

Article

A Physics-Informed Co-Simulation Framework for Resilience Assessment of Zonal Ship Central Cooling Systems

Xin Wu ¹, Ping Zhang ¹, Pan Su ^{1,*}, Wenshan Hu ² , Xianquan Zheng ¹, Bo Zhang ¹ and Jiechang Wu ¹¹ Naval University of Engineering, Wuhan 430033, China; 1208062011@nue.edu.cn (X.W.)² Wuhan University, Wuhan 430072, China

* Correspondence: 1920191101@nue.edu.cn

Abstract

In response to the challenges encountered in high-throughput resilience assessment of zonal ship central cooling systems, including numerical stiffness in physics-based dynamic models, abnormal solver termination, and insufficient continuity in batch simulation campaigns, a physics-informed co-simulation framework for resilience-oriented assessment is proposed. With control–physics orthogonal decoupling as its core, the framework separates the control-scheduling layer from the thermo-hydraulic solver at the software-execution level, while retaining information exchange through standardized interfaces. In addition, physics constraint-based pre-filtering, process-level fault isolation, and automatic recovery mechanisms are integrated to improve the robustness and continuity of automated batch assessment. A hierarchical reduced-order thermo-hydraulic model of the zonal ship central cooling system is established. Subsequently, the numerical stiffness characteristics of the fluid network and heat-transfer units under valve topology switching conditions are analyzed. A standalone C++ solver kernel is generated from a Simulink prototype model, and a Java/Web-based collaborative scheduling platform is constructed. Cross-environment consistency tests show that the C++ solver reproduces the Simulink prototype results under representative fast hydraulic and slow thermal scenarios, with steady-state and transient discrepancies below 0.05% and 1.08%, respectively. Physics constraint-based pre-filtering intercepted 42.6% of infeasible samples and reduced the total wall-clock runtime of the tested optimization task by approximately 38%. In 1000 fault-injection tests, the process-isolation mechanism isolated 12 abnormal solver terminations, while the main scheduling process remained alive and the remaining batch tasks were completed under the tested conditions. Finally, an abrupt pulse thermal-load increase in the forward zone was used as a representative scenario to demonstrate automatic extraction of temperature trajectories and quantitative evaluation using the cumulative temperature-exceedance severity (CTS) index. The results indicate that the proposed framework can support offline resilience-oriented assessment, reconfiguration-strategy screening, and batch evaluation of shipboard fluid–thermal systems under the tested conditions.



Academic Editor: Marek Markowski

Received: 7 May 2026

Revised: 4 July 2026

Accepted: 6 July 2026

Published: 10 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\) license](#).

Keywords: ship thermal management; zonal central cooling system; resilience assessment; co-simulation; process-level fault isolation

1. Introduction

With the development of naval platforms and large special-purpose ships toward integrated electrification [1–3], sustained attention has been devoted to the modeling, control, and optimization of ship thermal management and central cooling systems [4–6]. Owing to

its topology, characterized by physical partitioning and flexible interconnection, the zonal thermal management system is considered beneficial for enhancing functional retention and structural reconfiguration capability under localized damage conditions [7,8]. Such a system exhibits typical multi-scale time-constant characteristics and involves coupling between discrete configuration changes and continuous thermo-hydraulic dynamics; it is therefore essentially a complex discrete–continuous hybrid dynamic system [9–11]. In quantitative resilience assessment, high-throughput simulation based on massive random damage or compound fault injection provides a useful means of exploring the statistical characteristics of system survivability boundaries [7,12–15]. In this study, system resilience is defined as the capability to maintain critical cooling functions under fault or damage disturbances and to restore the thermal safety boundary through structural reconfiguration.

Existing studies have explored the optimization of reconfiguration strategies and physics-informed pruning algorithms for zonal cooling systems, thereby achieving, to some extent, a balance between optimization efficiency and thermal safety [7]. Building on this foundation, the present study extends our previous research [7], which established the benchmark ZSCCS configuration, the system-level modeling basis, and the cumulative temperature-exceedance severity (CTS) resilience metric. While [7] focused on algorithmic aspects of strategy optimization, the present study shifts the emphasis to the computational execution architecture required for robust, high-throughput simulation campaigns. This shift is motivated by the computational difficulties encountered when such assessments are executed at scale; in particular, fluid networks are subject to the stiffness constraints imposed by differential-algebraic equations (DAEs). During high-frequency topological reconfiguration, the instantaneous interruption of fluid inertia and the formation of local zero-flow branches may give rise to numerical issues such as violation of constraint consistency, ill-conditioning of the system Jacobian matrix, and solver failure [16–23]. Under traditional monolithic or tightly coupled simulation architectures, control logic and physical solving are typically implemented in a coupled manner within a unified execution environment [10,24]. Once severe numerical anomalies occur in the underlying physical solver, the execution of the main program may be further affected, thereby undermining the continuity of large-scale automated simulation campaigns [24–27].

Furthermore, the scientific validity of resilience assessment depends heavily on performance statistics obtained under a large number of extreme operating conditions [7,12–15]. The simulation framework therefore faces a pronounced trade-off between modeling fidelity and computational robustness. Although physics-based dynamic models with topology-switching representations are useful for capturing topological transient characteristics, they may also exhibit numerical stiffness [16–23]. Moreover, computational instability tends to occur predominantly in vulnerable operating intervals where the system approaches its physical limits [16–23]. If the simulation framework lacks an effective anomaly isolation mechanism and unconverged samples are therefore excluded from statistical analysis, non-random missingness may be introduced, thereby leading to statistical bias and creating a risk that the final resilience assessment may overestimate the actual survivability of the system [28,29]. Therefore, to improve the scalability and operational robustness of resilience-assessment tasks for ship fluid systems, it is useful to develop a co-simulation architecture with execution-level separation and fault-isolation capability [24–27,30].

Based on the above considerations, a physics-informed co-simulation framework for resilience assessment of zonal ship central cooling systems is proposed in this study. With orthogonal decoupling between the control layer and the physical layer as its core concept, the proposed framework enables the coordinated solution of control logic, structural reconfiguration, and continuous thermo-hydraulic processes under fault scenarios by

embedding physical feasibility constraints and anomaly management mechanisms into the decision-making simulation process. The main contributions of this study are as follows:

- (1) A hierarchical co-simulation architecture is constructed, and independent physical solver kernels are automatically generated, thereby realizing execution-level orthogonal decoupling between discrete control simulation and continuous physical solving;
- (2) Isolation and recovery mechanisms for numerical anomalies are designed, and a pre-filtering strategy based on physical constraints is integrated at the scheduling layer to improve the continuous execution capability and computational efficiency of high-throughput fault simulation campaigns;
- (3) An automated assessment workflow for the zonal shipboard central cooling system is established, supporting the extraction of temperature trajectories and resilience-related metrics from physics-based dynamic simulations, and providing computational support for offline reconfiguration-strategy comparison and related optimization studies.

2. Study Object and System Structure

In this study, a zonal ship central cooling system (ZSCCS) is considered as the research object, as shown in Figure 1. The system consists of three cooling zones, namely Zones I, II, and III. Each zone contains a seawater (SW) loop, a freshwater (FW) loop, and a central cooler for heat exchange between the two sides. Zones I and II primarily serve the main propulsion system and generator sets, whereas Zone III mainly serves dynamic loads such as electronic equipment. The zones are interconnected through FW supply and return mains and tie valves, enabling inter-zone cooling support while preserving zonal independence under normal operating conditions.

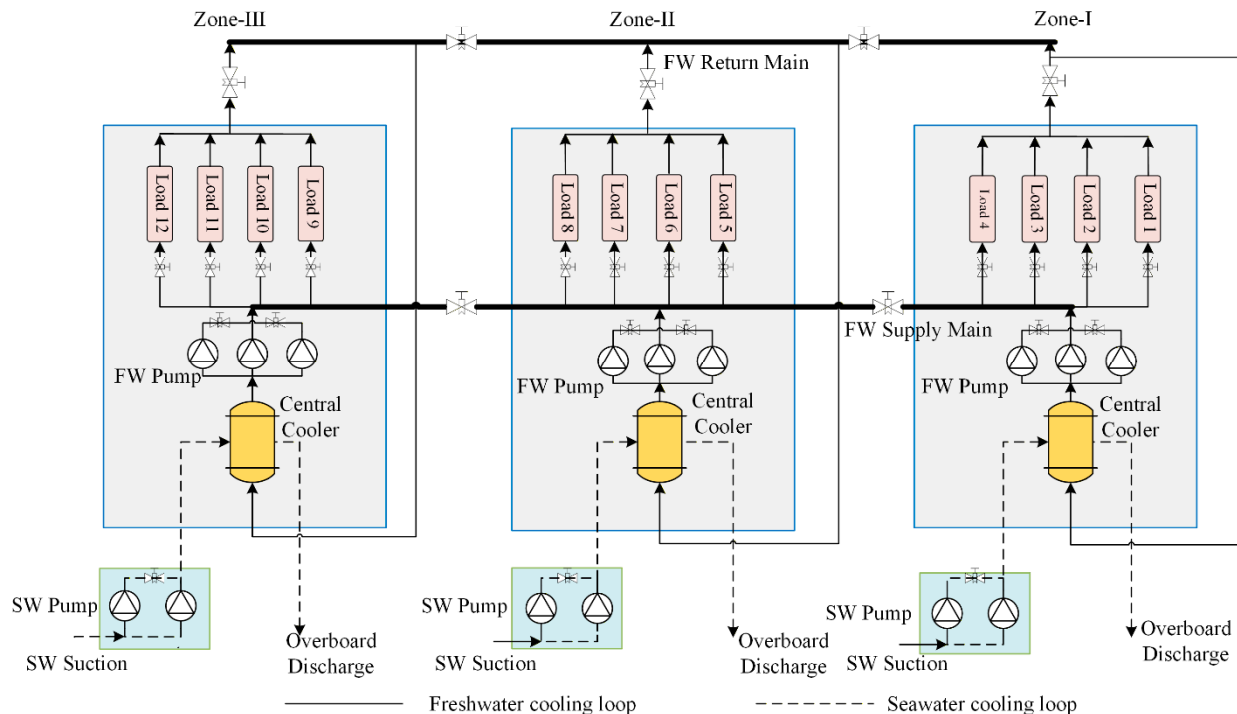


Figure 1. System-level schematic of the zonal ship central cooling system.

Figure 1 focuses on the low-temperature FW central cooling network involved in zonal reconfiguration. The high-temperature jacket-water circuits of the internal combustion engines are treated as equipment-side thermal loads and are not explicitly included in the reconfigurable network topology. For readability, only valves that participate in zonal

reconfiguration are shown. Local check valves, maintenance isolation valves, and pump-unit auxiliary valves are assumed to be integrated with the pump assemblies and are not treated as scheduling variables. The multiple FW pumps in each zone represent an N+1 duty/standby redundancy arrangement, and the active pump combination is determined by the prescribed operating scenario or reconfiguration command. These structural features make the ZSCCS a representative object for evaluating the proposed control–physics orthogonally decoupled co-simulation framework under topology-switching conditions.

3. System Modeling and Analysis of Numerical Characteristics

The ZSCCS is a typical discrete–continuous hybrid dynamical system with multiple time scales. The physical model established in this study constitutes the underlying solver kernel of the control–physics orthogonal decoupling (CPOD) co-simulation platform. It is formulated as a physics-based reduced-order lumped-parameter thermo-hydraulic model for capturing dominant system-level hydraulic and thermal transients associated with topology switching and load disturbances. The model is intended to reveal the numerical stiffness, Jacobian ill-conditioning, and singularity issues induced by high-frequency topological switching in zonal cooling networks, thereby providing a basis for the design of a robust co-simulation architecture.

3.1. Dynamic Topology-Coupled Modeling of the Fluid Network

The dynamic behavior of the fluid network is governed by mass and momentum conservation equations, while its specific transient response is determined by pipelines, valves, pumps, and flow-distribution components. To balance system-level modeling efficiency and fidelity in dynamic characteristics, the pipelines are represented as first-order dynamic elements incorporating fluid inertia and hydraulic damping. In addition, a discrete state variable is introduced to characterize the open/closed state of the valve. Following the lumped hydraulic-transient formulation commonly used in fluid-network modeling [18] and the differential-algebraic representation of constrained network systems [16,17], each branch is approximated as a first-order inertial-resistive element. The dynamic equation for the branch flow rate between nodes i and j can be expressed as follows:

$$\tau_{C,ij} \frac{dQ_{ij}(t)}{dt} + Q_{ij}(t) = \frac{\sigma_{ij}(t)}{R_{hyd,ij}} [P_i(t) - P_j(t)] \quad (1)$$

where $P_i(t)$ and $P_j(t)$ denote the nodal static pressures; $R_{hyd,ij}$ is the hydraulic resistance of the pipeline; and $\tau_{C,ij}$ is the fluid inertial time constant. Equation (1) is obtained from a first-order equivalent representation of the branch momentum balance under the assumptions of single-phase incompressible flow and lumped parameters.

Meanwhile, the pressure-flow state of the entire network must satisfy nodal mass conservation, namely the nodal continuity constraint. By stacking the branch flow rates into the vector, $\mathbf{Q}(t)$, the branch-wise momentum equations are coupled with the incidence matrix-based mass-conservation constraints of the network. By introducing the node-branch incidence matrix, $\mathbf{A}(\sigma)$, which is dynamically reconfigured with the topological state, $\sigma(t)$, the topological algebraic constraints of the system can be written as follows:

$$\mathbf{A}(\sigma)\mathbf{Q}(t) = \mathbf{b}(t) \quad (2)$$

where $\mathbf{A}(\sigma)$ is the node-branch incidence matrix varying with the network topology; $\mathbf{Q}(t)$ is the branch-flow column vector; and $\mathbf{b}(t)$ is the vector of external flow injections or extractions at the nodes.

By combining Equations (1) and (2), the fluid network can be described as a set of differential-algebraic equations (DAEs) driven by topological switching. Under high-

throughput fault injection or frequent valve-switching conditions, abrupt changes in the discrete variables cause transient variations in $\mathbf{A}(\sigma)$, thereby inducing phenomena such as local zero flow, degradation of network connectivity, and constraint reconfiguration. Such transient topological switching can significantly worsen the condition number of the system Jacobian matrix, causing conventional monolithic simulation architectures to face the risk of convergence failure or even time-step collapse.

3.2. Nonlinear Dynamic Coupling of Active Components

As the core power source of the ZSCCS, the pump set must be modeled in a manner that accounts for both the nonlinear head-flow characteristics and the transient hysteresis effects caused by mechanical inertia and fluid energy storage. To this end, an internal resistance–capacitance (RC)-equivalent element is introduced into the pump characteristic model to correct the additional hydraulic losses and dynamic energy storage at the pump–pipe coupling interface. The output pressure-rise model is given by the following:

$$\Delta P_{\text{out}} = f_{\text{HQ}}(Q_{\text{pump}}) - R_{\text{in}} \cdot Q_{\text{pump}} + \frac{1}{C_{\text{in}}} \int_0^t [Q_{\text{pump}}(\tau) - Q_{\text{out}}(\tau)] d\tau \quad (3)$$

where ΔP_{out} is the pump output pressure rise; $f_{\text{HQ}}(\cdot)$ is the nonlinear head function; Q_{pump} denotes the quasi-steady flow rate inferred from the pump head–flow characteristic under the current operating condition, whereas Q_{out} denotes the dynamically realized outlet flow rate after accounting for the lag introduced by the equivalent RC element; and R_{in} and C_{in} are equivalent parameters characterizing internal hydraulic losses and fluid energy storage. This RC equivalent element reflects the dynamic lag characteristics dominated by fluid inertia, internal holdup, and coupling feedback within the pump chamber and adjacent pipe sections. Under extreme fault conditions, the nonlinear term and integral feedback term in Equation (3) further intensify the time-scale separation of the system and enhance model stiffness, thereby imposing higher requirements on the numerical robustness of the underlying solver.

3.3. Numerically Robust Modeling of Dynamic Heat Transfer Units

In thermal management systems, the transient calculation of heat exchange components such as central coolers is a key factor affecting simulation stability. In particular, under extreme operating conditions involving low flow rates or near-zero flow rates, the conventional log mean temperature difference (LMTD) method is prone to degeneration of the thermodynamic transcendental equation, which may cause divergence in algebraic-loop iterations. To improve the numerical robustness of large-scale simulation, the arithmetic mean temperature difference (AMTD) method is adopted to reconstruct the dynamic energy balance equations:

$$M_{\text{h}} c_{p,\text{h}} \frac{dT_{\text{h,out}}}{dt} = \dot{m}_{\text{h}} c_{p,\text{h}} (T_{\text{h,in}} - T_{\text{h,out}}) - UA \Delta T_{\text{AMTD}} \quad (4a)$$

$$M_{\text{c}} c_{p,\text{c}} \frac{dT_{\text{c,out}}}{dt} = \dot{m}_{\text{c}} c_{p,\text{c}} (T_{\text{c,in}} - T_{\text{c,out}}) + UA \Delta T_{\text{AMTD}} \quad (4b)$$

where M_{h} and M_{c} are the equivalent masses of the fluid on the hot side and cold side, respectively; $c_{p,\text{h}}$ and $c_{p,\text{c}}$ are the constant-pressure specific heat capacities of the fluids on the two sides; $\dot{m}_{\text{h}}$ and $\dot{m}_{\text{c}}$ are the mass flow rates on the hot side and cold side; $T_{\text{h,in}}$ and $T_{\text{h,out}}$ are the inlet and outlet temperatures on the hot side; $T_{\text{c,in}}$ and $T_{\text{c,out}}$ are the inlet and outlet temperatures on the cold side; UA is the product of the overall heat transfer coefficient and heat transfer area; and ΔT_{AMTD} is the arithmetic mean temperature difference.

The arithmetic mean temperature difference, ΔT_{AMTD} , is defined as the arithmetic mean of the temperature differences at both ends of the heat exchanger:

$$\Delta T_{\text{AMTD}} = \frac{\Delta T_1 + \Delta T_2}{2} \quad (4c)$$

where ΔT_1 and ΔT_2 are determined by the specific flow arrangement of the heat exchanger.

The heat exchanger is modeled as reduced-order lumped control volumes on both sides, with outlet temperatures used as representative bulk temperatures. This system-level approximation aims to capture dominant thermal transients rather than local temperature distributions. The AMTD formulation is adopted to improve numerical robustness under low-flow conditions, where the logarithmic term in the LMTD formulation may become ill-conditioned. Accordingly, the AMTD-based cooler model is used as a system-level robust approximation, rather than as a detailed heat-exchanger design model over the full operating envelope.

3.4. Lumped-Parameter Modeling of Thermal Loads

For general heat-generating equipment in ship systems, such as diesel engines, electrical equipment, and auxiliary machinery, differentiated modeling strategies are adopted according to their physical characteristics. For diesel engines, whose internal heat transfer processes are complex, model order reduction is employed for appropriate simplification. By contrast, for electrical and auxiliary machinery with relatively uniform temperature distributions, a Lumped Parameter Model is adopted to describe the thermal dynamic behavior, and the equipment is equivalently treated as a cooled unit with concentrated thermal capacitance.

Taking cold-plate-cooled equipment as an example, the equipment body is regarded as an energy node with a thermal-capacitance buffering effect, and the transient heat balance equation is written as follows:

$$C_{\text{solid}} \frac{dT_{\text{solid}}}{dt} = \Phi_{\text{gen}} - hA(T_{\text{solid}} - T_{\text{fluid}}) \quad (5)$$

where C_{solid} is the equivalent thermal capacitance of the equipment body; Φ_{gen} is the heat generation power of the internal heat source; h is the convective heat transfer coefficient; A is the effective heat transfer area; and T_{solid} and T_{fluid} are the temperature of the solid body and the characteristic temperature of the cooling fluid, respectively.

In system-level analysis, this lumped-parameter model effectively preserves the thermal-capacitance buffering effect and provides a low-pass filtering characteristic for thermal dynamic evolution, thereby suppressing local stiffness induced by step thermal loads or abrupt boundary variations. The aforementioned physical submodels jointly constitute the unified underlying solver kernel and interact with the upper-level control system through the CPOD mechanism, thus supporting co-simulation analysis under complex fault conditions.

3.5. Model Implementation, Parameters, and Simulation Settings

Based on the hierarchical physics-based model of the ZSCCS, a standardized tag-based interface was developed to achieve an exact one-to-one mapping between the functional logic layer and the underlying C++ physical solver kernel. As shown in Figure 2, the underlying physics-based model was implemented in Simulink in a modular manner, where coupling among zonal modules was realized through uniformly named tag-based interfaces. This implementation also provides a model prototype for subsequent topology reconfiguration and for the generation of a standalone C++ solver kernel. In addition, the interface supports event-scheduled configuration updates during dynamic topology reconfiguration.

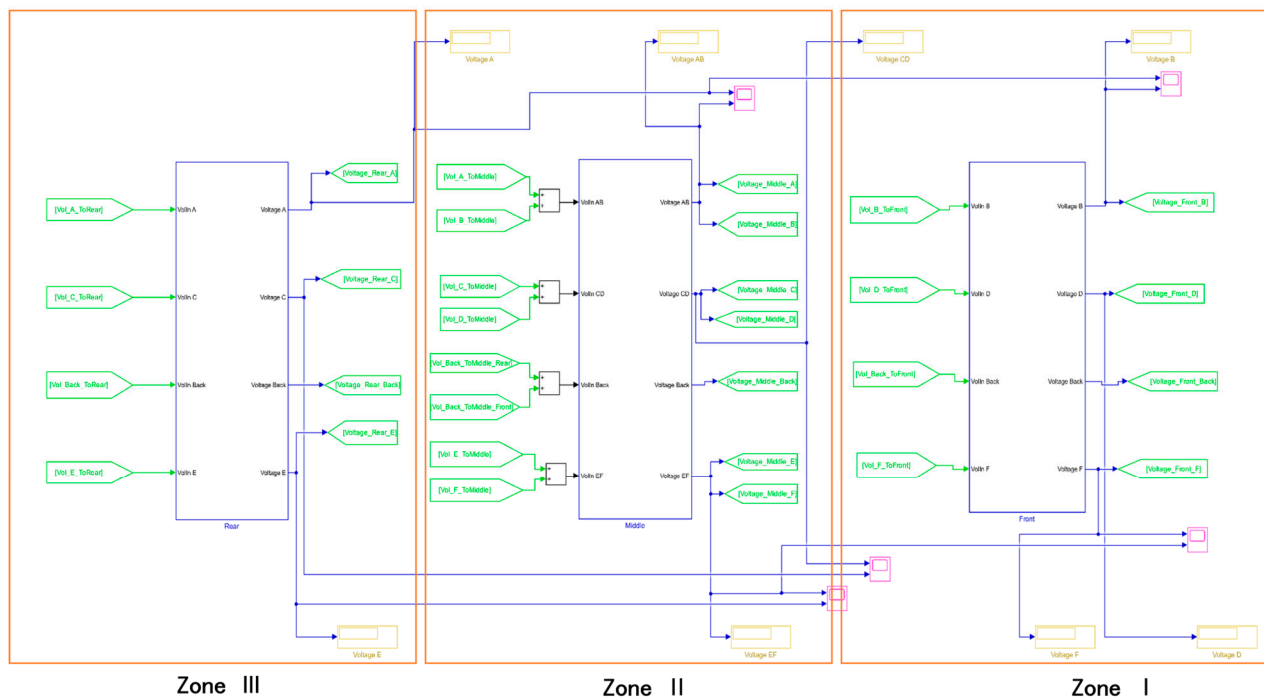


Figure 2. Modular Simulink implementation of the hierarchical physics-based model for the zonal ship central cooling system.

To evaluate the numerical robustness of the proposed platform, a representative ZSCCS was selected as the benchmark case. Its physical topology and major equipment thermal loads were constructed based on previous work [7]. Table 1 summarizes the key design parameters and boundary conditions of the low-temperature freshwater central cooling circuit considered in this study. These parameters were inherited from the benchmark configuration in Ref. [7] and calibrated according to ship design data and engineering specifications [31]. Considering that operations such as pump–valve switching may induce flow pulsations and multiscale coupling, a fixed-step integration scheme with a physical step size of 100 ms was adopted as the basic simulation setting of the platform. Event-triggered actions, such as pump startup, valve switching, and load variation, were aligned with the discrete simulation timeline and executed internally by the solver according to the input schedule. Within the range considered in this study, this time step provided a satisfactory compromise between the accuracy of capturing key transient characteristics and the overall computational efficiency.

Table 1. Key design and operating parameters of the benchmark zonal ship central cooling system used in the platform test case.

Parameter	Unit	Zones I and II	Zone III
Central cooler design parameters			
Heat transfer area	m ²	1297.0	145.1
Overall heat transfer coefficient	W/(m ² ·K)	3500	3500
Design heat load	kW	25,539.2	2879.6
Freshwater inlet/outlet temperature	°C	41.76/36.0	41.75/36.0
Freshwater mass flow rate	kg/s	1062.7	119.8
Seawater inlet/outlet temperature	°C	32.0/34.0	32.0/34.0
Seawater mass flow rate	kg/s	3047.6	343.6

Table 1. *Cont.*

Parameter	Unit	Zones I and II	Zone III
Major equipment heat loads			
Main propulsion diesel engine (per engine)	kW	9000	-
Diesel generator (per generator)	kW	1200	1200
Gearbox cooler (per cooler)	kW	360	-
Auxiliary machinery cooler (per set)	kW	408	36
Electrical equipment cooler (per set)	kW	136	16

4. Co-Simulation Architecture and Implementation Methodology Based on Orthogonal Decoupling

The ZSCCS is characterized by a multilevel valve network and strong fluid–thermal coupling. Conventional simulation methods are challenged in simultaneously achieving high computational efficiency and flexibility in control strategy implementation. To address the challenges of multiphysics coupling and high computational load in large-scale resilience analysis, a co-simulation architecture based on CPOD is proposed. Under the separation-of-concerns design philosophy, the continuous-time solution of physical dynamics and the discrete-time scheduling of control tasks are separated at the execution level while remaining functionally connected through standardized data exchange. Furthermore, an automated workflow from mathematical models to high-performance executable code is established based on the model-driven development paradigm.

4.1. Overall Technical Architecture

In response to the complex fluid-network topology of the system and the nonlinear evolution of faults, a CPOD architecture was constructed, as shown in Figure 3. In this context, orthogonal decoupling indicates that the control logic and physical evolution remain informationally coupled at the functional level, while being isolated at the execution level in terms of memory space, process lifecycle, and exception propagation paths. This design confines solver-side numerical abnormalities within independent computational processes, thereby providing fault isolation and recovery capability for large-scale automated simulation tasks. The architecture is divided into a physical simulation kernel layer implemented in C++ and a co-simulation application layer implemented in Java and Web-based technologies. The former is responsible for the time integration of the physics-based thermo-hydraulic model, which may become stiff under topology-switching conditions, whereas the latter is responsible for system-level orchestration and human–machine interaction, supporting topology visualization and configuration, status monitoring, and batch JSON task generation.

In the present implementation, the control variables include binary valve states, pump start/stop commands, component availability states, fault-injection times, load-variation events, and scheduled reconfiguration actions. These variables are encoded in the task description and passed to the physical solver together with the initial thermo-hydraulic states and boundary conditions. The solver returns time histories of flow rates, temperatures, actuator states, and resilience-related quantities for subsequent metric extraction.

The current platform is primarily intended for offline batch assessment of predefined or optimization-generated event sequences. During a simulation run, event-triggered actions are applied by the C++ solver according to the input schedule. Continuous step-by-step feedback between the Java scheduling layer and the physical solver is not performed in the present implementation; incorporating such feedback would require additional state exchange during numerical integration. Accordingly, the present implementation is best suited to offline task-level assessment, whereas continuously feedback-driven reconfiguration would require stepwise synchronization between the scheduling layer and the solver.

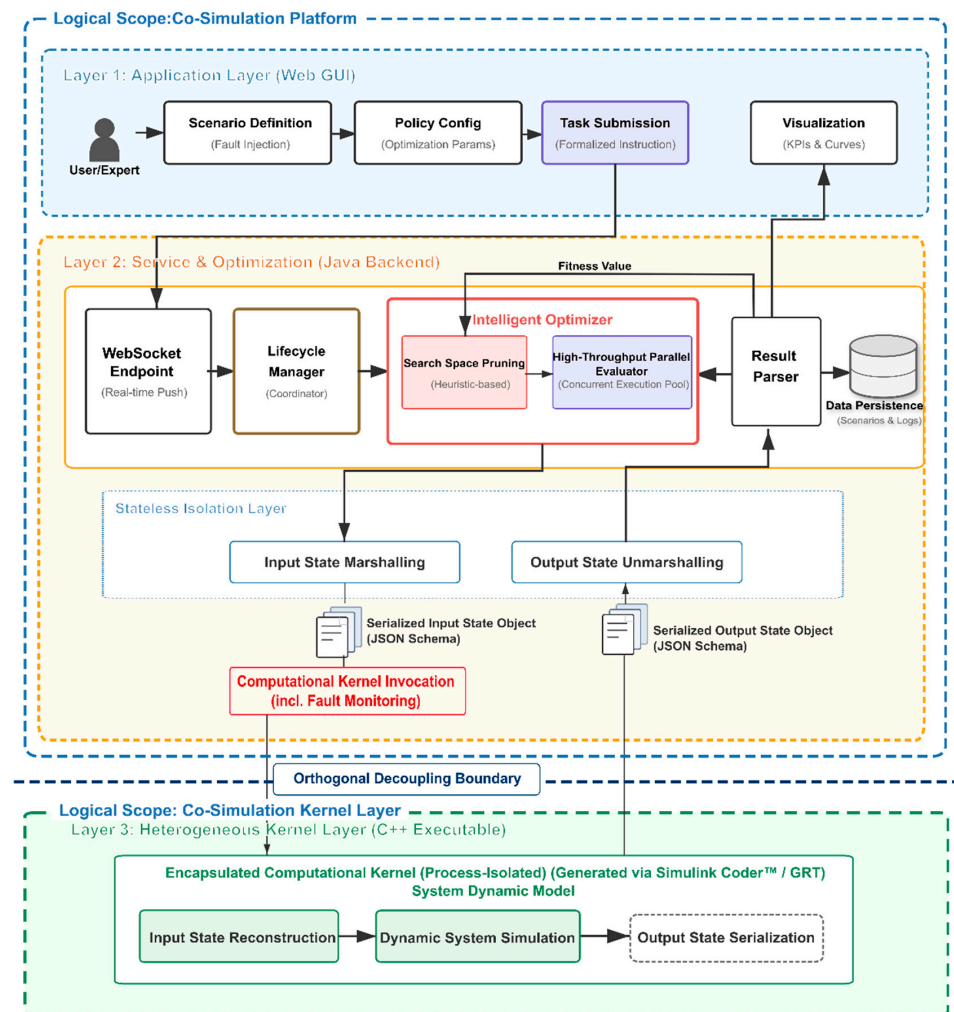


Figure 3. Overall architecture of the CPOD-based co-simulation platform for the zonal ship central cooling system.

4.2. Key Implementation Technologies

To support the stable operation of the CPOD architecture under fault-injection and topology-switching conditions, emphasis was placed on heterogeneous model interoperability and fault isolation. At the physical kernel layer, a model-solver separation strategy was adopted. Boundary conditions were first standardized into an input/output vector space through causality analysis. Subsequently, standalone C++ source code was generated using Simulink Coder R2022a and compiled into native machine-code executables independent of the MATLAB environment, without introducing an additional custom model-translation tool. In this manner, dependency on the MATLAB/Simulink runtime environment was removed, and lightweight computational support was provided for parallel batch-evaluation tasks.

For the interaction mechanism, an operating system-based process-level isolation scheme was designed, as shown in Figure 4. The Java main program launches each C++ solver instance as an independent child process through asynchronous system calls, while a watchdog thread monitors its execution status. The two parts communicate through standardized JSON files for task initialization and result retrieval. Before each simulation run, the Java layer serializes the task configuration, including topology, boundary conditions, control commands, fault events, load-variation events, and simulation settings, into an input file. The C++ solver reconstructs the simulation state, advances the physical model over the prescribed time horizon, and writes the computed trajectories and status information to an output file.

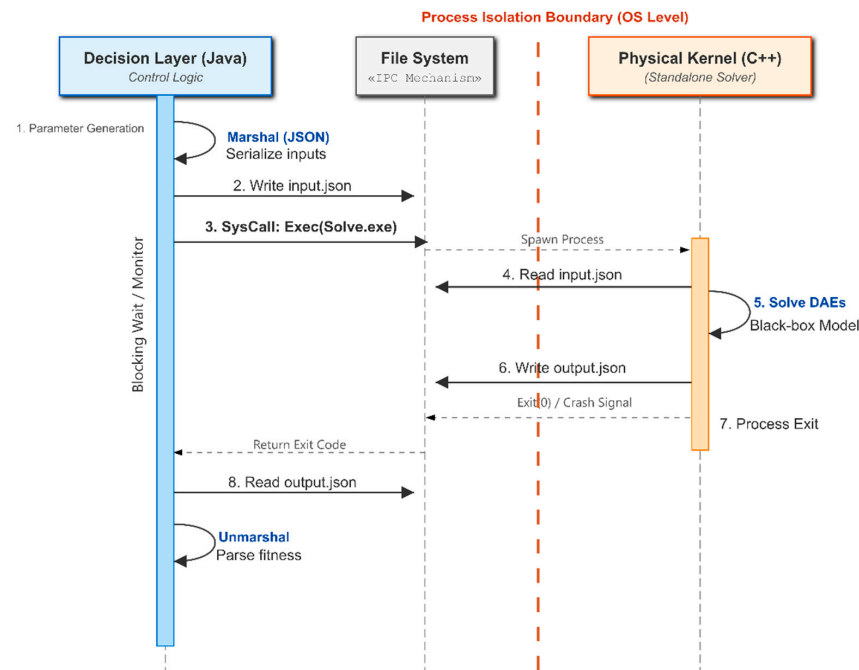


Figure 4. Task-level interaction and process-isolation mechanism between the Java scheduling layer and the standalone C++ physical solver. Arrows indicate the task execution and JSON-based data-exchange sequence; the red dashed line denotes the operating-system-level process-isolation boundary; and different colors distinguish the Java scheduling layer, file system, and C++ physical solver.

Through this loose-coupling mechanism, a computational sandbox is established. The interaction is performed at the task level rather than at every integration step. In this study, the physical integration step is 100 ms, and event-triggered actions such as pump startup, valve switching, and load variation are executed internally by the solver according to the input schedule. Once abnormal termination of a child process is detected, the watchdog records the exit status, releases occupied resources, marks the corresponding task as unsuccessful, and continues with subsequent tasks according to the batch schedule. The current implementation does not roll back to an intermediate simulation state within a failed run. Thus, improved robustness is achieved with limited I/O overhead, thereby supporting batch-task continuity under the tested conditions.

4.3. Physics-Informed Optimization and Simulation Workflow

A genetic algorithm (GA), developed in the previous study [7], was integrated into the platform as an optimization-oriented task generator. In the present work, the GA is used to generate candidate reconfiguration schedules for batch evaluation, rather than to represent a complete onboard real-time control optimizer. The decision vector, $\mathbf{x}$, includes discrete pump operating states, isolation-valve and tie-valve states, component-availability states, load-assignment decisions, and the timing of inter-zone support commands. The optimization objective is formulated as follows:

$$J(\mathbf{x}) = \text{CTS}(\mathbf{x}) + P_{\text{topo}}(\mathbf{x}) + P_{\text{cap}}(\mathbf{x}) + P_{\text{fail}}(\mathbf{x}) + P_{\text{sw}}(\mathbf{x}) \quad (6)$$

where $\text{CTS}(\mathbf{x})$ denotes the cooling-transition score, and $P_{\text{topo}}(\mathbf{x})$, $P_{\text{cap}}(\mathbf{x})$, $P_{\text{fail}}(\mathbf{x})$, and $P_{\text{sw}}(\mathbf{x})$ penalize topological infeasibility, source-load capacity mismatch, solver failure, and unnecessary switching actions, respectively. The constraints include binary actuator bounds, supply-return reachability, source-load capacity compatibility, structural mass-balance compatibility, actuator timing limits, and the thermal-safety threshold used in CTS evaluation. By leveraging the isolation properties of CPD, an iterative batch-evaluation

workflow in which filtering precedes simulation was established, as shown in Figure 5. Its core component is a physics-informed pruning filter. For each candidate configuration, the Java layer constructs a topology graph by removing closed valves and failed branches. Before invoking the C++ solver, the filter checks supply-return reachability, rated source-load capacity compatibility, structural mass-balance feasibility based on graph connectivity and boundary-condition consistency, and equipment-limit admissibility. Rejected samples are assigned penalized fitness values, whereas borderline but physically possible cases are retained for dynamic simulation. In this way, the pre-filter reduces evidently infeasible evaluations without excluding severe but physically admissible operating cases. Only valid samples are dispatched to the C++ kernel for dynamic solution, after which system resilience metrics are evaluated and returned.

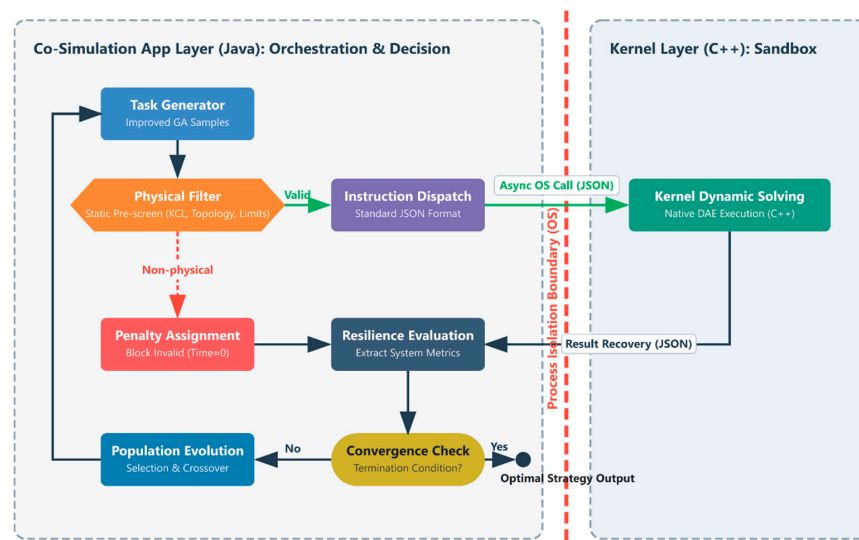


Figure 5. Iterative batch-evaluation workflow with physics-constraint pre-filtering based on the CPOD architecture. Arrows indicate the data flow and iterative evaluation sequence; the dashed red line denotes the boundary between the co-simulation application layer and the C++ kernel layer; and colors distinguish different functional modules.

5. Solver Consistency Verification and Performance Evaluation of the Co-Simulation Platform

In this section, the proposed platform is examined from four aspects: cross-environment solver consistency, computational scheduling efficiency, process-level robustness, and automated assessment capability. The comparison between the standalone C++ solver kernel and the Simulink prototype is used as a code-generation and deployment consistency verification, rather than as an independent physical validation of the thermo-hydraulic model. Therefore, the numerical consistency of the standalone C++ solver is first assessed through cross-comparison under representative operating conditions. Subsequently, the improvements in computational efficiency and operational stability achieved by the physics-constraint pre-filtering and process-isolation mechanisms are evaluated. Finally, the end-to-end automated capability of the platform for dynamic simulation, trajectory extraction, and resilience-oriented metric calculation is demonstrated through a case study involving a pulse-like thermal-load surge in the forward ship zone.

5.1. Cross-Environment Solver Consistency Verification

This subsection focuses on the cross-environment consistency between the standalone C++ solver kernel and the Simulink prototype model. The benchmark configuration, model basis, and selected dynamic-response characteristics were described in a previous work [7].

On this basis, the numerical consistency of the automatically generated standalone C++ solver kernel is assessed after deployment outside the MATLAB/Simulink environment, with emphasis on code-generation and deployment consistency. Two representative scenarios are selected for consistency verification: regional seawater pump N + 1 switching, which represents fast hydraulic dynamics, and a sudden increase in regional thermal load, which represents slow thermodynamic dynamics.

Figure 6a shows the variation in the total system seawater flow rate during the pump-switching process. The black solid line represents the results obtained from the Simulink prototype model, whereas the red dashed line represents those obtained from the standalone C++ solver kernel. The two curves exhibit a high degree of overlap, indicating that the standalone solver kernel can accurately capture the key high-frequency dynamic processes, including transient flow-rate drop, standby pump startup, and hydraulic re-stabilization. Figure 6b presents the evolution trajectory of the regional outlet temperature under a sudden thermal-load increase fault. During this slow dynamic process characterized by a large time constant, the results of the Simulink reference model and the standalone C++ solver kernel remain highly consistent in both the thermal shock response stage and the thermodynamic re-equilibration stage, with no observable integration drift.

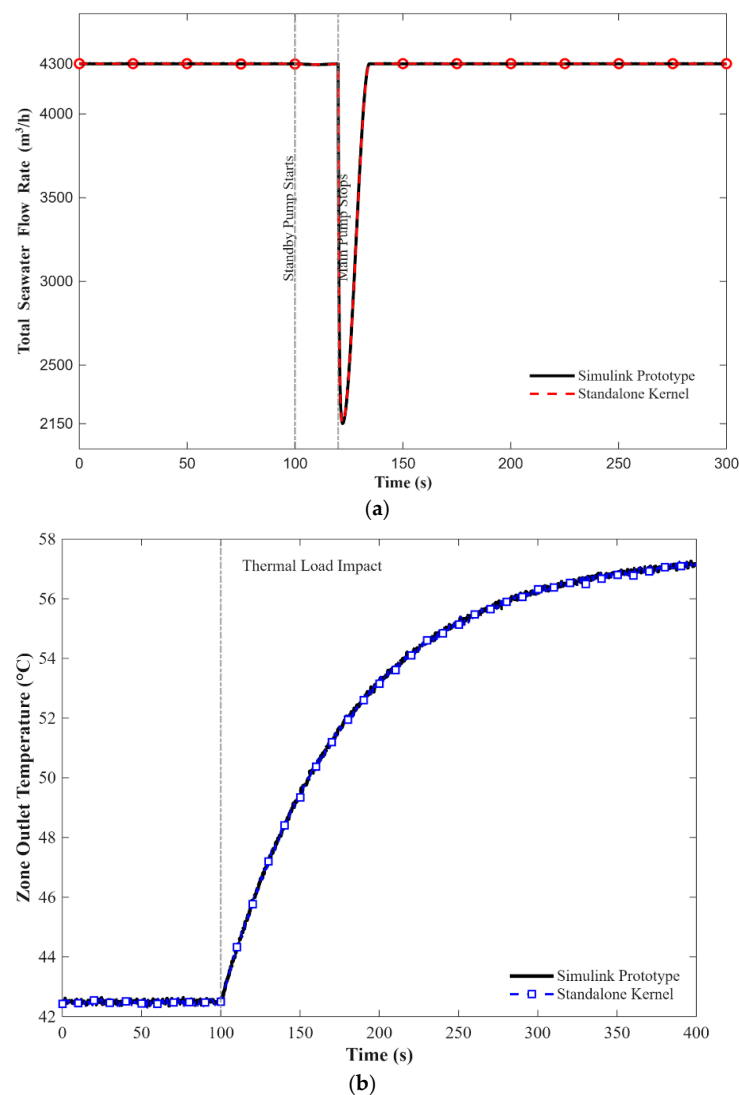


Figure 6. Comparison of dynamic responses between the standalone C++ solver kernel and the Simulink prototype model under representative switching and fault scenarios. (a) Dynamic response consistency during N + 1 pump switching. (b) Temperature-response consistency under a thermal fault.

A quantitative error analysis of the key state variables is provided in Table 2. Under rated steady-state conditions, the relative errors in both hydraulic and thermal state variables are less than 0.05%. During transient evolution, the maximum relative error is controlled within 1.08%, mainly appearing in derived indicators such as the maximum temperature rise rate and the flow-rate recovery stabilization time. These minor deviations are primarily attributed to differences in the underlying numerical integration strategies across environments, namely the discrepancy between the native variable-step integrator in Simulink and the fixed-step iterative strategy used in the standalone C++ solver kernel, rather than to any distortion of the physical model.

Table 2. Quantitative analysis of cross-environment calculation errors for key state variables.

Test Scenarios	Key Variables	Simulink Prototype Model	Standalone Solver Kernel	Maximum Relative Error (%)
Steady-state	System steady-state total flow rate (m ³ /h)	4300.0	4299.8	<0.01%
	Regional steady-state outlet temperature (°C)	42.50	42.51	0.02%
Seawater pump N + 1 switching	Minimum flow rate during switching (m ³ /h)	2150.4	2153.6	0.15%
	Flow-rate recovery stabilization time (s)	12.5	12.6	0.80%
Thermal load surge fault	Peak temperature during fault evolution (°C)	57.35	57.42	0.12%
	Maximum temperature rise rate (°C/s)	0.185	0.187	1.08%

The above results demonstrate that the standalone C++ solver kernel developed in this study can reproduce the numerical behavior of the Simulink prototype under the tested scenarios. This cross-environment consistency supports subsequent large-scale batch evaluation of reconfiguration strategies outside the MATLAB/Simulink environment.

5.2. Computational Efficiency and Process-Isolation Robustness Tests

In this subsection, the computational scheduling performance and underlying operational stability of the proposed co-simulation architecture for large-scale resilience analysis tasks are evaluated. Two groups of tests were designed, one to examine the effectiveness of integrating physics-prior rules into the scheduling framework, and the other to examine the behavior of the process-isolation mechanism under extreme operating conditions.

5.2.1. Physics-Constraint Pre-Filtering and Computational Scheduling Efficiency

To address the high computational cost of physics-based simulation in high-dimensional optimization-oriented batch evaluation, the physics-constraint pre-filter proposed in previous work [7] was integrated into the scheduling layer of the platform. Table 3 summarizes the settings and outcomes of this optimization-oriented batch-evaluation task, including the GA configuration, candidate-sample screening results, computing environment, solver settings, and runtime-profiling results. Runtime was measured as the wall-clock time required to complete the full batch-evaluation task with eight parallel workers, including scheduling, JSON-based data exchange, process invocation, physical simulation, result parsing, and, where applicable, pre-filtering. The standalone C++ solver used fixed-step integration with a 100 ms physical step; thus, variable-step relative and absolute tolerances were not applicable.

Table 3. Computational setup and optimization-oriented batch-evaluation outcomes for the GA-based experiment.

Category	Parameter	Value	Unit/Remark
Samples	Generated candidate samples	1000	cases
Samples	Samples dispatched to the C++ solver	574	cases
Samples	Samples intercepted by the pre-filter	426	cases, 42.6%
Genetic algorithm (GA) settings	Population size	50	—
Genetic algorithm (GA) settings	Evaluated generations	20	—
Genetic algorithm (GA) settings	Crossover probability	0.80	—
Genetic algorithm (GA) settings	Mutation probability	0.05	—
Computing environment	CPU	Intel Core i7-14700	—
Computing environment	RAM	16	GB, DDR4 3200 MHz
Computing environment	Operating system	Windows 11	64-bit
Computing environment	Parallel workers	8	processes
Solver settings	Simulation horizon	250	s
Solver settings	Physical integration step	100	ms
Solver settings	Integration steps per simulation	2500	steps
Solver settings	Integration scheme	Fixed-step	—
Runtime profiling	Total runtime without pre-filter	68.4	s
Runtime profiling	Total runtime with pre-filter	42.4	s
Runtime profiling	Runtime reduction	38.0	%
Runtime profiling	Mean C++ solver time per dispatched case	0.512	s/case
Runtime profiling	Mean pre-filter time per generated case	1.8	ms/case
Runtime profiling	Mean process-spawn overhead	34	ms/call
Runtime profiling	Abnormal-process recovery time	≈100	ms

Table 3 reports not only runtime settings but also the optimization-oriented batch-evaluation outcomes, including the number of GA-generated candidates, pre-filtered infeasible candidates, simulations dispatched to the C++ solver, and the resulting runtime reduction. Under this configuration, the GA generated 1000 candidate samples for the optimization-oriented evaluation. These candidate schedules were evaluated by the combined pre-filtering and C++ simulation workflow rather than by runtime profiling alone. The pre-filter intercepted 426 candidates before physical simulation, reducing the number of samples dispatched to the C++ solver to 574. The total wall-clock runtime decreased from 68.4 s without pre-filtering to 42.4 s with pre-filtering, corresponding to a runtime reduction of 38.0%. The mean pre-filtering time was 1.8 ms per generated case, whereas the mean C++ solver time was 0.512 s per dispatched case. The mean process-spawn overhead for one C++ solver invocation was approximately 34 ms. These results show that the optimization-oriented batch-evaluation workflow reduced the number of expensive physical simulations with only a limited screening overhead.

The intercepted samples were mainly caused by two types of conflicts with physical rules, as shown in Figure 7a. The first was topological inaccessibility, accounting for approximately 28.4%, such as cooling-water flow interruption caused by the closure of multiple isolation valves. The second was structural mass-balance infeasibility, accounting for approximately 14.2%, such as isolated source nodes, sink nodes without return paths, or conflicting boundary-flow specifications within disconnected subnetworks. These cases were identified from graph topology and boundary-condition compatibility before hydraulic solving; they are structural infeasibilities rather than nodal mass-balance residuals obtained after hydraulic solving. By performing these conservative checks before simulation execution, clearly invalid samples were prevented from entering the time-consuming dynamic solution stage.

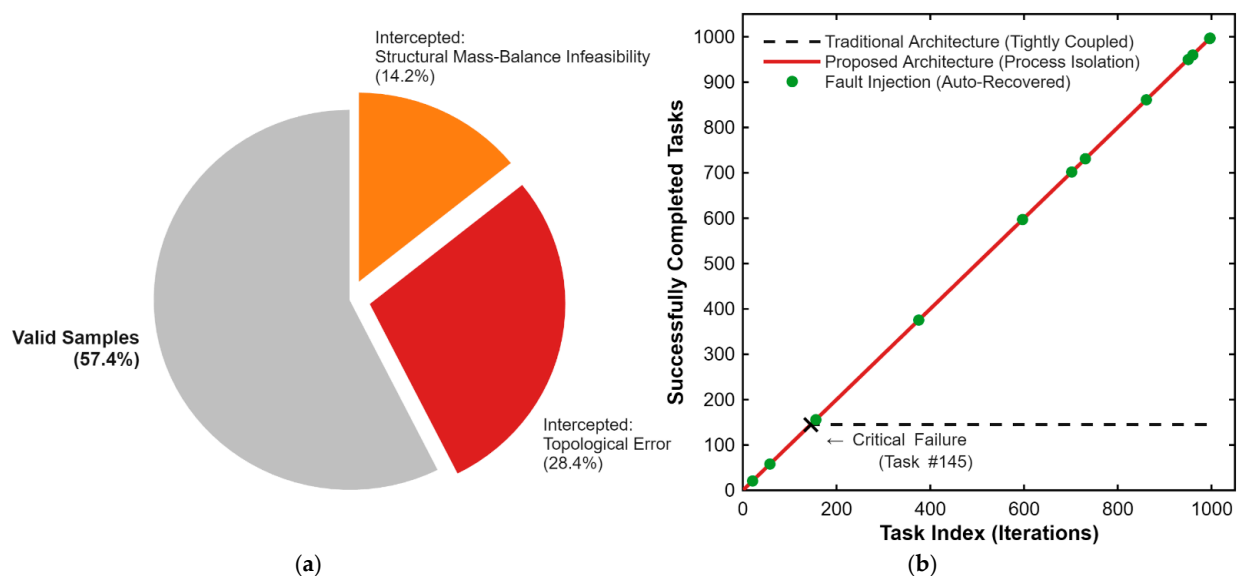


Figure 7. Evaluation of platform computational scheduling efficiency and underlying robustness. (a) Invalid samples intercepted by the filter. (b) Architecture robustness under fault injection.

It should be noted that Table 3 reports the computational outcome of an optimization-oriented batch-evaluation task, not only runtime parameters. The GA-based task generated candidate reconfiguration schedules, the physics-informed pre-filter removed infeasible candidates, and the remaining candidates were evaluated by the C++ solver. The results demonstrate the computational efficiency and operational continuity of the CPOD-based batch-evaluation framework under this workload. They do not constitute a comprehensive comparison of all possible reconfiguration strategies or a final onboard optimization policy. Broader multi-scenario strategy optimization, including additional damage patterns and resilience indicators, will be investigated in future work.

5.2.2. Stability Verification of the Process-Isolation Mechanism Under Extreme Conditions

To evaluate the long-term stability of the platform under high-load conditions, 1000 random fault-injection tests involving 12 groups of extreme boundary conditions were conducted, as shown in Figure 7b. Under such extreme conditions, complex fluid networks are prone to problems such as non-convergence of algebraic loops and memory overflow.

Comparative tests show that, in conventional tightly coupled architectures, such as direct Java Native Interface (JNI) invocation or MATLAB shared-memory interaction, severe exceptions may cause the main program to crash, thereby interrupting the entire batch of tasks, as observed in Task #145. In contrast, in the proposed platform, the process-isolation mechanism executes the solver in an independent child process. When Task #145 triggered an exception, the main control program was able to capture the exit code, reclaim resources, and restart the computational node within approximately 100 ms. The failed solver instance was recorded in the execution log and marked as an unsuccessful evaluation; in optimization-oriented runs, such a case was assigned a penalized fitness value rather than being silently discarded or counted as a successful physical response. As reported in Table 3, the mean process-spawn overhead during normal solver invocation was approximately 34 ms per call, which was small compared with the mean physical-solver time of 0.512 s per dispatched case. Across all tests, the platform successfully isolated 12 potential crashes, preserved batch-task continuity under the tested conditions, and completed the remaining tasks. These results indicate that the orthogonal decoupling design of the CPOD architecture can effectively prevent cascading fault propagation and provides the operational robustness required for unattended batch-evaluation applications.

5.3. Demonstration of the Platform's Automated Assessment Capability

Based on the cross-environment solver consistency and the physics-based dynamic simulation capability of the platform, this subsection further examines its functions for automatic metric extraction and quantitative evaluation. A pulse-like thermal-load surge in the forward zone was selected as a representative scenario. In the platform configuration module, the cooling-water outlet temperature of the forward zone was selected as the observation variable, and the safety threshold, the thermal-conduction time constant, and the electromechanical response time constant of the interzonal valve were configured in the underlying simulation logic.

At the platform interaction layer, the local self-recovery strategy without interzonal support (B1) and the platform-level interzonal coordinated strategy (B2) were encapsulated as standardized control commands and transmitted to the simulation kernel through the interface. After command parsing, the underlying thermodynamic model was driven by the kernel to carry out dynamic simulation. As shown in Figure 8, the simulation system represents both the heat-conduction process and the system temperature response trajectory (Figure 8a), as well as the hierarchical nonlinear actuation sequence of actuators with significant inertia (Figure 8b). In Figure 8b, the load surge is detected at 50 s, the local standby pump is started at 52 s, and the interzonal support command is triggered at 65 s. The interconnection valve then follows the configured electromechanical response time constant rather than changing instantaneously. Therefore, Figure 8b shows the transition from local recovery to interzonal support and explains why the thermal response continues to evolve after the control command has been issued.

After the simulation was completed, the automatic metric-extraction module performed a quantitative analysis of the temperature trajectory. The cumulative temperature exceedance severity (CTS) metric, previously defined in [7], was adopted to characterize the cumulative temperature exceedance relative to the safety threshold, and it was used as one of the proxy resilience indicators for critical thermal-function degradation during the fault period. It is expressed as follows:

$$CTS = \int_{t_0}^{t_1} \max(T(t) - T_{\text{safe}}, 0) dt \quad (7)$$

where $T_{\text{safe}} = 46.0^\circ\text{C}$ is the safety threshold for the cooling-water outlet temperature of the forward zone. This metric jointly reflects the magnitude and duration of temperature exceedance; a larger value indicates a higher thermal safety risk.

As shown in Figure 8, under strategy B1, the load surge was detected by the system at 50 s, and the standby pump in the local zone was started at 52 s after a delay of 2 s. In Figure 8b, the left axis represents the local standby FW pump speed, whereas the right axis represents the cross-cabin/interzonal valve opening. Owing to the limited cooling support capacity within a single zone, the temperature continued to rise and reached a peak of 56.7°C , corresponding to a CTS value of $1572.0^\circ\text{C}\cdot\text{s}$. Under strategy B2, local self-recovery was first executed during 52–65 s, and the command to open the interconnection valve of the adjacent midship zone was subsequently triggered at 65 s. Due to the combined effects of valve mechanical inertia and thermal-conduction delay shown in Figure 8b, the temperature continued to increase smoothly after 65 s and reached a peak of approximately 47.8°C at about 77.3 s. With interzonal coordination, the CTS value was reduced to $46.0^\circ\text{C}\cdot\text{s}$.

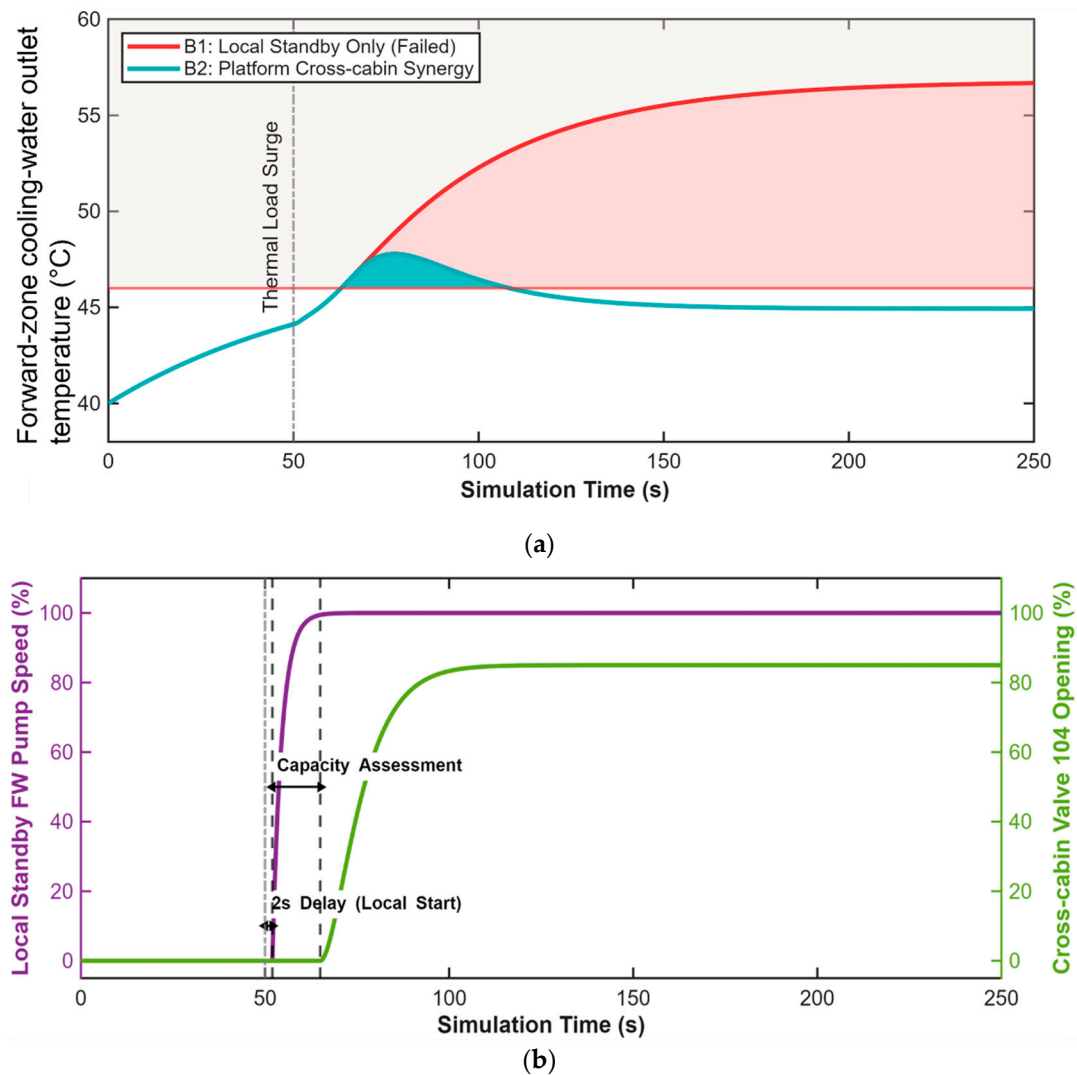


Figure 8. Automated evaluation results of the platform under a pulse-like thermal-load surge scenario in the forward zone: (a) temperature response and CTS reduction; and (b) hierarchical actuator dynamics from local recovery to interzonal support, where the left axis denotes local standby FW pump speed and the right axis denotes cross-cabin/interzonal valve opening. The shaded regions in (a) indicate cumulative temperature exceedance used for CTS calculation, and the vertical dashed lines in (b) mark key event times.

These results demonstrate that effective decoupling was achieved between the underlying multiphysics evolution and the upper-level automated software evaluation. For large shipboard equipment systems, significant thermal inertia and actuator action delays are common. Under such conditions, conventional simplified evaluation methods based on steady-state assumptions or step-response approximations are often inadequate. By contrast, the proposed platform can effectively capture complex dynamic features such as nonlinear temperature escalation and delayed peak occurrence. Users are only required to configure the scheduling sequence and inertial parameters at the front end, after which the platform can automatically complete the process from physics-based reduced-order dynamic simulation to the quantitative calculation of resilience-related metrics such as CTS. This provides a computational evaluation basis for resilience assessment and strategy optimization of shipboard fluid–thermal systems.

6. Conclusions

To address the difficulty of reconciling physics-based reduced-order thermo-hydraulic solving with robust computational scheduling in high-throughput resilience assessment of the ZSCCS, a physics-informed co-simulation framework, termed CPOD, is proposed in this study. Centered on the orthogonal decoupling of the control layer and the physical layer, the framework enables the coordinated solution of control logic, structural reconfiguration, and continuous thermo-hydraulic processes under fault scenarios by embedding physical-feasibility constraints and an exception self-recovery mechanism. The main conclusions are summarized as follows:

- (1) The automatically generated standalone C++ solver kernel was able to reproduce the numerical outputs of the Simulink prototype under the tested scenarios. This result is interpreted as cross-environment solver consistency verification and code-generation/deployment consistency verification, rather than as independent physical validation of the thermo-hydraulic model. Under typical operating conditions, including N+1 seawater-pump switchover and abrupt zonal thermal-load surges, the steady-state relative errors of key state variables were below 0.05%, while the maximum transient relative error was maintained within 1.08%. Therefore, model support was provided for large-scale self-reconfiguration testing independent of commercial software platforms.
- (2) By integrating a physical-constraint pre-filter at the scheduling layer, 42.6% of invalid samples were intercepted in advance, including 28.4% due to topological inaccessibility and 14.2% due to structural mass-balance infeasibility. In the tested batch-evaluation task, the total wall-clock runtime was reduced from 68.4 s to 42.4 s, corresponding to a reduction of approximately 38.0%. This decreased the overhead associated with ineffective evaluations in the high-dimensional optimization process.
- (3) In 1000 random fault-injection tests involving 12 groups of extreme boundary conditions, 12 potential crashes were successfully isolated by virtue of the process-isolation and exception-recovery mechanisms. Consequently, batch-task continuity was preserved under the tested conditions, and the interference of numerical anomalies with the large-scale resilience-assessment workflow was mitigated.
- (4) Using an abrupt pulse-load surge in the forward zone as a representative demonstration scenario, end-to-end automated metric extraction was demonstrated, spanning from physics-based dynamic simulation to resilience-related metrics such as CTS. Moreover, dynamic characteristics such as actuator hysteresis and heat-transfer delay could be represented. In particular, the cross-cabin coordinated strategy reduced the CTS from 1572.0 °C·s to 46.0 °C·s, thereby demonstrating the capability of the platform to quantitatively evaluate the effectiveness of reconfiguration strategies.

In summary, the proposed CPOD co-simulation framework provides computational support for offline resilience-oriented batch assessment of zonal ship cooling networks under topology-switching conditions. It is intended for offline resilience evaluation, reconfiguration-strategy screening, and redundancy-configuration studies, rather than for direct real-time onboard control deployment in its present form. Broader validation based on additional ship-design data, experimental or operational data, and more diverse fault or damage scenarios remains necessary. Future work will focus on extending the framework to broader fault scenarios, additional resilience indicators, and feedback-based reconfiguration decision-making with stepwise state exchange.

Author Contributions: Conceptualization, X.W. and P.S.; methodology, X.W. and P.Z.; validation, X.W., P.S., W.H., X.Z. and J.W.; resources, P.S. and J.W.; data curation, X.W.; writing—original draft preparation, X.W.; writing—review and editing, P.Z., P.S., W.H., X.Z., B.Z. and J.W.; supervision, P.Z.

and P.S.; project administration, P.Z.; funding acquisition, P.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research and the APC were funded by the Independent Project of the Naval University of Engineering (2023502080).

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Aboelezz, A.M.; Sedhom, B.E.; El-Saadawi, M.M.; Eladl, A.A.; Siano, P. State-of-the-Art Review on Shipboard Microgrids: Architecture, Control, Management, Protection, and Future Perspectives. *Smart Cities* **2023**, *6*, 1435–1484. [\[CrossRef\]](#)
2. McCoy, T.J. Trends in Ship Electric Propulsion. In *Proceedings of the IEEE Power Engineering Society Summer Meeting*; IEEE: New York, NY, USA, 2002; pp. 243–346. [\[CrossRef\]](#)
3. Doerry, N.; Amy, J.V.; Krolick, C. History and the Status of Electric Ship Propulsion, Integrated Power Systems, and Future Trends in the U.S. Navy. *Proc. IEEE* **2015**, *103*, 2243–2251. [\[CrossRef\]](#)
4. Fang, C.; Cai, B.; Ma, S.; Li, B. Simulation and Experimental Study on Flow and Heat Transfer of Ship Central Cooling System. *MATEC Web Conf.* **2018**, *213*, 04001. [\[CrossRef\]](#)
5. Jeon, T.Y.; Jung, B.G. A Study of PI Controller Tuning Methods Using the Internal Model Control Guide for a Ship Central Cooling System as a Multi-Input, Single-Output System. *J. Mar. Sci. Eng.* **2023**, *11*, 2025. [\[CrossRef\]](#)
6. Miao, Z.; Zhang, Y.; Wang, S. Modeling and decoupling control of marine central cooling water systems under variable load conditions. *Ocean Eng.* **2021**, *234*, 109258. [\[CrossRef\]](#)
7. Wu, X.; Zhang, P.; Su, P.; Wu, J.; Yuchen, L. Physics-Informed Dynamic Resilience Assessment and Reconfiguration Strategy for Zonal Ship Central Cooling Systems. *J. Mar. Sci. Eng.* **2026**, *14*, 598. [\[CrossRef\]](#)
8. Li, J.; Liu, F.; Chen, Y.; Shao, C.; Wang, G.; Hou, Y.; Mei, S. Resilience Control of DC Shipboard Power Systems. *IEEE Trans. Power Syst.* **2018**, *33*, 6675–6685. [\[CrossRef\]](#)
9. Henzinger, T.A. The Theory of Hybrid Automata. In *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS'96)*; IEEE Computer Society: Washington, DC, USA, 1996; pp. 278–292.
10. Cellier, F.E.; Kofman, E. *Continuous System Simulation*; Springer: New York, NY, USA, 2006. [\[CrossRef\]](#)
11. Lee, E.A.; Seshia, S.A. *Introduction to Embedded Systems: A Cyber-Physical Systems Approach*, 2nd ed.; MIT Press: Cambridge, MA, USA, 2017.
12. Santos, T.A.; Guedes Soares, C. Monte Carlo Simulation of Damaged Ship Survivability. *Proc. Inst. Mech. Eng. Part M. J. Eng. Marit. Environ.* **2005**, *219*, 25–35. [\[CrossRef\]](#)
13. O'Connor, P.D.T.; Kleyner, A. *Practical Reliability Engineering*, 6th ed.; Wiley: Hoboken, NJ, USA, 2021.
14. Modarres, M. *Risk Analysis in Engineering: Techniques, Tools, and Trends*; CRC Press: Boca Raton, FL, USA, 2006.
15. Hosseini, S.; Barker, K.; Ramirez-Marquez, J.E. A review of definitions and measures of system resilience. *Reliab. Eng. Syst. Saf.* **2016**, *145*, 47–61. [\[CrossRef\]](#)
16. Brenan, K.E.; Campbell, S.L.; Petzold, L.R. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*; SIAM: Philadelphia, PA, USA, 1996. [\[CrossRef\]](#)
17. Hairer, E.; Wanner, G. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 1996. [\[CrossRef\]](#)
18. Chaudhry, M.H. *Applied Hydraulic Transients*, 3rd ed.; Springer: New York, NY, USA, 2014. [\[CrossRef\]](#)
19. Pantelides, C.C. The Consistent Initialization of Differential-Algebraic Systems. *SIAM J. Sci. Stat. Comput.* **1988**, *9*, 213–231. [\[CrossRef\]](#)
20. Mattsson, S.E.; Söderlind, G. Index Reduction in Differential-Algebraic Equations Using Dummy Derivatives. *SIAM J. Sci. Comput.* **1993**, *14*, 677–692. [\[CrossRef\]](#)
21. Plouraboué, F. Review on water-hammer waves mechanical and theoretical foundations. *Eur. J. Mech.-B/Fluids* **2024**, *108*, 237–271. [\[CrossRef\]](#)
22. Zimmer, D. Towards Hard Real-Time Simulation of Complex Fluid Networks. In *Proceedings of the 13th International Modelica Conference*; Linköping University Electronic Press: Regensburg, Germany, 4–6 March 2019; pp. 579–586. [\[CrossRef\]](#)
23. Xiao, S.; Praditia, T.; Oladyshkin, S.; Nowak, W. Global sensitivity analysis of a CaO/Ca(OH)₂ thermochemical energy storage model for parametric effect analysis. *Appl. Energy* **2021**, *285*, 116456. [\[CrossRef\]](#)
24. Gomes, C.; Thule, C.; Broman, D.; Larsen, P.G.; Vangheluwe, H. Co-Simulation: A Survey. *ACM Comput. Surv.* **2018**, *51*, 49. [\[CrossRef\]](#)

25. Blochwitz, T.; Otter, M.; Arnold, M.; Bausch, C.; Clauss, C.; Elmqvist, H.; Junghanns, A.; Mauss, J.; Monteiro, M.; Neidhold, T.; et al. The Functional Mockup Interface for Tool Independent Exchange of Simulation Models. In *Proceedings of the 8th International Modelica Conference*; LiU Electronic Press: Linköping, Sweden, 2011; pp. 105–114. [[CrossRef](#)]
26. Arnold, M.; Clauß, C.; Schierz, T. Error Analysis and Error Estimates for Co-Simulation in FMI for Model Exchange and Co-Simulation v2.0. *Arch. Mech. Eng.* **2013**, *60*, 75–94. [[CrossRef](#)]
27. Gomes, C.; Thule, C.; Deantoni, J.; Larsen, P.G.; Vangheluwe, H. Co-simulation: The Past, Future, and Open Challenges. In *Leveraging Applications of Formal Methods, Verification and Validation. Distributed Systems*; Springer: Cham, Switzerland, 2018; pp. 504–520. [[CrossRef](#)]
28. Little, R.J.A.; Rubin, D.B. *Statistical Analysis with Missing Data*, 2nd ed.; Wiley: Hoboken, NJ, USA, 2002.
29. Montewka, J.; Manderbacka, T.; Ruponen, P.; Tompuri, M.; Gil, M.; Hirdaris, S. Accident susceptibility index for a passenger ship—A framework and case study. *Reliab. Eng. Syst. Saf.* **2022**, *218*, 108145. [[CrossRef](#)]
30. Huerta, J.R.; Silva, R.S.; De Tomi, G.; da Silva, A.L.M.A. A dynamic simulation approach to support operational decision-making in underground mining. *Simul. Model. Pract. Theory* **2022**, *115*, 102458. [[CrossRef](#)]
31. Huang, H. *Practical Handbook of Ship Design: Marine Engineering Volume*, 3rd ed.; National Defense Industry Press: Beijing, China, 2013.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.