t) + \mathbf{f}^N_{ext}(t) \tag{4}$$

In order to simplify the description of FMI-based HFT, it is convenient to reformulate (4) as,

$$\Delta\mathbf{f}^P(\mathbf{u}, t) + \Delta\mathbf{f}^N(\mathbf{u}, t) = \mathbf{0} \tag{5}$$

where PS and NS unbalanced forces are defined as,

$$\Delta\mathbf{f}^P(\mathbf{u}, t) = \mathbf{f}^P_{int}(\mathbf{u}, \theta^P(t)) - \mathbf{f}^P_{ext}(t)\ ,\ \Delta\mathbf{f}^N(\mathbf{u}, t) = \mathbf{f}^N_{int}(\mathbf{u}, \theta^N(t)) - \mathbf{f}^P_{ext}(t) \tag{6}$$

As one can notice from (6), the time dependency of the unbalanced forces $\Delta\mathbf{f}^P(\mathbf{u}, t)$ and $\Delta\mathbf{f}^N(\mathbf{u}, t)$ hides the time-dependency of temperature field and externally applied loading for each substructure.

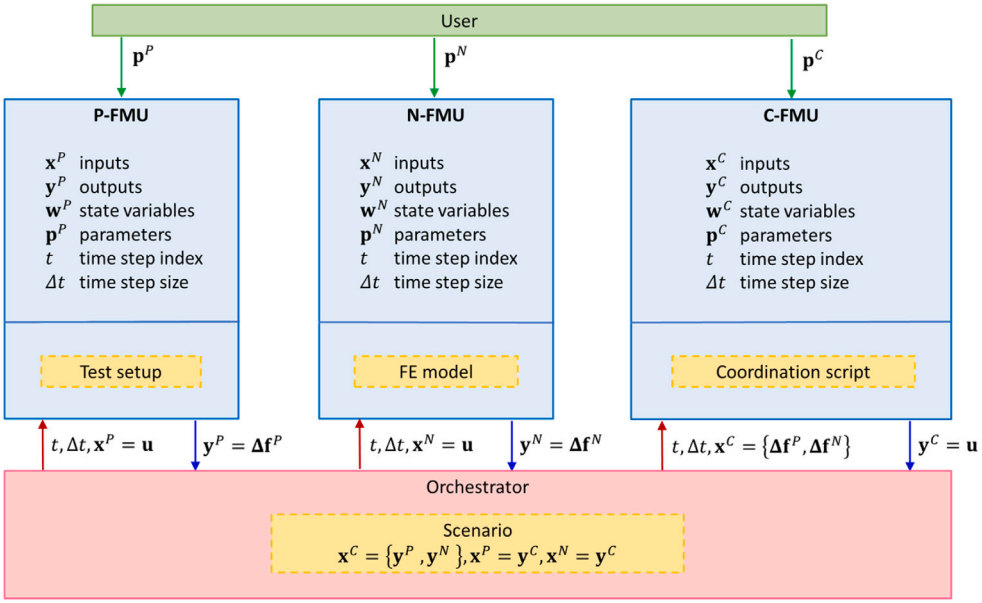

Fig. 1. Block diagram of Co-Simulation HFT considering one PS and one NS.

The majority of coordination algorithms have been developed for *displacement-control* mode HFT. In principle, the coordination algorithm updates $\mathbf{u}$ until the norm of (5) is less of a given tolerance. A few exceptions are the procedures presented in [35], and [36], which perform HFT in *force-control* mode. Also, coordination algorithms can be categorized in two groups, namely *control-based* and *simulation-based* coordination algorithms. The control-based group contains all coordination algorithms obtained as the solution to a control system design problem. The coordination algorithms presented in [31,36–38] belong to this category. The simulation-based group contains all coordination algorithms obtained from numerical solvers for systems of nonlinear differential/algebraic equations. The coordination algorithms presented in [14,16,17,32,39–41] belong to this category. As displacement-control mode HFT is the dominant setting, this paper demonstrates Co-Simulation-based HFT for this specific case considering both a control-based and a simulation-based coordination algorithm taken from the state of the art [31,32].

2.2. FMUs and Co-Simulation scenario

The block diagram of Fig. 1 illustrates the Co-Simulation-based HFT concept for a hybrid model comprising one PS and one NS. In detail, the *P-FMU* incorporates the API to the test setup hosting the PS, while the *N-FMU* incorporates the simulation model utilized as NS. As HFT is performed in displacement-control mode, both P-FMU and N-FMU receive displacements as input variables $\mathbf{x}^{(\bullet)}$ and produce unbalanced forces as output variables $\mathbf{y}^{(\bullet)}$. If the solution algorithm utilized to compute the NS response or the control algorithm utilized to compute the PS BCs requires additional state variables, they are stored in vector $\mathbf{w}^{(\bullet)}$. Vector $\mathbf{p}^{(\bullet)}$ contains all constant parameters. As an example, if some of the PS DoFs are loaded but not included in the HFT simulation loop, $\mathbf{p}^P$ stores the parameters defining the BC histories applied to such DoFs. The HFT coordination algorithm is implemented as an additional FMU, namely the *C-FMU*. As HFT is performed in displacement-control mode, the C-FMU receives unbalanced forces as input variables $\mathbf{x}^C$ and produces displacement as output variable $\mathbf{y}^C$. Additional state variables and constant parameters utilized by the coordination algorithms are stored in $\mathbf{w}^C$ and $\mathbf{p}^C$, respectively. At the Co-Simulation start, the orchestrator executes the `init()` method of each FMU once so that $\mathbf{x}^{(\bullet)}$, $\mathbf{x}^{(\bullet)}$, $\mathbf{w}^{(\bullet)}$ and $\mathbf{p}^{(\bullet)}$ are initialized. At each time step, the orchestrator executes the `doStep()` method of each FMU to carry out one analysis step. Then, the orchestrator transfers data according to the input–output relationships encoded in the Co-Simulation scenario. The procedure requires one iteration to enforce the input–output relationships specified by the Co-Simulation scenario. Then, the Co-Simulation moves to the next analysis step. The proposed methodology does not alter the inner workings of the specific coordination algorithm. Therefore, if a given coordination algorithm is endowed with proof of validation, one can confidently adopt it in the present FMI-based implementation. For the Co-Simulation scenario illustrated in Fig. 1, special attention must be paid to the algebraic dependencies introduced by the coordination algorithm. Noteworthy, the `doStep` method of the C-FMU must be invoked after having provided the new inputs, originating from the outputs of the already stepped P-FMU and N-FMU. The consequence is that the Co-Simulation framework must accept a description of the ordering in which the FMUs shall be stepped [42–45]. The FMI standard 2.0 offers no such mechanism, which is, nevertheless, implemented in Maestro [33], the orchestrator adopted in this study. In addition, based on the Co-Simulation scenario and the specifications of each single FMU, Maestro uses *formal methods* to verify, for example, that the execution of FMUs does not violate causality or that exchanged signals have consistent units. Warning/error messages are generated accordingly [46]. Maestro is supported and maintained by the INTO-CPS non-for-profit association [47]. A number of alternative packages exist, and an incomplete list of similar projects includes *Coral*, *DACCOSIM*, and *FMIGo* [48].

The need for supporting a large number of simulation tools has made the FMI standard extremely versatile. The consequence is that one can adopt alternative strategies to craft the same Co-Simulation, some of which are more efficient than others.

The overall philosophy of Co-Simulation is that the orchestrator schedules the execution of the FMUs and enforces their input/output relationships as listed in the scenario. It follows that the orchestrator typically allows for little customization [49,50]. In the preliminary stage of this study, the coordination algorithm was implemented in the orchestrator, Maestro, which is written in *Java*. Java is designed to enable the development of portable, high-performance applications for the widest range of computing platforms possible. Platform independency is not the sole reason behind the selection of Java. Java is a *strongly typed* and *object oriented* programming language, which means that it demands the declaration of each variable with a data type that cannot be changed dynamically. This feature is crucial to implement

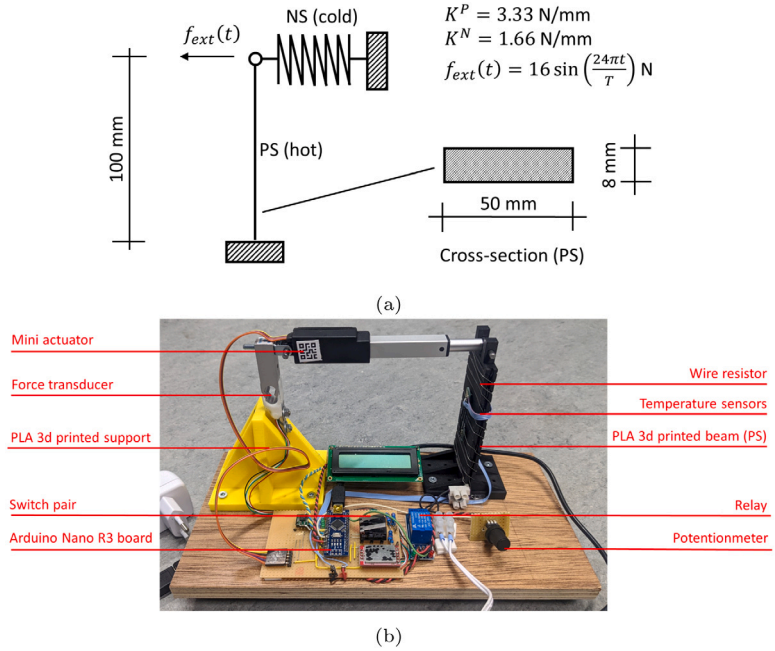

Fig. 2. Experimental case study utilized to validate the Co-Simulation-based HFT framework: (a) hybrid model; (b) experimental setup.

formal methods for model verification. However, Java is inefficient for the matrix computations involved in numerical analysis. It follows that the authors abandoned this approach in favor of implementing the coordination algorithm into a dedicated FMU, namely, the C-FMU using the *Python* template provided by UniFMU [26]. Furthermore, this approach allows the user to swap the coordination algorithm of the hybrid model easily.

3. Experimental validation

3.1. Hybrid model

The experimental case study utilized to demonstrate the proposed framework consists of a single-DoF hybrid model where a 3D printed cantilever beam made of Polylactic Acid (PLA) (PS) is coupled with a linear elastic spring (NS). The PS is exposed to a heat source, whereas the NS is cold. The single-DoF hybrid model is subjected to a sinusoidal force, which produces a cyclic displacement response. The hybrid model is schematically depicted in Fig. 2(a) while the experimental setup with the PS installed is reported in Fig. 2(b).

As can be appreciated from Fig. 2(b), a mini linear electric actuator (50 mm stroke, 200 N force capacity) control the PS deflection. The actuator is fixed to a force transducer that measures the restoring force, whereas an internal resolver measures the position. Another PLA block provides fixation to the force transducer and, therefore, to the actuator. Both displacement- and force-control modes are possible for the actuator. Thermal loading is applied to the PS using a 26 Ohm wire resistor supplied with 5 V through a digital switch. Four digital temperature sensors measure the PS temperature response. An Arduino Nano R3 board runs the bang–bang controllers [51] of both the actuator and wire resistor. A Windows PC sends the actuator position/force setpoint and the temperature setpoint to the Arduino board using a serial interface implemented in Python. Using a pair of switches, one can manually adjust the actuator position inwards and outwards to facilitate the installation of the specimen. Similarly, the temperature setpoint can be adjusted using a potentiometer.

HFT is implemented in displacement-control mode. Therefore both N-FMU and P-FMU receive a displacement command as input and send corresponding unbalanced forces as output. In both cases, FMUs are implemented as Python scripts using UniFMU [26]. In the N-FMU, the unbalanced force is computed as,

$$\Delta f^N = K^N u - 0.5 f_{ext}(t) \tag{7}$$

where K^N is the stiffness of the NS, u the displacement and $f_{ext}(t)$ is the externally applied loading as defined in Fig. 2(a). The P-FMU wraps the Python interface to the Arduino board, which receives displacement and temperature setpoints and feeds the corresponding response signals back. The PS restoring force is corrected as suggested in [52] to compensate for limited actuation accuracy. The expression of the unbalanced force reads,

$$\Delta f^P = f^P_{int,mes} + K^P(u - u_{mes}) - 0.5 f_{ext}(t) \tag{8}$$

where u_{mes} and $f^P_{int,mes}$ are the displacement and restoring force measured from the PS, and K^P is an estimate of the PS tangent stiffness obtained with small displacement perturbations in cold conditions.

In order to promote the dissemination of HFT, the experimental setup has been purposely designed to fit a 150×300 mm plate and relies on cheap standard electronics available out-of-shelf. All remaining parts are 3D printed. The full documentation of the project, including the schematic of the Arduino board, the list of components, the geometry files for 3D printing, and the source code, are available at [53]. The authors deem it important to remark that the choice of crafting an application example characterized by a single-DoF PS is meant to facilitate the reader in reproducing the experiment independently with little budget. However, the methodology presented in Section 2 is not limited to single-DoF hybrid models.

3.2. Coupling FMUs

In the following, the Co-Simulation-based implementation of HFT is illustrated for the control-based coordination algorithm of Mergny et al. [31], and a monolithic variant of the simulation-based coordination algorithm of Abbiati et al. [32]. Both algorithms are conceived for HFT in displacement-control mode and are illustrated to solve the equilibrium equation of the hybrid model as formulated in (4). To simplify the notation, displacement, and time dependencies in $\Delta\mathbf{f}^{(\bullet)}(\mathbf{u}, t)$ are omitted hereinafter.

3.2.1. Control-based coordination algorithm

The selected control-based HFT algorithm is the one proposed by Mergny et al. [31]. The algorithm consists of a PI controller designed to minimize unbalanced forces between PS and NS while displacement compatibility is assumed a priori. The procedure for computing the hybrid model response from t_k to t_{k+1} using a time step $\Delta t = t_{k+1} - t_k$ is reported as a sequence of steps:

1. Compute the displacement command at t_{k+1} as

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{L}_e\mathbf{e}_k + \mathbf{L}_j\mathbf{j}_k \tag{9}$$

   where $\mathbf{L}_e$ and $\mathbf{L}_j$ are gain matrices for the instantaneous interface force unbalance error $\mathbf{e}_k$ and the corresponding integral $\mathbf{j}_k$.
2. Evaluate the corresponding unbalanced forces for both NS and PS as

$$\Delta\mathbf{f}^P_{k+1} = \mathbf{f}^P_{int,k+1} - \mathbf{f}^P_{ext,k+1} \tag{10}$$

$$\Delta\mathbf{f}^N_{k+1} = \mathbf{f}^N_{int,k+1} - \mathbf{f}^N_{ext,k+1} \tag{11}$$

   $\Delta\mathbf{f}^N_{k+1}$ is numerically evaluated by solving the static response of the NS subjected to the displacement field $\mathbf{u}_{k+1}$ and the temperature field θ^N_{k+1}, e.g., using a FEA software. $\Delta\mathbf{f}^P_{k+1}$ is experimentally measured after imposing the displacement field $\mathbf{u}_{k+1}$ and the temperature field θ^P_{k+1} to the PS using the transfer system.
3. Compute the instantaneous residual force $\mathbf{e}_{k+1}$ and update the corresponding integral $\mathbf{j}_{k+1}$ as

$$\mathbf{e}_{k+1} = -(\Delta\mathbf{f}^N_{k+1} + \Delta\mathbf{f}^P_{k+1}) \tag{12}$$

$$\mathbf{j}_{k+1} = \mathbf{j}_k + \mathbf{e}_k\Delta t \tag{13}$$

The described procedure loops from (1) to (3) until the end of the loading sequence. In [31], it is proposed to compute gain matrices as

$$\mathbf{L}_e = 2\left(\mathbf{K}^P + \mathbf{K}^N\right)^{-1} c_e \tag{14}$$

$$\mathbf{L}_j = \left(\mathbf{K}^P + \mathbf{K}^N\right)^{-1} c_j \tag{15}$$

where

$$\mathbf{K}^P = \frac{\partial \mathbf{f}^P_{int}}{\partial \mathbf{u}}\Big|_{\mathbf{u}=\mathbf{0},\theta=\mathbf{0}}, \mathbf{K}^N = \frac{\partial \mathbf{f}^N_{int}}{\partial \mathbf{u}}\Big|_{\mathbf{u}=\mathbf{0},\theta=\mathbf{0}} \tag{16}$$

The NS stiffness matrix $\mathbf{K}^N$ is estimated numerically, whereas the PS stiffness matrix $\mathbf{K}^P$ is experimentally measured by applying small displacement perturbations to the specimen. The parameters $c_e, c_j \in [0, 1]$ are used to modulate the controller gain. In the proposed implementation, the `init()` methods of P-FMU and N-FMU save the tangent stiffness matrices of the PS and the NS and are executed before the `init()` method of the C-FMU. The latter reads such matrices and computes the gain matrices $\mathbf{L}_e$ and $\mathbf{L}_j$. A block diagram of the C-FMU obtained from the control-based coordination algorithm of [31] is reported in Fig. 3.

3.2.2. Simulation-based coordination algorithm

The selected simulation-based HFT coordination algorithm is a monolithic variant of the time integration scheme proposed by Abbiati et al. [32]. The algorithm coincides with the *dynamic relaxation* method described in [54]. The basic idea behind dynamic relaxation is to obtain the displacement solution of a static structural problem by computing the transient response of an equivalent dynamic system, whose equation of motion reads,

$$\mathbf{M}\ddot{\mathbf{u}}_k + \mathbf{C}\dot{\mathbf{u}}_k + \mathbf{f}^P_{int,k} + \mathbf{f}^N_{int,k} = \mathbf{f}^P_{ext,k} + \mathbf{f}^N_{ext,k} \tag{17}$$

where $\mathbf{M}$ and $\mathbf{C}$ are fictitious mass and damping diagonal matrices computed as

$$M_{ii} = \frac{(1.1\Delta t)^2}{4}\sum_j |K_{ij}| \tag{18}$$

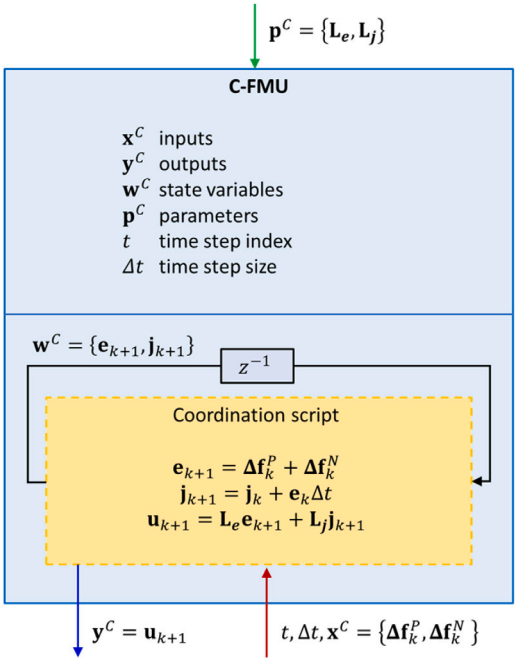

Fig. 3. Block diagram of the C-FMU implementation of HFT based on the control-based coordination algorithm of [31].

$$C_{ii} = 2\omega_0 M_{ii} \tag{19}$$

where K_{ij} is a generic entry of the initial tangent stiffness of the hybrid model $\mathbf{K} = \mathbf{K}^P + \mathbf{K}^N$ as defined in (16), and ω_0 is the lowest undamped frequency associated with the matrix pair $\{\mathbf{K}, \mathbf{M}\}$ while Δt is the time step size of the equivalent transient analysis. The central difference algorithm, which is equivalent to the Newmark algorithm [55] with $\gamma = \frac{1}{2}$ and $\beta = 0$, is used to solve (17). The procedure for computing the hybrid model response from t_k to t_{k+1} using a time step $\Delta t = t_{k+1} - t_k$ is reported as a sequence of steps:

1. Compute displacement and pseudo-velocity vectors using Newmark's extrapolation equations

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \dot{\mathbf{u}}_k\Delta t + \frac{1}{2}\Delta t^2\ddot{\mathbf{u}}_k \tag{20}$$

$$\dot{\mathbf{u}}_{k+1} = \dot{\mathbf{u}}_k + \frac{1}{2}\Delta t\ddot{\mathbf{u}}_k \tag{21}$$

2. Evaluate the corresponding unbalanced forces for both NS and PS as

$$\Delta\mathbf{f}^P_{k+1} = \mathbf{f}^P_{int,k+1} - \mathbf{f}^P_{ext,k+1} \tag{22}$$

$$\Delta\mathbf{f}^N_{k+1} = \mathbf{f}^N_{int,k+1} - \mathbf{f}^N_{ext,k+1} \tag{23}$$

   $\Delta\mathbf{f}^N_{k+1}$ is numerically evaluated by solving the static response of the NS subjected to the displacement field $\mathbf{u}_{k+1}$ and the temperature field θ^N_{k+1}, e.g., using a FEA software. $\Delta\mathbf{f}^P_{k+1}$ is experimentally measured after imposing the displacement field $\mathbf{u}_{k+1}$ and the temperature field θ^P_{k+1} to the PS using the transfer system.[1]
3. Compute the pseudo-acceleration vector at t_{k+1}

$$\ddot{\mathbf{u}}_{k+1} = -\mathbf{D}^{-1}\left(\Delta\mathbf{f}^P_{k+1} + \Delta\mathbf{f}^N_{k+1} + \mathbf{C}\dot{\mathbf{u}}_{k+1}\right) \tag{24}$$

   with

$$\mathbf{D} = \mathbf{M} + \mathbf{C}\frac{\Delta t}{2} \tag{25}$$

[1] This step is identical to step (2) of the control-based HFT procedure described in 3.2.1

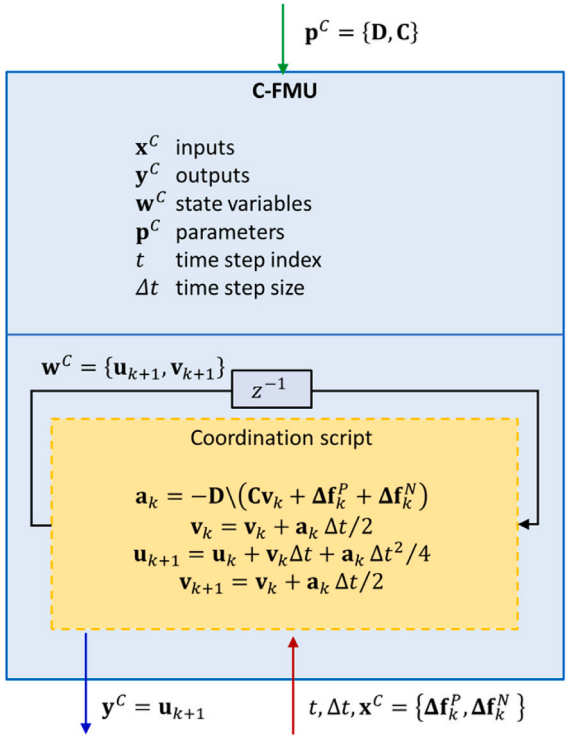

Fig. 4. Block diagram of the C-FMU implementation of HFT based on the simulation-based coordination algorithm of [32].

computed and inverted once before the HFT.

4. Update the pseudo-velocity vector based on the computed pseudo-acceleration vector

$$\dot{\mathbf{u}}_{k+1} = \dot{\mathbf{u}}_{k+1} + \ddot{\mathbf{u}}_{k+1}\frac{1}{2}\Delta t \tag{26}$$

The described procedure loops from (1) to (4) until the end of the loading sequence.

The expression of M_{ii} forces the upper bound ω_{max} of the eigenfrequencies associated with the matrix pair $\{\mathbf{K}, \mathbf{M}\}$ to be slightly smaller than $\omega_{lim} = 2/\Delta t$, which defines the stability limit of the central-difference method. Specifically, the upper bound ω_{max} is defined upon Gershgorin's circle theorem [56]. It is well known that the rate of convergence of the dynamic relaxation method deteriorates when $\omega_0/\omega_{max} \ll 1$. Therefore, in [32], dynamic relaxation is used in combination with model-order reduction. In the proposed implementation, the `init()` methods of P-FMU and N-FMU save the tangent stiffness matrices of the PS and the NS. The `init()` method of the C-FMU is executed afterward and reads such matrices and computes the dynamic tangent stiffness matrix **D** and the fictitious damping matrix **C** utilized by the coordination algorithm. A block diagram of the C-FMU obtained from the simulation-based coordination algorithm of [32] is reported in Fig. 4.

3.3. Results and discussion

Table 1 provides an overview of the performed HFTs. As can be appreciated, for all tests, a sinusoidal force of 16 N amplitude is applied as mechanical loading and equally distributed between PS and NS. With regard to thermal loading, Test #1 is exposed to a power source $\phi(t)$ of 1.0 W for the entire test, resulting in a parabolic-like time–temperature response. Tests #2 and #3 are performed by imposing to the PS a linear temperature ramp from 22 °C (room temperature) to 50 °C with temperature rate $\dot{\theta}(t) = 7.5 \times 10^{-3}$ °C/s. Fig. 5 collects all experimental results related to Test #1, namely, displacement,

Table 1
List of HFT experiments.

Test ID	Coordination algorithm	Thermal loading	Duration
1	Mergny et al. [31]	$\psi(t) = 1.0$ W	1720 s
2	Mergny et al. [31]	$\dot{\theta}(t) = 7.5 \times 10^{-3}$ °C/s	3370 s
3	Abbiati et al. [32]	$\dot{\theta}(t) = 7.5 \times 10^{-3}$ °C/s	3370 s

force, and temperature response of the PS as well as both PS and NS unbalanced forces and related residual. As can be appreciated, the force residual is small compared to PS and NS unbalanced forces meaning that the coordination algorithm described in Section 3.2.1 successfully established balance between substructures. Figs. 6 and 7 summarize the results of Tests #2 and #3. As one can see, in both experiments, a sudden force peak occurred at about ≈ 1670 s in Test #2 and ≈ 970 s in Test #3, respectively. After examining the PS force response history acquired through the force transducer, it was observed that, in both Test #2 and Test #3, suddenly, the force measurement jumped to ≈ −55.0 N. These sudden force peaks, which did not occur in Test #1, propagated to the displacement computed by the HFT algorithm in the subsequent time step. In order to understand the cause of the restoring force peaks, 5–6 additional tests have been conducted for each of the settings characterizing Test #1, #2, and #3. The same sudden restoring force peak of ≈ −55.0 N systematically occurred for all tests with the same setting of Test #2 and #3 at a random point in time, whereas, the same restoring force peak did not manifest in all tests conducted with the same setting of Test #1.

The main difference between Test #1 and the other two tests is that, during Test #1, the relay module was constantly *on* to deliver constant electric power to the wire resistor. In contrast, during Tests #2 and #3, the temperature controller switched the relay *on* and *off* approximately every 2 s to track the linear temperature ramp. As the time allocated to the experimental campaign was limited, the malfunctioning of the force transducer was not fully resolved. However, the most plausible explanation is that the relatively fast switching of the relay caused some electric disturbance to the force transducer, which produced a wrong measurement at the systematic value of ≈ −55.0 N.

Instead of discarding these results, they have been reported as they provide a deep insight into the behavior of the two coordination algorithms under *stress testing*. As can be appreciated from Fig. 6, during Test #2, which utilizes the control-based HFT algorithm of [31], the sudden force perturbation is damped out almost instantaneously. A very similar perturbation in Test #3, which is carried out with the simulation-based HFT algorithm of [32], is damped out after a few displacement cycles as demonstrated in Fig. 7. It is reasonable to conclude that the control-based HFT algorithm shows better performance in terms of disturbance rejection.

4. Conclusions

A hybrid fire testing experiment cannot be easily reproduced in two different laboratories if they use different middlewares and simulation tools. An important consequence is that the feasibility of round-robin verification hybrid fire testing experiments is limited. In response to this limitation, this paper proposes a methodology for conducting hybrid fire testing using Co-Simulation as framed by the Functional Mock-up Interface standard. Specifically, the methodology defines how to partition the hybrid model into functional mock-up units for the physical substructure, the numerical substructure, and the coordination algorithm. The methodology is demonstrated for an Arduino-based experimental case study characterized by a single degree of freedom considering two existing coordination algorithms. The same methodology is valid for multiple-degrees-of-freedom hybrid models. The advantage of implementing the coordination algorithm as a functional mock-up unit is that the coordination algorithm is disentangled from the orchestrator. Therefore, the user can swap the coordination algorithm more

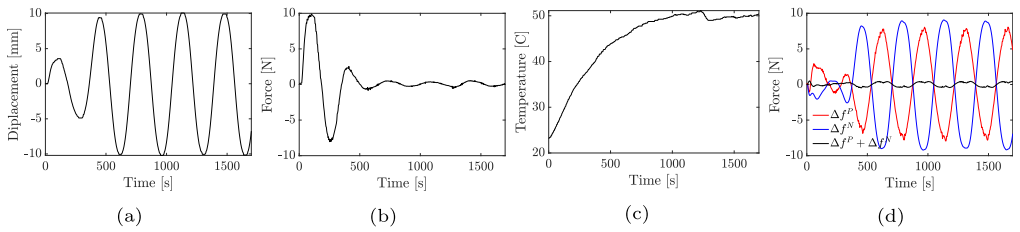

Fig. 5. Experimental results from Test #1 (constant heat power source, control-based HFT algorithm of [31]): (a) displacement, (b) restoring force, and (c) temperature response of the PS, and (d) unbalanced forces of PS and NS compared to residual force error.

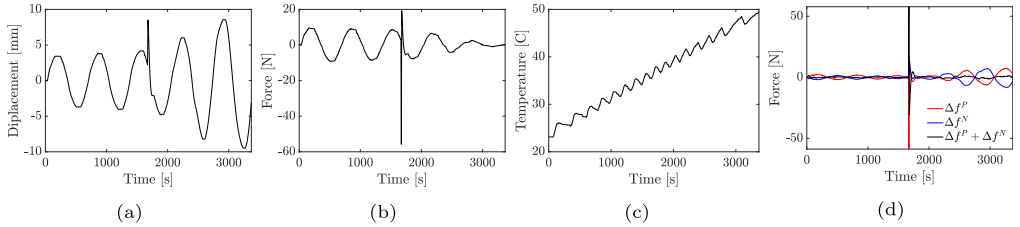

Fig. 6. Experimental results from Test #2 (linear temperature ramp, control-based HFT algorithm of [31]): (a) displacement, (b) restoring force, (c) temperature response of the PS, and (d) unbalance forces of PS and NS compared to residual force error.

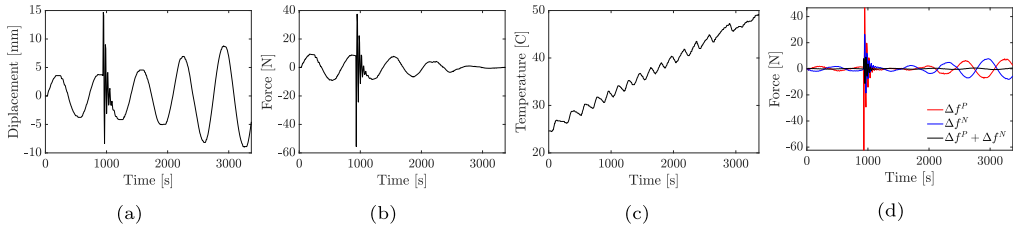

Fig. 7. Experimental results from Test #3 (linear temperature ramp, simulation-based HFT algorithm of [32]): (a) displacement, (b) restoring force, (c) temperature response of the PS, and (d) unbalance forces of PS and NS compared to residual force error.

safely. Moreover, the implementation of the coordination algorithm is not constrained to the programming language in which the orchestrator is implemented, which might not be suited for matrix computations.

CRediT authorship contribution statement

G. Abbiati: Conceptualization, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. **E.E. Baş:** Conceptualization, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. **C. Gomes:** Conceptualization, Methodology, Software, Writing – original draft. **P.G. Larsen:** Conceptualization, Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

The authors acknowledge the support of the Poul Due Jensen Foundation for funding the Centre for Digital Twin Technology at Aarhus University, Denmark, which triggered this research collaboration. Prof. Dr. Roberto Felicetti, Politecnico di Milano, Italy, is acknowledged for inspiring the Arduino-based application example. The technical support of Mr. Jørgen Holm, Electrical Workshop of Aarhus University, is also acknowledged.

References

[1] L. Bisby, J. Gales, C. Maluk, A contemporary review of large-scale non-standard structural fire testing, Fire Sci. Rev. 2 (1) (2013).

[2] J.-M. Franssen, T. Gernay, Modeling structures in fire with SAFIR: Theoretical background and capabilities, J. Struct. Fire Eng. 8 (3) (2017) 300–323.

[3] A. Sauca, N. Mortensen, A. Drustrup, G. Abbiati, Experimental validation of a hybrid fire testing framework based on dynamic relaxation, Fire Saf. J. 121 (2021) 103315.

[4] M. Nakashima, Hybrid simulation: An early history, Earthq. Eng. Struct. Dyn. 49 (10) (2020) 949–962.

[5] O.-S. Kwon, N. Nakata, A. Elnashai, B. Spencer, Technical note a framework for multi-site distributed simulation and application to complex structural systems, J. Earthq. Eng. 9 (5) (2005) 741–753.

[6] P. Pan, M. Tada, M. Nakashima, Online hybrid test by internet linkage of distributed test-analysis domains, Earthq. Eng. Struct. Dyn. 34 (11) (2005) 1407–1425.

[7] Y.-S. Yang, S.-H. Hsieh, K.-C. Tsai, S.-J. Wang, K.-J. Wang, W.-C. Cheng, C.-W. Hsu, ISEE: Internet-based simulation for earthquake engineering—Part I: Database approach, Earthq. Eng. Struct. Dyn. 36 (2007) 2291–2306.

[8] K.-J. Wang, K.-C. Tsai, S.-J. Wang, W.-C. Cheng, Y.-S. Yang, ISEE: Internet-based simulation for earthquake engineering—Part II: The application protocol approach, Earthq. Eng. Struct. Dyn. 36 (2007) 2307–2323.

[9] A. Schellenberg, S.A. Mahin, G.L. Fenves, A software framework for hybrid simulation of large structural systems, in: Structural Engineering Research Frontiers, American Society of Civil Engineers, Long Beach, California, United States, 2007, pp. 1–16.

[10] T.L. Karavasilis, C.Y. Seo, J. Ricles, HybridFEM: A Program for Dynamic Time History Analysis of 2D Inelastic Framed Structures and Real-Time Hybrid Simulation, Tech. Rep., ATLSS Center, Bethlehem, PA, 2008.

[11] I. Lamata Martinez, F.O. Santacana, M.S. Williams, A. Blakeborough, U.E. Dorka, Celestina-sim: Framework to support distributed testing and service integration in earthquake engineering, J. Comput. Civ. Eng. 30 (1) (2016) 04014119.

[12] X. Huang, O.-S. Kwon, A generalized numerical/experimental distributed simulation framework, J. Earthq. Eng. 24 (4) (2020) 682–703.

[13] M. Peroni, P. Pegon, F.J. Molina, P. Buchet, Ethernet-based servo-hydraulic real-time controller and DAQ at ELSA for large scale experiments, J. Earthq. Eng. (2021) 1–32.

[14] C.A. Whyte, K.R. Mackie, B. Stojadinovic, Hybrid simulation of thermomechanical structural response, J. Struct. Eng. 142 (2) (2016) 04015107.

[15] A. Sauca, C. Zhang, A. Chernovsky, M. Seif, Communication framework for hybrid fire testing: Developments and applications in virtual and real environments, Fire Saf. J. 111 (2020) 102937.

[16] X. Wang, R.E. Kim, O.-S. Kwon, I. Yeo, Hybrid simulation method for a structure subjected to fire and its application to a steel frame, J. Struct. Eng. 144 (8) (2018) 04018118.

[17] X. Wang, R.E. Kim, O.-S. Kwon, I.-H. Yeo, J.-K. Ahn, Continuous real-time hybrid simulation method for structures subject to fire, J. Struct. Eng. 145 (12) (2019) 04019152.

[18] ABAQUS Inc., ABAQUS, 2023, URL www.simulia.com.

[19] T. Blochwitz, M. Otter, J. Akesson, M. Arnold, C. Clauss, H. Elmqvist, M. Friedrich, A. Junghanns, J. Mauss, D. Neumerkel, H. Olsson, A. Viel, Functional mockup interface 2.0: The standard for tool independent exchange of simulation models, in: Linköping Electronic Conference Proceedings, Linköping University Electronic Press, 2012.

[20] C. Gomes, C. Thule, D. Broman, P.G. Larsen, H. Vangheluwe, Co-simulation: A survey, ACM Comput. Surv. 51 (3) (2018) 49:1–49:33.

[21] FMI 2.0, Functional mock-up interface for model exchange and Co-simulation, 2014, URL https://fmi-standard.org/downloads/.

[22] Modelica Association, Functional mock-up interface standard, website, 2022, URL https://fmi-standard.org/tools/.

[23] Modelisar Project, MODELISAR ITEA-2 research project, website, 2023, URL https://web.archive.org/web/20120328054431/http://www.modelisar.com/modelisar.html.

[24] C. Bertsch, E. Ahle, U. Schulmeister, The functional mockup interface-seen from an industrial perspective, in: Proceedings of the 10th International Modelica Conference; March 10–12; 2014; Lund; Sweden, (096) Linköping University Electronic Press, 2014, pp. 27–33.

[25] M. Aslan, U. Durak, K. Taylan, MOKA: An object-oriented framework for FMI Co-simulation, in: Conference on Summer Computer Simulation, Society for Computer Simulation International San Diego, CA, USA, Chicago, Illinois, 2015, pp. 1–8.

[26] C.M. Legaard, D. Tola, T. Schranz, H.D. Macedo, P.G. Larsen, A universal mechanism for implementing functional mock-up units, in: Proceedings of the 11th International Conference on Simulation and Modeling Methodologies, Technologies and Applications, SIMULTECH, July 7-9, 2021, Virtual Event, 2021 pp. 121–129.

[27] N. Battle, C. Thule, C. Gomes, H.D. Macedo, P.G. Larsen, Towards a static check of FMUs in VDM-SL, in: Formal Methods. FM 2019 International Workshops, in: Lecture Notes in Computer Science, vol. 12233, Springer International Publishing, Porto, Portugal, 2020, pp. 272–288.

[28] R. Kübler, W. Schiehlen, Two methods of simulator coupling, Math. Comput. Model. Dyn. Syst. 6 (2) (2000) 93–113.

[29] J. Bastian, C. Clauß, S. Wolf, P. Schneider, Master for Co-simulation using FMI, in: 8th International Modelica Conference, Linköping University Electronic Press, Linköpings universitet, Dresden, Germany, 2011, pp. 115–120.

[30] C. Gomes, C. Thule, J. DeAntoni, P.G. Larsen, H. Vangheluwe, Co-simulation: The past, future, and open challenges, in: Symposium on Leveraging Applications of Formal Methods, Verification and Validation, in: Lecture Notes in Computer Science, vol. 11246, Springer Verlag, Limassol, Cyprus, 2018.

[31] E. Mergny, G. Drion, J.-M. Franssen, Stability in hybrid fire testing using PI control, Exp. Tech. (2020).

[32] G. Abbiati, P. Covi, N. Tondini, O. Bursi, B. Stojadinovic, A real-time hybrid fire simulation method based on dynamic relaxation and partitioned time integration, ASCE J. Eng. Mech. (2020).

[33] C. Thule, K. Lausdahl, C. Gomes, G. Meisl, P.G. Larsen, Maestro: The INTO-CPS Co-simulation framework, Simul. Model. Pract. Theory 92 (2019) 45–61.

[34] M.S. Allen, D. Rixen, M. van der Seijs, P. Tiso, T. Abrahamsson, R.L. Mayes, Substructuring in Engineering Dynamics: Emerging Numerical and Experimental Techniques, CISM International Centre for Mechanical Sciences, vol. 594, Springer International Publishing, Cham, 2020.

[35] S. Renard, J.-C. Mindeguia, F. Robert, S. Morel, J.-M. Franssen, An adaptive controller for hybrid fire testing, Exp. Tech. (2020).

[36] F. Robert, S. Rimlinger, C. Collignon, Promethee: Fire resistance facility taking into account the surrounding structure, in: 1st International Workshop on Concrete Spalling Due To Fire Exposure, 2009.

[37] M. Korzen, G. Magonette, P. Buchet, Mechanical loading of columns in fire tests by means of the substructuring method, Z. Angew. Math. Mech. 79 (1999) S617–S618.

[38] E. Mergny, J.-M. Franssen, Multi-degree-of-freedom hybrid fire testing of a column in a non-linear environment, Fire Technol. (2022).

[39] A. Sauca, E. Mergny, T. Gernay, J.M. Franssen, A method for hybrid fire testing: Development, implementation and numerical application, in: M. Gillie, Y. Wang (Eds.), Applications of Fire Engineering, CRC Press, 2017, pp. 225–234.

[40] P. Schulthess, M. Neuenschwander, K.M. Mosalam, M. Knobloch, A computationally rigorous approach to hybrid fire testing, Comput. Struct. 238 (2020) 106301.

[41] M. Neuenschwander, P. Schulthess, K.M. Mosalam, M. Knobloch, Validating hybrid fire testing with full-physical twin experiments, Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci. 478 (2266) (2022) 20220180.

[42] C. Gomes, C. Thule, L. Lúcio, H. Vangheluwe, P.G. Larsen, Generation of Co-simulation algorithms subject to simulator contracts, in: J. Camara, M. Steffen (Eds.), Software Engineering and Formal Methods, in: Lecture Notes in Computer Science, vol. 12226, Springer International Publishing, Oslo, Norway, 2020, pp. 34–49.

[43] B.J. Oakes, C. Gomes, F.R. Holzinger, M. Benedikt, J. Denil, H. Vangheluwe, Hint-based configuration of Co-simulations with algebraic loops, in: Simulation and Modeling Methodologies, Technologies and Applications, vol. 1260, Springer International Publishing, Cham, 2021, pp. 1–28.

[44] C. Gomes, B.J. Oakes, M. Moradi, A.T. Gamiz, J.C. Mendo, S. Dutre, J. Denil, H. Vangheluwe, HintCO - hint-based configuration of Co-simulations, in: International Conference on Simulation and Modeling Methodologies, Technologies and Applications, Prague, Czech Republic, 2019, pp. 57–68.

[45] C. Gomes, L. Lucio, H. Vangheluwe, 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, IEEE, Munich, Germany, 2019, pp. 784–789.

[46] S.T. Hansen, C. Gomes, P.G. Larsen, J. Van de Pol, Synthesizing Co-simulation algorithms with step negotiation and algebraic loop handling, in: 2021 Annual Modeling and Simulation Conference, ANNSIM, IEEE, Fairfax, VA, USA, 2021, pp. 1–12.

[47] C. Thule, K. Lausdahl, C. Gomes, G. Meisl, P.G. Larsen, Maestro: The INTO-CPS Co-simulation framework, Simul. Model. Pract. Theory 92 (April) (2019) 45–61.

[48] L.I. Hatledal, A. Styve, G. Hovland, H. Zhang, A language and platform independent Co-simulation framework based on the functional mock-up interface, IEEE Access 7 (2019) 109328–109339.

[49] C. Andersson, J. Åkesson, C. Führer, Pyfmi: A Python Package for Simulation of Coupled Dynamic Models with the Functional Mock-Up Interface, Centre for Mathematical Sciences, Lund University Lund, 2016.

[50] J. Brembeck, A. Pfeiffer, M. Fleps-Dezasse, M. Otter, K. Wernersson, H. Elmqvist, Nonlinear state estimation with an extended FMI 2.0 Co-simulation interface, in: 10th International Modelica Conference, Linköping University Electronic Press; Linköpings universitet, Lund, Sweden, 2014, pp. 53–62.

[51] W.S. Levine, The Control Handbook: Control System Fundamentals, second ed., CRC Press, 2011.

[52] O.S. Bursi, P.-S. Shing, Evaluation of some implicit time-stepping algorithms for pseudodynamic tests, Earthq. Eng. Struct. Dyn. 25 (4) (1996) 333–355.

[53] G. Abbiati, E.E. Bas, Hybrid testing using arduino, project repository, 2022, URL https://www.hackster.io/461627/hybrid-testing-using-arduino-f0aea9.

[54] P. Underwood, Dynamic relaxation techniques: A review, Comput. Methods Transient Anal. (1983).

[55] N.M. Newmark, A method of computation for structural dynamics, J. Eng. Mech. Div. (3) (1959) 67–94.

[56] G.H. Golub, C.F. Van Loan, Matrix Computations, Fourth ed., Johns Hopkins studies in the mathematical sciences, The Johns Hopkins University Press, Baltimore, 2013.