Array 30 (2026) 100738

Contents lists available at ScienceDirect

Array

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

HEDA-HF: A formally verified hybrid edge–cloud digital twin architecture for heart failure management

Mohamed Ramdani [a], Teodoro Montanaro [b], Yousra Ben Aissa [a], Luigi Patrono [b,*]

[a] University of Biskra, Algeria
[b] University of Salento, Italy

ARTICLE INFO

Keywords:
Digital twin
Heart failure
Edge–cloud architecture
Formal verification
Model checking
Real-time systems
Safety-critical healthcare

ABSTRACT

Heart failure (HF) is a prevalent life-threatening chronic condition requiring continuous, patient-specific management. Digital twin (DT) technology offers real-time patient state modeling and predictive decision support. However, current HF DT frameworks lack formal guarantees of safety, timing, reliability, and provide limited coordination between edge devices and cloud analytics-Undermining clinical trust and deployment. We present HEDA-HF, a formally verified hybrid edge–cloud DT architecture for HF management. HEDA-HF provides design-time mathematical guarantees that all safety, liveness, and timing requirements hold across every modeled execution scenario. Every data-driven inference, alert, and synchronization event is verified at design-time against rigorously defined temporal properties before influencing patient care at the deployment time, ensuring strict adherence to safety and timing constraints. HEDA-HF explicitly embeds formal verification in the DT pipeline. The architecture is modeled as a network of timed automata and exhaustively verified in UPPAAL against Computation Tree Logic (CTL) and Timed CTL specifications. We validate HEDA-HF across five canonical HF-monitoring scenarios, confirming all functional properties across the full reachable state space. Furthermore, statistical model checking shows that critical alerts meet clinically acceptable deadlines with probability above 0.99%. By integrating mathematically proven guarantees at design time, HEDA-HF establishes a robust foundation for trustworthy clinically dependable HF DTs.

1. Introduction

Heart failure (HF) is one of the most widespread and critical public health challenges, currently affecting over 64.3 million people globally [1]. HF is a progressive clinical syndrome characterized by the heart's inability to maintain adequate cardiac output to meet the body's metabolic demands. Common symptoms include fatigue, breathlessness, edema, arrhythmias, and sudden weight gain—reflecting fluid retention and declining cardiac function [2,3].

The disease often follows a nonlinear trajectory, with periods of compensation interrupted by acute decompensation episodes. These events may escalate rapidly and require timely intervention to prevent irreversible damage or hospitalization. Subtle physiological signals — such as elevated heart rate, reduced oxygen saturation, or minor changes in body weight — can precede clinical deterioration by hours to days, but are often missed by conventional care practices.

Despite advances in pharmacotherapy and device-based interventions, HF continues to cause high hospital readmission rates, significant mortality, and escalating healthcare expenditures [4]. This burden is compounded by HF's dynamic and unpredictable trajectory, where subtle physiological changes can escalate into acute decompensation within hours. Traditional management strategies — based on periodic follow-ups, symptom reporting and delayed interventions — often fail to detect deterioration early enough to prevent hospitalization [5]. This reactive approach contributes to the persistently high morbidity and highlights the need for continuous, real-time monitoring and decision support tailored to each patient's evolving condition [6]. Indeed, its progressive and multifactorial nature requires continuous monitoring, proactive risk stratification, and adaptive therapy adjustment. In the long term, the telemonitoring may lead to early detection of clinical deterioration and early interventions that exceed the limits of traditional care workflows and reduce hospitalization [7].

In this context, digital twin (DT) technologies offer a promising paradigm shift for managing chronic and dynamic conditions such as HF. A digital twin in healthcare functions as a continuously synchronized virtual model of an individual patient, integrating multimodal data from wearable sensors, electronic health records (EHRs), and contextual sources to represent the patient's physiological state in near real time [8–10]. Unlike traditional static models, DTs enable proactive

* Corresponding author.
E-mail address: luigi.patrono@unisalento.it (L. Patrono).

https://doi.org/10.1016/j.array.2026.100738
Received 11 December 2025; Received in revised form 26 February 2026; Accepted 1 March 2026
Available online 24 March 2026

monitoring by anticipating physiological deviations and forecasting adverse events based on both mechanistic and data-driven insights. In HF care, DTs can capture and simulate evolving cardiac function, identify early warning signs of decompensation, and support clinician decision-making through patient-specific predictive modeling. By shifting from reactive interventions to predictive, personalized management, digital twins have the potential to reduce hospitalizations, optimize therapy timing, and improve long-term outcomes.

Recent developments in digital health and artificial intelligence have introduced new opportunities for proactive, data-driven HF management. Indeed, recent studies highlight an explosion of interest in digital twin technologies across domains [11], underlining their promise for real-time modeling and decision support. Among these innovations, *digital twins* (DTs) have emerged as a promising paradigm for real-time patient state representation and predictive decision support [12].

In the context of HF, a digital twin offers a powerful mechanism to track patient status continuously, simulate physiological evolution, and anticipate adverse outcomes before they manifest. By enabling personalized, dynamic models of cardiac health, DTs can bridge the gap between episodic care and the need for constant vigilance in chronic disease management. DT continuously integrates multimodal physiological, clinical, and contextual data from sources such as wearable sensors, imaging systems, and electronic health records (EHRs) [13]. In cardiovascular applications, DTs enable in silico therapy testing, early decompensation detection, and personalized outcome modeling [14]. For example, Gu et al. [15] developed patient-specific twins for HF prognosis, while Koopsen et al. [16] implemented virtual pacing simulations to optimize cardiac resynchronization therapy—demonstrating the potential of DTs to support clinical decision-making through real-time virtual companions.

Despite these advances, the deployment of clinically reliable and scalable DTs for HF remains limited. Most existing frameworks operate primarily in offline simulation mode, lacking closed-loop continuity between edge devices (e.g., wearables, home monitors) and cloud-based analytics [17,18]. This separation restricts responsiveness and real-time adaptability. Furthermore, current DT architectures often lack modular design principles and fail to support runtime feedback or adaptive control—capabilities essential for managing chronic HF [19,20]. More critically, *none of the current HF DT solutions incorporate formal verification mechanisms* to guarantee that safety and timing constraints hold under all possible execution scenarios [21,22]. By formal verification mechanisms, we refer to mathematically rigorous techniques — such as model checking — that exhaustively analyze all possible system behaviors against specified logical and temporal properties [23]. These techniques allow us to prove, with certainty, that critical safety rules (e.g., "an alert is never missed") and timing guarantees (e.g., "alerts are issued within 5 s") always hold during operation, even in the presence of uncertainty or system delays. In a safety-critical domain like HF, where erroneous or delayed decisions can have life-threatening consequences, the absence of provable correctness poses major barriers to clinical adoption and regulatory approval. Formal verification introduces a level of rigor and predictability that is essential to build clinician trust, ensure patient safety, and support future certification pathways for autonomous digital health systems.

Motivated by these gaps in the state of the art, we propose **HEDA-HF**, a *Formally Verified Hybrid Edge–Cloud DT Architecture* for HF management. The formal guarantees provided by HEDA-HF are enforced through model checking during the system design phase. They are not enforced via runtime filters but embedded into the system logic, ensuring that runtime behavior complies with pre-verified properties. HEDA-HF is explicitly designed to (i) enable real-time sensing, analysis, and feedback through tightly coordinated edge–cloud processing, and (ii) embed formal verification as a new-class architectural layer based on model checking of timed automata [24,25].

In HEDA-HF, the core *operations subject to verification* include all safety-critical actions that occur during patient monitoring and system coordination—such as data acquisition, edge inference, cloud alert generation, clinician acknowledgment, and edge–cloud synchronization. Each of these operations is represented in the formal model as a transition or event in a network of timed automata. By design, every data-driven inference, alert, or synchronization event is automatically checked against rigorously defined *temporal logic properties* describing safety (nothing unsafe ever happens), liveness (desired events eventually occur), and timeliness (all actions complete within clinical deadlines) before it can influence patient care. This formal modeling ensures that the system not only behaves as expected in test cases, but across every possible scenario defined by its logical structure and timing constraints.

For this purpose, HEDA-HF is formally modeled using timed automata and verified with UPPAAL — a well-established formal verification toolchain — during the design phase. This enables exhaustive verification of Computation Tree Logic (CTL) and Timed CTL (TCTL) properties, as well as statistical model checking under uncertainty [26].

This new approach directly addresses the shortcomings of prior DT frameworks that relied solely on empirical tests or simulations, enabling continuous assurance that clinical operations are provably correct across all modeled execution paths. In summary, this paper makes the following key contributions:

(i) **Novel DT Architecture with Integrated Verification:** We propose *HEDA-HF*, a hybrid edge–cloud DT architecture that introduces a dedicated *Formal Verification Layer (FVL)*. This layer ensures that every sensing, inference, alerting, and synchronization operation undergoes logical and temporal safety checks, distinguishing HEDA-HF from prior HF DT frameworks that lacked provable assurance.

(ii) **Provably Correct Behavior Baseline:** HEDA-HF achieves complete satisfaction of a rigorously defined set of CTL/TCTL properties across the entire reachable state space of the formal system model. This establishes a new formally proven correctness baseline for an HF digital twin, ensuring that key safety conditions (e.g., no missed alerts, no contradictory states) and liveness/timeliness requirements are always met.

(iii) **Quantified Reliability via Statistical Verification:** Using UPPAAL Statistical Model Checking, we quantify HEDA-HF's reliability under uncertainty, obtaining probabilistic guarantees for essential service-level objectives. For instance, the architecture ensures — with probability above 99% — that clinical alerts are delivered within the prescribed time bounds, failover mechanisms restore operation within deadlines, and no deadlocks or data losses occur even under stochastic conditions.

(iv) **Validated Clinical Use-Case Scenarios:** We formalize and verify five canonical HF care scenarios — routine monitoring, early risk alerting, clinician-in-the-loop adjustment, edge-device failover recovery, and cloud synchronization conflict resolution — to demonstrate the architecture's operational trustworthiness across a spectrum of realistic HF management situations.

The remainder of this paper is structured as follows. Section 2 discusses background and related work. Section 3 introduces the HEDA-HF architecture, detailing its layered design and how formal verification is embedded into the edge–cloud workflow. Section 4 describes the formal modeling of HEDA-HF and our verification approach using timed automata and model checking. Section 5 presents the verification results and performance evaluation, demonstrating both exhaustive property checks and statistical reliability analysis. Section 6 discusses the implications of our findings, including architectural insights, clinical relevance, and limitations. Finally, Section 7 concludes the paper and outlines future research directions.

2. Related work

2.1. Digital twins in heart failure and healthcare

Digital twins (DTs) are virtual patient models that integrate real-time physiological data with predictive computational simulations to support continuous monitoring, prognosis, and personalized therapy [8,27–29]. Originally developed in engineering and manufacturing, DTs have since been applied in domains such as smart industrial IoT [30–32]. In healthcare, DTs are increasingly used to simulate treatment strategies and predict health trajectories. Notably, blockchain-enhanced Internet of Medical Things (IoMT) frameworks have been explored for early anomaly detection in patient data [33], underscoring the growing role of DT concepts in digital health. In cardiovascular medicine — and particularly in heart failure (HF) — DTs promise to bridge continuous sensing with adaptive management [34]. They are envisioned to mirror cardiac physiology in real time, providing predictive insights for therapy optimization and timely intervention.

Recent research has advanced DTs for cardiac modeling, risk prediction, and therapy personalization. Gu et al. [15] developed patient-specific cardiovascular twins using multimodal data from 343 HF patients, achieving improved prognostic accuracy and model interpretability compared to standard clinical risk scores. Trayanova et al. [35] reviewed multiscale electrophysiological models demonstrating how patient-specific heart simulations can predict arrhythmic risk and guide optimal therapies such as pacing or ablation [14]. Similarly, Koopsen et al. [16] created a twin of the heart to simulate biventricular pacing in HF patients, successfully predicting individual responses to cardiac resynchronization therapy. Other efforts have combined wearable sensor streams with closed-loop cardiovascular simulations to anticipate HF decompensation events, reflecting a growing synergy between mechanistic modeling and data-driven analytics in cardiology.

Despite these advances, current cardiac DT systems largely remain cloud-dependent, operating as offline or periodic simulations with limited real-time coupling to patients. They offer no formal guarantees of safety, timing, or correctness during continuous operation. Critically, no existing HF DT has been formally verified to ensure that: (i) patient-specific alert thresholds are correctly and consistently applied under all conditions; (ii) strict timing constraints (e.g., delivering an alert within a 3-minute window of detecting deterioration) are always met; (iii) system behavior remains fail-safe during network outages or edge–cloud disconnections; or (iv) predictive accuracy and safety properties hold across the physiological variability and sensor noise represented in the model.

Coorey et al. [36] confirmed that most cardiovascular DTs are essentially static digital snapshots updated only periodically, rather than functioning as real-time, continuously coupled replicas. Extending this concept, Iyer et al. [37] introduced "TwinCardio", a digital-twin-based platform for continuous cardiac monitoring using on-body sensors and a specialized neural network ("TwinNet") for HF classification and prediction. However, even such advanced prototypes incorporate no formal safeguards or mathematically verifiable mechanisms to guarantee safe operation under asynchronous conditions or component failures. In short, while HF DTs demonstrate strong modeling and predictive capabilities, they lack formal assurances of behavioral correctness and timing safety in live clinical deployments.

2.2. Formal verification in medical cyber–physical systems

Formal verification techniques, such as model checking and temporal logic reasoning, provide a mathematical foundation for establishing system correctness and have achieved widespread success in safety-critical cyber–physical domains [38–40]. In the medical device arena, formal methods have begun to prove their value. For instance, Chen et al. [41] demonstrated the use of symbolic model checking to verify closed-loop control logic in an intensive care unit (ICU) insulin delivery system, ensuring that the controller maintains patient glucose within safe limits under all modeled conditions. These works illustrate how translating clinical knowledge and device behavior into mathematical specifications can uncover edge-case failures and provide guarantees unreachable by testing alone. Tools based on timed automata, such as Uppaal [26], enable exhaustive verification of time-bounded properties in reactive systems, and have already been applied to medical guidelines and device protocols. For example, Bottrighi et al. [42] and Traoré et al. [43] used timed automata model checking to validate the temporal consistency of clinical guidelines, detecting subtle timing conflicts in recommended care pathways. Alternative formal approaches have also emerged: Huang et al. [44] recently applied TLA+ (Temporal Logic of Actions) to specify and verify properties of a prototype DT, illustrating the generality of formal methods for complex healthcare systems. In addition, probabilistic model checking tools like PRISM have been used to quantify reliability in medical devices, for example evaluating the probability of pacemaker logic maintaining heart rhythm under various failure modes [45] or computing risk metrics for implantable device safety according to regulatory standards [46].

Despite this progress in applying formal verification to medical devices and algorithms, no current HF digital twin leverages such techniques. In particular, there have been no reports of heart-failure DTs integrating model checking (deterministic or probabilistic) to guarantee alert correctness, timing deadlines, or safe behavior under all possible patient and network conditions. This gap in formal assurance is especially notable given recent calls for trustworthy AI and digital health: the National Academies' report on DTs explicitly highlights Verification, Validation, and Uncertainty Quantification (VVUQ) as pillars of trust in high-risk applications [47] and digital health systems [21]. Likewise, the biomedical community's V3 framework for digital health technologies emphasizes rigorous Verification of sensor and algorithm performance, alongside analytical and clinical validation, as prerequisites for deployment [48]. In practice, this means that without formal verification and thorough validation, clinicians cannot fully trust DT's recommendations—especially if uncertainty bounds are not provided for the twin's predictions [21]. Our work addresses the verification component (mathematically proving system correctness) as a foundation, recognizing that it must be complemented by broader validation and uncertainty quantification in future clinical evaluation.

2.3. Edge–cloud architectures and reliability challenges

From another perspective, reducing latency and ensuring reliability are major challenges for DT systems that rely on remote cloud computation [49]. Cloud-centric architectures, while computationally powerful, introduce network latency and single points of failure—serious vulnerabilities in time-sensitive healthcare scenarios. In industrial domains, edge computing (processing data near its source) has proven effective in improving real-time responsiveness and resiliency [50]. Healthcare is beginning to follow suit: Younas et al. [51] and García et al. [52] propose hybrid edge–cloud paradigms that preprocess patient data at the edge to enhance responsiveness and privacy, with the cloud providing heavy analytical power. Likewise, Farivar et al. [53] developed an IoT-enabled patient monitoring system using edge-based fuzzy logic control to adjust anesthesia in real time, demonstrating improved reliability and timing in a critical care setting. Mohamed et al. [54] and Kabir et al. [55] explored distributed intelligence for remote health monitoring, where decision-making is split between on-device agents and cloud services.

However, coordinating a distributed DT across edge and cloud layers raises new safety questions that past works have not formally addressed. In existing designs, there are no deterministic guarantees that edge-level decisions will remain consistent with cloud-level analytics, or that a network disruption will trigger a safe fallback state. For example, Krzysiak et al. [56] proposed an explainable-AI-enhanced

hybrid DT for cardiology, but their framework provides no guarantees of correct operation if the edge device loses connectivity to the cloud or if the two compute layers make conflicting inferences. Overall, fault tolerance and safety in distributed medical DT deployments remain evaluated only empirically. Important properties — such as ensuring that data synchronization meets timing requirements, that alerts issued during connectivity losses are still valid and safe, and that edge decisions do not diverge from cloud guidance — lack formal proof in current architectures. These gaps indicate that simply offloading computation to the edge, while beneficial for latency, is not sufficient without verified coordination mechanisms in place.

2.4. Limitations of existing HF digital twin frameworks

In summary, despite promising progress in DT applications for heart failure, current systems fall short of the reliability and responsiveness required in safety-critical clinical deployments.

1. **Absence of formal verification and trust guarantees**: Existing HF DTs lack mathematically rigorous assurances on their behavior. No prior system offers formal proof of correctness for its sensing-to-alert pipeline, nor alignment with emerging VVUQ/V3 credibility frameworks. This leaves their clinical outputs vulnerable to undetected errors or unsafe decisions in edge-case scenarios.
2. **Cloud-centric latency and reliability bottlenecks**: Predominantly cloud-based architectures introduce communication delays and single points of failure. Few systems support fail-safe local operation or edge-level decision-making when connectivity degrades, risking unacceptable delays or downtime during critical health events.
3. **Static or open-loop operation**: Many DTs function as offline simulations or periodic batch updates rather than continuously adaptive systems. Real-time reactivity to evolving patient data or clinician input is largely absent, limiting their usefulness in acute care or dynamic management of HF.
4. **Lack of modularity and standards compliance**: Current implementations tend to be bespoke and monolithic, complicating their integration into existing clinical IT ecosystems. Little attention is paid to interoperability standards (e.g., HL7 FHIR: Fast Healthcare Interoperability Resources) or modular design, hampering collaboration, extensibility, and regulatory approval.
5. **Explainability as an afterthought**: While some projects have started integrating explainable AI techniques [56], interpretability is typically retrofitted via external tools rather than built into the twin's decision logic. This reactive approach limits clinicians' trust and the actionable insight they can gain from twin recommendations.

These persistent gaps directly motivate the development of **HEDA-HF**: a formally verified, hybrid edge–cloud DT architecture for adaptive heart failure management. By embedding model-checking-based verification into the sensing, inference, and synchronization pipeline, HEDA-HF provides mathematically proven assurances of alert correctness, timing compliance, and safe coordination between edge and cloud—capabilities that no existing heart-failure DT currently offers. In the next sections, we detail the design of HEDA-HF and how it addresses the above shortcomings.

3. HEDA-HF: A Verified Hybrid Edge–Cloud Architecture for HF Digital Twins

3.1. Motivation: Bridging the safety gap in HF digital twins

DTs are virtual counterparts of physical entities that continuously synchronize with the real world to reflect an entity's state in (near) real time. In healthcare, a patient's DT integrates heterogeneous data streams — physiological signals, electronic health records (EHR data), wearable sensor readings, and contextual information — to estimate the patient's current status and anticipate future events. For HF, a digital twin can serve as a real-time "virtual companion", processing multi-modal data from sensors (e.g., ECG patches, blood pressure monitors, weight scales) to detect anomalies and predict impending decompensation [17,57].

Most healthcare DTs today adopt layered pipelines encompassing sensor data acquisition, edge/cloud processing, physiological modeling (simulation) or AI inference, and clinician decision-support interfaces. Frameworks such as *TwinCardio* [37] exemplify this pattern by integrating IoT-based vital sign monitoring with cardiovascular simulations and deep neural networks for risk stratification. However, a critical limitation persists across these architectures: *the absence of formal verification for correctness and safety*. In practice, system reliability is typically inferred only from pilot studies or empirical testing [58]. While such evaluations are valuable, they cannot guarantee safe operation across the full spectrum of runtime conditions—especially in asynchronous, data-intensive environments where rare edge cases (sensor failures, network delays, etc.) may arise.

To address this gap, we introduce **HEDA-HF**, a hybrid edge–cloud DT architecture that embeds *formal verification* as a first-class component of the system design. The objective of HEDA-HF is not solely to enhance performance or reduce latency, but to ensure that the DT's behavior remains *correct under all operational scenarios*. This includes handling edge cases such as sensor malfunctions, intermittent connectivity, or atypical patient responses. By explicitly incorporating a verification layer, HEDA-HF advances beyond conventional "smart monitoring" toward a clinically trustworthy, safety-critical tool for HF management. In other words, HEDA-HF treats patient monitoring and alerting with the same rigor as that applied to safety-critical control systems, providing strong assurances that cannot be obtained through testing alone [44].

Fig. 1 illustrates the high-level interactions among the system's actors and components. The patient is continuously monitored by wearable sensors; the collected data are processed locally at the edge and synchronized with cloud-based analytics. Clinicians are kept in the loop to review and respond to alerts or recommendations generated by the twin. The closed-loop nature of HEDA-HF — where sensing, inference, decision, and clinical intervention are all interconnected — demands rigorous validation, which motivates the introduction of a formal verification layer overseeing these processes. To reflect this explicitly, Fig. 1 clearly represents the role of the Formal Verification Layer (FVL) within the HEDA-HF workflow. While the FVL operates at design time, it conceptually governs runtime behavior by embedding verified behavioral constraints into the system. These constraints ensure that the deployed system behaves safely and predictably under all modeled scenarios (e,g., ensuring that every edge or cloud module adheres to provably correct logic and timing guarantees). As shown, the FVL consumes formal models and canonical HF scenarios, applies model checking (e.g., via UPPAAL), and outputs verification artifacts that constrain alert generation, failover handling, and synchronization timing. This design-time process ensures clinical safety without

3.2. Key differences from classical DT architectures

Conventional healthcare DT architectures emphasize layered data processing, physiological modeling, and AI-driven analytics to deliver decision support [12]. While frameworks such as *TwinCardio* [37] integrate these components, they generally lack rigorous mechanisms to *guarantee* correct behavior under all permissible operating conditions. Safety in prior systems is often an implicit assumption grounded in empirical testing and clinical intuition. Even extensive simulations cannot cover the full operational state space, leaving potential corner cases unchecked [59]. This limitation is especially problematic in HF

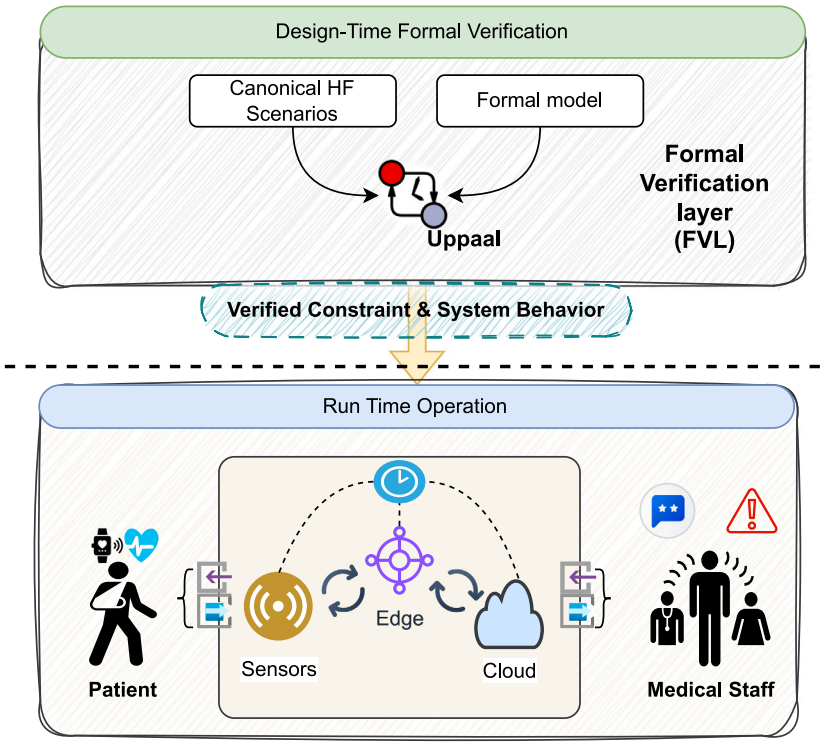

Fig. 1. System overview of the HEDA-HF digital twin architecture.

care, where a missed alert or an erroneous recommendation can have critical consequences.

HEDA-HF departs from the classical paradigm by introducing a dedicated *Formal Verification Layer* into the DT pipeline. As illustrated in Fig. 2, this layer acts as a design-time safety gatekeeper: every inference, alert, or recommendation is pre-validated against formally specified safety and timing properties before it can influence clinical workflows. In effect, HEDA-HF interposes a verification step that ensures only behavior proven to be safe and correct (under formal models) is permitted. Although illustrated as an architectural layer, the Formal Verification Layer (FVL) operates exclusively at design time. Using UPPAAL, we formally model and verify temporal and logical properties of the full sensing-to-alert pipeline prior to deployment. FVL does not act as a runtime filter, but provides enforceable contracts that constrain downstream logic during execution.

As shown in Fig. 2, Formal Verification Layer (FVL) operates during the design phase to validate the safety and timing behavior of Edge/Cloud layer, the DT Core and AI& Analysis Layer. It does not verify individual inference outcomes or predictions, but instead guarantees that the overall sensing–inference–alerting workflow complies with formal safety properties under all modeled conditions. The FVL guarantees that alerting and synchronization logic comply with safety and timing requirements. This separation allows runtime behavior to remain fast and adaptive, while still conforming to verified correctness boundaries.

To further elucidate the functional structure and architectural roles within HEDA-HF, Table 1 provides a layer-by-layer summary of the system's components. Each layer is described in terms of its core function, technical capabilities, data inputs, and outputs. This modular organization reflects HEDA-HF's end-to-end workflow — from physiological sensing and data preprocessing to formal verification, inference, and clinician interaction — designed to meet the stringent safety and responsiveness demands of heart failure management.

Real-time capabilities are essential in HF management due to the narrow intervention windows during decompensation events. For example, if fluid accumulation or arrhythmic deterioration is detected, clinical response must occur within minutes to prevent hospitalization or cardiac arrest [4,57]. Accordingly, HEDA-HF incorporates design-time guarantees on alert latency, edge–cloud failover handling, and synchronization timing to ensure that no critical information is delayed or lost. The architecture addresses concrete technical challenges such as variable sensor sampling rates, network instability, asynchronous edge/cloud inference timing, and potential data drift across compute layers. While the Formal Verification Layer guarantees workflow safety, temporal correctness, and inter-layer consistency, it does not verify model accuracy or medical efficacy of predictions—those are handled through clinical validation. If edge and cloud inference results diverge or if an alert fails verification, the system defaults to a conservative path: alerts are held for bounded-time clinician review, conflicting recommendations are suppressed, and escalation logic is triggered to avoid unsafe actions. This hybrid of formal safety guarantees and human-in-the-loop mitigation ensures resilience and trustworthiness in safety-critical contexts.

The core innovations of HEDA-HF's design are summarized as follows:

- **Formal Verification Layer (FVL):** A built-in verification module that evaluates each significant event (e.g., sensor reading, anomaly detection, alert issuance) against a predefined set of logical and temporal rules, ensuring consistency and safety before allowing progression.
- **Exhaustive Behavioral Coverage:** Through model checking techniques, HEDA-HF systematically explores all possible execution paths and state transitions of the twin. This exhaustive analysis can uncover failure modes or race conditions that traditional testing might overlook, thereby improving robustness.
- **Temporal Logic Specifications:** Key clinical requirements (for example, "a high-severity alert must be issued within 3 min of detecting decompensation" or "every alert eventually gets acknowledged by a clinician") are formally encoded as temporal logic properties (CTL/TCTL). Satisfying these properties provides provable guarantees of liveness (desired events eventually occur), safety (undesired situations never occur), and timeliness (operations complete within deadlines).

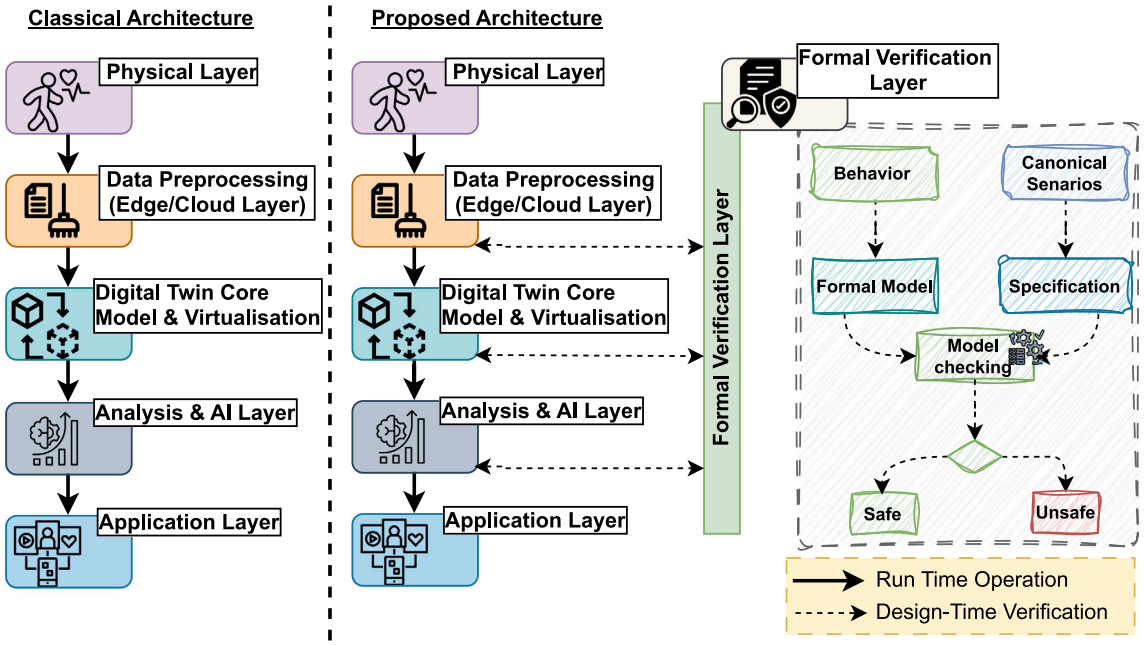

Fig. 2. Comparison of a classical healthcare DT architecture versus the proposed HEDA-HF pipeline with an integrated design-time Formal Verification Layer (FVL).

Table 1
Functional summary of HEDA-HF architectural layers.

Layer	Function	Core capabilities	Inputs	Outputs
Physical Layer	Captures real-world physiological and contextual data from the patient	Wearable sensing, ambient monitoring, real-time signal generation	ECG, BP, weight, motion, context streams	Raw sensor data
Data Preprocessing Layer	Cleans, synchronizes, and buffers raw input streams before analysis	Noise filtering, signal alignment, edge buffering, data compression	Raw sensor streams	Normalized input vectors, preprocessed signals
Digital Twin (DT) Core	Maintains a synchronized virtual model of the patient's health state	Data fusion, temporal state estimation, physiological trajectory modeling	Preprocessed sensor data, EHR, patient profile	State vector, detected deviations, anomaly flags
Formal Verification Layer (FVL)	Ensures design-time behavioral correctness and timing safety	UPPAAL model checking, CTL/TCTL property verification, timing constraint synthesis	Formal system model, HF scenarios, clinical timing specs	Verified safety contracts, logical and temporal guarantees
AI/Analysis Layer	Performs predictive inference, risk stratification, and alert generation	Anomaly detection, risk scoring, temporal pattern analysis	DT Core outputs, historical patterns, contextual signals	Alerts, recommendations, escalation triggers
Application Layer	Interfaces with clinicians for decision support and oversight	Alert visualization, interaction logging, user acknowledgment, intervention tracking	Alerts, recommendations, workflow logs	Clinician inputs, system response trace, audit trail

- **UPPAAL Toolchain Integration:** The architecture's formal models and properties are implemented in UPPAAL [26], a mature model checker for timed automata. This choice enables both design-time verification (exhaustive model checking) and on-line simulation/monitoring capabilities, leveraging a well-established verification toolchain.
- **Safety-First Design Philosophy:** Every decision-making component in HEDA-HF (e.g., the AI inference engine or alert manager) is paired with traceability and verifiability. This means the system is built from the ground up to be inspectable and auditable, facilitating future extensions like certified deployment or integration of explainable rules alongside machine learning components.

By incorporating these features, HEDA-HF elevates DTs from purely data-driven monitoring platforms to *formally assured clinical decision-support systems* that can meet the stringent safety expectations of cardiovascular care. In summary, HEDA-HF is designed to not only perform advanced HF monitoring and prediction, but to do so with a verifiable guarantee that its outputs and behaviors remain within safe bounds under all circumstances.

3.3. Canonical runtime scenarios and use cases

To concretize HEDA-HF's operational behavior and lay the groundwork for formal verification, we define a set of five canonical runtime scenarios, denoted as S_i, where $i \in \{1, \ldots, 5\}$. Each scenario is described from two perspectives: a *clinical use-case perspective* (i.e., what the system is intended to do in an HF care workflow) and a *formal verification perspective* (i.e., what properties the system must satisfy in that context). These scenarios cover routine monitoring, early risk escalation, clinician-in-the-loop adjustments, fault tolerance, and distributed edge–cloud consistency.

For each scenario S_i, we associate a set of safety, liveness, and/or timing properties expressed in temporal logic. Let $\Phi_i = \{\varphi_{i,1}, \varphi_{i,2}, \ldots, \varphi_{i,k_i}\}$ denote the set of CTL/TCTL properties to be verified for scenario S_i. Each $\varphi_{i,j}$ formally encodes a critical requirement of that scenario (e.g., "an alert is eventually issued if the patient begins decompensating" or "if the edge device loses connectivity, the system eventually recovers and resynchronizes") as a verifiable behavioral constraint within the HEDA-HF model. Below we outline the five scenarios and their corresponding formal property sets:

1. *Routine Monitoring and Daily Assessment (S_1):* From the clinical perspective, the twin continuously maintains an up-to-date patient status using periodic vital signs and symptom reports (e.g., daily weight measurements, heart rate, blood pressure, patient-reported symptoms). HEDA-HF performs proactive health assessments at regular intervals (daily or more frequently), detecting subtle trends indicative of early HF decompensation [17,57]. This scenario ensures the twin can handle long-term streaming data and remain tightly synchronized with the patient's status. From the formal verification perspective, S_1 is defined as follows:

   - *Operational focus:* Stable-state tracking with periodic data ingestion and continuous edge–cloud synchronization.
   - *Verified property group:* $\{\Phi_{\text{sync\_continuity}}, \Phi_{\text{no\_data\_loss}}\}$, ensuring no data is lost and the twin's state is consistently updated.
   - *Example property:*

     $$A <> (Data\_Ready \Rightarrow Inferencing),$$

     which in plain terms guarantees that whenever new sensor data is produced, it will eventually be processed by the inference engine (no readings are permanently ignored or stuck).
   - *Related components: Sensors, Edge, and Synchronization Channel.*

2. *Early Risk Detection and Alerting (S_2):* Clinically, this scenario covers real-time analysis of patient data streams to catch early warning signs of HF exacerbation. If the twin's predictive models detect a pattern suggestive of impending decompensation (for example, a combination of rising thoracic impedance, increasing resting heart rate, and patient-reported fatigue), the system issues a time-sensitive alert for clinician review. Timely alerts can prompt interventions before a full-blown crisis occurs. Remote hemodynamic monitoring trials underscore the value of such early detection—clinicians can intervene prior to overt symptom escalation, potentially averting emergency visits or hospitalization [57]. In HEDA-HF, any high-risk condition triggers an immediate alert that must be delivered within a specified deadline (we follow the risk prediction timing guidelines from TwinCardio [37]). From a verification standpoint, S_2 includes the following:

   - *Operational focus:* Continuous risk monitoring with time-bounded alert generation for early intervention.
   - *Verified property group:* $\{\Phi_{\text{timely\_alert}}, \Phi_{\text{alert\_liveness}}\}$, enforcing that alerts are issued within the required time window and that they reach the clinician.
   - *Example property:*

     $$A\Diamond_{\leq 180s}(Patient.Decompensating \rightarrow Cloud.Alerting),$$

     meaning "if the patient's condition deteriorates to a decompensating state, then within 180 s an alert is raised in the cloud subsystem".
   - *Related components: Edge, Cloud, and Communication Network.*

3. *Clinician-in-the-Loop Updates (S_3):* In this use case, clinicians actively interact with the DT system. Indeed, this scenario captures the iterative refinement of the twin: if a clinician adjusts a patient's medication based on twin insights, that action updates the twin's simulation parameters; conversely, if the clinician provides a correction (e.g., "this alert was not clinically significant"), the twin can learn from that input. Indeed, clinician acknowledgments, overrides, or annotations are fed back to the twin to refine parameters and maintain alignment with expert knowledge. Modern DT frameworks highlight the importance of such bidirectional learning [60]. From a formal perspective, S_3 is characterized as follows:

   - *Operational focus:* Human-in-the-loop acknowledgment, overrides, and system state updates based on clinician feedback.
   - *Verified property group:* $\{\Phi_{\text{ack\_received}}, \Phi_{\text{override\_consistency}}\}$, ensuring that every issued alert is eventually acknowledged and that any clinician override leads to a consistent state update (no conflicting alerts remain).
   - *Example property:*

     $$A\Diamond(AlertIssued \rightarrow Physician.Acknowledging),$$

     guaranteeing that for every alert generated, there will eventually be a corresponding physician acknowledgment action in the model (no alert goes ignored indefinitely).
   - *Related components: Cloud, Clinician Interface, and Feedback_Channel.*

4. *Edge Disconnection and Failover (S_4):* This scenario addresses resilience and fault tolerance. Clinically, it represents situations where the patient's edge device (wearable or home gateway) loses connectivity or fails. HEDA-HF is designed with a failover mechanism: if the edge cannot transmit data, the system should detect this condition and either fall back to alternate data sources or safely pause certain operations until reconnection, without triggering false alarms. Literature on fault-tolerant healthcare IoT suggests using cloud back-ends to ensure continuity of service [61]. The formal verification layer can be used here to prove properties such as "loss of edge connectivity will eventually invoke cloud takeover" and "no critical data will be unaccounted for during failover". This guarantees that a single-point failure at the edge will not compromise patient monitoring. So, formally, S_4 is specified as follows:

   - *Operational focus:* Resilient operation under network disruptions or edge device failures, with safe degradation and recovery.
   - *Verified property group:* $\{\Phi_{\text{failover\_trigger}}, \Phi_{\text{recovery\_timely}}\}$, ensuring that upon an edge disconnection the system enters a known safe state and that it eventually recovers or resynchronizes once the connection is restored.
   - *Example property:*

     $$A[](Edge\_Disconnected \Rightarrow \Diamond \neg Cloud\_Predicting),$$

     which guarantees that if the edge goes offline, any ongoing cloud processing is halted (preventing stale or unsafe cloud actions during disconnection) and the system transitions to a safe mode.
   - *Related components: Edge, Cloud, and Communication Network.*

5. *Edge–Cloud Inference Conflict Resolution (S_5):* In distributed intelligent systems, it is possible the edge and cloud components may temporarily have differing views or analysis outcomes (for instance, an edge AI model flags an issue while the cloud model does not, or vice versa). Clinically, the discrepancies between edge heuristics and cloud models are resolved via bounded-time consensus and, if necessary, escalation to a human reviewer. This scenario addresses how HEDA-HF resolves such inconsistencies. We implement a consensus mechanism: for example, requiring cloud confirmation for any high-severity alert that originates from the edge, or automatically escalating to a human reviewer if edge and cloud analyses disagree significantly. Conflict resolution protocols have been studied in IoT to ensure consistency between local and global analyses [62]. In HEDA-HF, a simple hierarchical approach could be: edge detections are

forwarded to cloud for validation unless time-critical, whereas cloud may override edge false negatives by issuing its own alert. We formally verify that any conflict is resolved within a bounded time and that contradictory outputs are never simultaneously presented. Recent work has applied formal methods to verify synchronization in physical–digital twin systems [44] and to prove safety properties of distributed control algorithms in cardiac devices [63]. Following these examples, we specify properties like "the system never issues both an 'alarm' and 'no alarm' for the same event" (consistency) and use model checking to validate the conflict resolution logic. Formally, S_5 is described as follows:

- *Operational focus:* Consistency enforcement and arbitration when edge and cloud analyses diverge, ensuring one source of truth for alerts.
- *Verified property group:* $\{\Phi_{\text{consistency}}, \Phi_{\text{arbiter}}\}$, which ensure that contradictory inferences do not lead to unchecked operation—either an agreement is reached or a safe default action is taken.
- *Example property:*

  $$A[]\neg(Edge.Alerting \wedge Cloud.Normal),$$

  No inconsistent state where the edge raises an alert but the cloud believes the patient is normal.
- *Related components: Edge, and Cloud.*

These scenarios collectively provide a thorough testbed for HEDA-HF's capabilities. Each scenario's workflow is encoded in our formal model to enable exhaustive verification, as described next. These dual representations provide clinical context and a verification-ready formalization, together defining a comprehensive testbed for the HEDA-HF architecture.

3.4. Formal verification layer: Specification and process

To provide provable guarantees about the correctness, timeliness, and safety of HEDA-HF, we formalize the architecture's verification layer as an overarching supervisory module. While prior DT efforts have emphasized personalization or performance, HEDA-HF's distinguishing feature is that *formal verification is embedded into its core*, ensuring all decision-making processes adhere to explicitly declared behavioral contracts.

Unlike domain-specific components (such as the ML risk predictor or the cardiac simulator), the Formal Verification Layer (FVL) operates over the entire system behavior. It observes events and state transitions across both edge and cloud and checks them against formally specified safety and liveness properties. The FVL allows the proposed architecture to incorporate formal verification as a first-class design principle, ensuring that the system's logic and workflows are proven correct before deployment. By validating all possible execution paths at design time, the architecture guarantees that no unsafe or inconsistent behavior can arise once the system is operational.

Formally, we define the global system model for HF management as:

$$S_{HF} = \langle E, S, R, T, \Psi \rangle,$$

where we define each element as follows:

1. E is the set of all input *events* originating from the physical or clinical environment. These include: (i) physiological sensor readings (e.g., heart rate, ECG waveform, SpO_2, blood pressure); (ii) clinician interactions (e.g., acknowledgment of an alarm, annotation of a teleconsultation report, override of a therapy recommendation); and (iii) system-level triggers (e.g., timers, network status changes, exception flags). Clinicians are thus an active component of E, providing manual or semi-automatic input that contributes to the system's safety loop.
2. S is the finite set of *internal states* the system can occupy, encompassing both clinical and computational dimensions. It includes: (i) the patient's estimated clinical condition, (ii) the edge device's operational mode, (iii) the cloud backend's execution state and (iv) the physician actions in the system. The global system state is represented as $S = (s_P, s_E, s_C, s_{Ph})$, where s_P, s_E, s_C and s_{Ph} denotes the current **patient/twin clinical state**, the edge state, the cloud service state and the clinician states respectively.
3. R is the set of observable *reactions* or outputs that the system can perform in response to events or state changes. Examples include: (i) generating an alert notification to a clinician, (ii) issuing a therapy recommendation or guideline-based care suggestion, (iii) uploading new data or model updates to the cloud, and (iv) adjusting internal model parameters or thresholds (twin self-adaptation).
4. T is the set of *time-bounded requirements* on event-to-reaction relationships or state transitions. These represent timing constraints, such as: "an urgent alert must be issued within 180 s of detecting a critical condition", "edge inference results must be transmitted to cloud within 5 min", or "if connectivity is lost, the system must retry within 10 s". The elements of T define the basis for the TCTL properties we will verify.
5. Ψ is the set of formal properties that the system must satisfy in all canonical scenarios, where: $\Psi = \bigcup_{i=1}^{5} \Phi_i$ (expressed in temporal logic (CTL and TCTL)). These include safety properties (nothing bad ever happens), liveness properties (something good eventually happens), reachability properties (the system can reach certain desirable states), and explicit timing constraints derived from T. These specifications drive the design-time model checking process, through which all relevant execution paths are exhaustively analyzed prior to deployment. The verification process ensures that the modeled system satisfies the required safety, liveness, and timing properties under all admissible behaviors.

We then define the Formal Verification Layer as an abstract function:

$$\mathcal{V} : Tr(S_{HF}) \rightarrow \{\texttt{safe}, \texttt{unsafe}\},$$

where:

1. $Tr(S_{HF})$ is the set of all possible finite execution traces (histories of states and events) that the HEDA-HF system may exhibit under its operational semantics.
2. For any given execution trace $\pi \in Tr(S_{HF})$, $\mathcal{V}(\pi) = \texttt{safe}$ if and only if $\pi \vDash \Psi$ (meaning the trace satisfies all the required functional/ temporal logic properties in Φ).
3. Conversely, $\mathcal{V}(\pi) = \texttt{unsafe}$ if $\pi \nvDash \Psi$, i.e., the trace violates one or more safety/timing requirements.

In essence, $\mathcal{V}$ "accepts" only those execution traces that conform to every specified property, and flags any trace that shows a violation. Importantly, the Formal Verification Layer does not perform runtime monitoring or live intervention. Instead, it establishes verified behavioral and temporal contracts during the design phase. The deployed HEDA-HF system executes according to these pre-validated contracts, ensuring that runtime behavior remains constrained within formally proven safety boundaries.

To manage the complexity of verification in a distributed edge–cloud environment, we decompose the global state space into several interacting domains:

$$\Sigma = \Sigma_P \times \Sigma_E \times \Sigma_C \times \Sigma_{Ph},$$

where:

1. Σ_P is the set of discrete patient-level states (as listed for s_P above).

2. Σ_E is the set of discrete edge-layer states.
3. Σ_C is the set of discrete cloud-layer states.
4. Σ_{Ph} is the set of discrete Physician-level states.

Each domain Σ_i contains the relevant local states for that component, including:

- **Patient/Twin States** Σ_P ={*Stable, AtRisk, Decompensating, Intervened*}.
- **Edge Device States** Σ_E ={*Idle, Sensing, Preprocessing, Inferencing, Uploading, Disconnected*}.
- **Cloud Service States** Σ_C ={*Awaiting Data, Predicting, Alerting, WaitAck, Updating_Model, Failed*}.
- **Physician-level States** Σ_{Ph} ={*Available, Alert Received, Responding, Acknowledging*}.

Transitions $\tau \in \mathcal{T}$ define the permitted transitions between these composite states $s = (s_P, s_E, s_C, s_{Ph})$. For instance, a transition might capture "edge finishes inferencing and uploads data" (edge state: Inferencing → Uploading; cloud state: Idle → UpdatingModel; patient state may remain AtRisk). The network of such transitions is modeled as a network of timed automata for formal analysis.

Using UPPAAL, we verify that under all possible sequences of events, the system respects properties in Φ. For example, a critical safety property might be:

$$AG(Critical\_Alarm \rightarrow AF(Clinician\_Acknowledged))$$

meaning "on all paths (AG), if a critical alarm is raised then on all future states (AF) eventually a clinician acknowledgment occurs". A timing property example in TCTL could be:

$$AG(DecompState \rightarrow^{\leq 180s} AlarmIssued)$$

meaning "whenever the patient twin enters a decompensating state, an alarm is issued within 180 s". Our formal verification ensures that HEDA-HF satisfies such properties or flags the design if it does not, allowing us to refine the logic until all properties hold.

To better illustrate how these verification steps are operationalized, Fig. 3 summarizes the end-to-end workflow. Starting from the canonical runtime scenarios S_1–S_5, system behaviors are abstracted into timed automata, temporal properties are expressed in CTL/TCTL, and both exhaustive and statistical model checking (MC/SMC) are executed in UPPAAL.

The results of model checking — whether property satisfaction or violation — are generated exclusively during the design phase and incorporated into the Formal Verification Layer as verified behavioral and temporal contracts. These contracts are subsequently embedded into the HEDA-HF architecture and constrain runtime execution.

No live or runtime model checking is performed after deployment. Instead, the deployed system operates according to pre-validated specifications derived from offline formal analysis. Through this design-time modeling and verification process, we obtain formal assurance that HEDA-HF satisfies its defined safety, liveness, and timing properties under the modeled assumptions. This rigorous validation approach supports clinical trust and regulatory readiness while maintaining a clear separation between design-time verification and runtime system operation.

4. Formal verification of HEDA-HF

To ensure that the proposed **HEDA-HF** architecture meets rigorous safety and timeliness criteria for heart failure (HF) monitoring, we formally specify and verify its critical operational behaviors through *model checking*. The architecture integrates a dedicated *Formal Verification Layer (FVL)*, conceived as a design-time assurance mechanism that validates the correctness of runtime modules prior to deployment. This section does not evaluate the FVL's internal logic; rather, it presents a formal model of the entire HEDA-HF workflow — including patients, sensors, edge devices, cloud services, and clinician interactions — to verify that the integrated system preserves its behavioral guarantees across all operational conditions.

While the FVL is structurally embedded within the architecture, its function is confined to design-time verification. The model checking process assumes a continuously operating runtime system and captures real-time behaviors such as asynchronous sensing, network delays, edge–cloud synchronization, and clinical response cycles. Under these assumptions, all safety, liveness, and timing properties are formally validated to ensure end-to-end reliability in realistic HF monitoring scenarios.

The system is represented as a network of synchronized *timed automata*, where each automaton models one architectural component and its interactions. This formalization allows reasoning about both functional correctness (e.g., every clinical alert is eventually acknowledged) and real-time constraints (e.g., alerts are delivered within medical deadlines). Verification is performed using the UPPAAL toolchain, which supports both exhaustive and statistical model checking over Computation Tree Logic (CTL) and Timed CTL (TCTL) properties. Indeed, we describe the scope of the formal model, the modeling assumptions, the construction of the timed-automata network in UPPAAL, and the verification of the properties associated with the scenarios $\{S_1, \ldots, S_5\}$ defined earlier.

4.1. Modeling scope and assumptions

The formal model of HEDA-HF captures the essential components and behaviors relevant to heart failure monitoring and alerting. To keep the state space tractable yet realistic, we make several abstractions and assumptions:

- **Patient Dynamics:** The patient's health state is abstracted into four discrete modes: *Stable*, *AtRisk*, *Decompensating*, and *Intervened*. Transitions between these states are governed by stochastic parameters reflecting clinical progression (e.g., a Stable patient can transition to AtRisk due to worsening vitals). We assume these transitions follow a Markovian process with parameters calibrated from clinical data (literature sources or domain expertise).
- **Sensor and Edge Behavior:** The edge device (or wearable sensor hub) periodically samples vital signs and trigger edge-side analysis. Edge devices preprocess signals and, depending on the inferred risk, may upload the processed data to the cloud for further analysis. We assume sensors produce new data at a regular interval (with an upper-bound period) and that each cycle of edge processing must complete within a known deadline (e.g., an inferencing deadline of 60 s as per system specification).
- **Cloud Analytics:** The cloud component receives processed data from the edge, performs any deeper predictive analytics (e.g., more complex risk models), raises alerts and coordinates with the physician interface. We assume the cloud processing has its own time bound (e.g., completes within 120 s per batch of data).
- **Clinician Interaction:** A *Physician* actor is modeled to capture acknowledgment of alerts or application of interventions. We abstract the clinician's response time as a non-deterministic delay bounded by a reasonable limit (e.g., must acknowledge within a few minutes). The physician acknowledges or rejects alerts with probabilistic delays that reflect human response variability.
- **Communications and Synchronization Architecture**: The ecosystem communicates through synchronized channels and flag systems that model realistic communication and interactions. Synchronization channels capture interactions across modules as follow: (i) `dataready_ok/dataready_risk` for edge to cloud data transmission, (ii) `cloud_alert` for cloud-to-clinician notifications, and (iii) `Feedback_ok/ Ignore/ Ack` for physician acknowledgment and feedback. Global boolean flags and integer variables track certain conditions (e.g., a flag that the edge

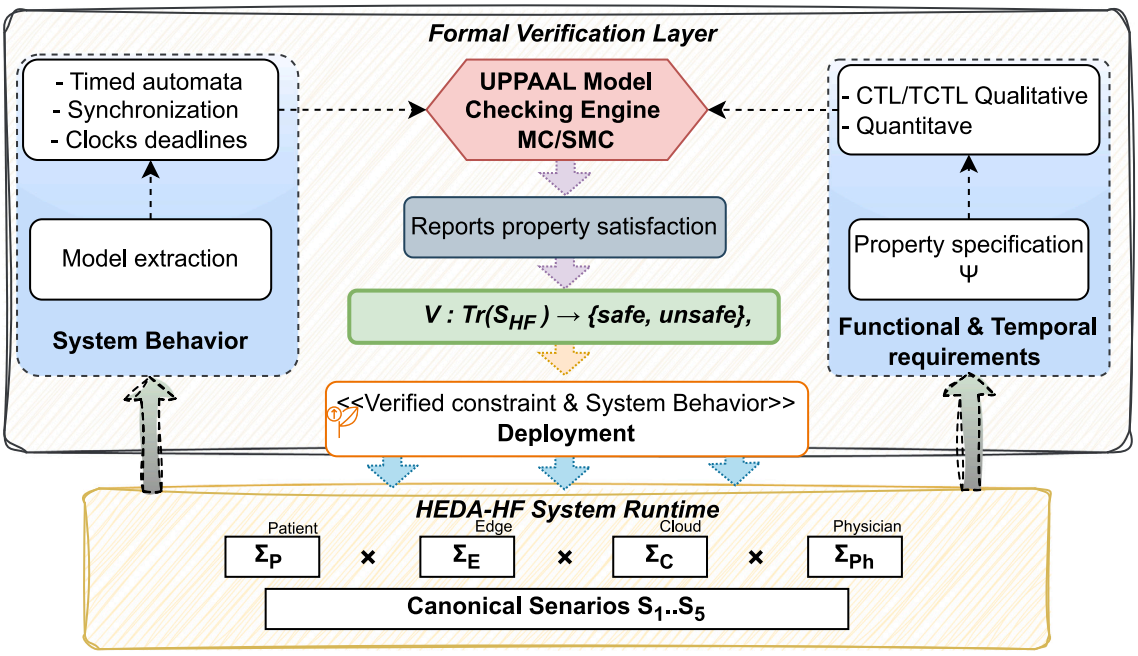

Fig. 3. Design-time verification workflow of the HEDA-HF system.

is disconnected `edge_down`, or counters for retry attempts). In addition, we explicitly model failure scenarios, such as edge disconnection and cloud delays, and incorporate recovery transitions (failover to cloud or edge recovery).

- **Timing Model:** All timing in the model is captured using clocks and timing constraints. Clock variables enforce clinically-relevant deadlines derived from heart failure management protocols for sensing, inference, upload, alerting, acknowledgment, failover, and arbitration (e.g., sensing windows). Key timing assumptions include: (i) sensors adhere to periodic sampling with bounded jitter, (ii) edge/cloud analysis has bounded deadlines, and (iii) clinician acknowledgments occur within hospital protocol limits. All clocks and variables are bounded to ensure the model is finite-state and exhaustively checkable.

These timing bounds follow clinically endorsed service-level objectives (SLOs) for HF remote monitoring and alert triage, as outlined in the ESC Heart Failure Guidelines [4] and recent telecardiology protocols [57]. Our complete formal model is implemented in UPPAAL for reproducibility and independent verification. The implementation incorporates the following technical specifications:

Clock Variables and Timing Semantics:

- `global_time`: Master temporal reference for system-wide coordination
- `t_sense`: Edge sensing cycle timer (bound: `PERIOD` = 60 s)
- `inference_timer`: Local AI processing deadline (bound: `MAX_INFERENCE_TIME` = 60 s)
- `alert_timer`: End-to-end alert delivery constraint (bound: `MAX_ALERT_DELAY` = 180 s)
- `connection_timer`: Edge reconnection timeout (bound: `MAX_RTO` = 60 s)

Clinical Parameter Constants:

- `NORMAL_PERIOD` = 200 s: Monitoring interval for stable patients.
- `RISK_PERIOD` = 60 s: Increased monitoring frequency for at-risk patients.
- `CRIT_PERIOD` = 20 s: Critical monitoring urgency for decompensating patients.
- `MAX_CLOUD` = 60 s: Cloud processing service level agreement.
- `MAX_UPDATE` = 180 s: Model update completion deadline.

These modeling choices ensure that the verification focuses on critical HF management behaviors (monitoring, alerting, etc.) without extraneous complexity. The abstractions (such as discrete patient states and fixed deadlines) are conservative with respect to safety: if a property holds under these assumptions, it should hold in any more detailed implementation that respects the same timing and logic constraints. By combining these abstractions with real-time constraints, the HEDA-HF model provides a faithful yet analyzable representation of the closed-loop patient monitoring process.

4.2. Model specification (Timed automata)

The HEDA-HF system was implemented as a network of timed automata using UPPAAL. Each subsystem is represented by an independent automaton with local clocks and state variables, synchronized via broadcast channels to capture inter-component interactions. This modular representation reflects the distributed nature of the edge–cloud–clinician loop. This approach enables us to rigorously capture both functional behavior and temporal constraints, which are critical in heart failure monitoring where late or inconsistent responses can have severe consequences.

- **Patient Automaton** Σ_P: The Patient automaton Σ_P cycles between *Stable*, *AtRisk*, *Decompensating*, and states, with stochastic rates governing progression (Fig. 4). An *Intervened* state models successful physician intervention. Transition triggers include physiological signals (normal or critical) received by sensors and physician actions. This abstraction allows us to reason about disease progression and medical response under time-constrained conditions.
- **Edge Device Automaton** Σ_E: The edge automaton Σ_E represents the processing pipeline at the gateway or wearable device. It explicitly models *Idle, Sensing, Preprocessing, Inferencing, Uploading,* and *Disconnected* states (Fig. 5). These capture data acquisition, local analysis, communication with the cloud, and possible connectivity interruptions. By formalizing these behaviors, we can verify that inference and upload deadlines are respected even under adverse network conditions.

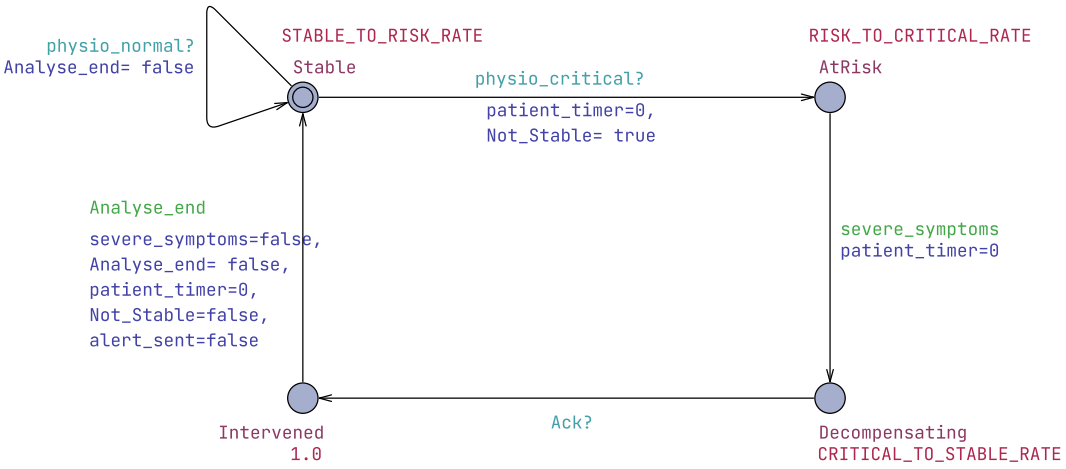

Fig. 4. Patient automaton.

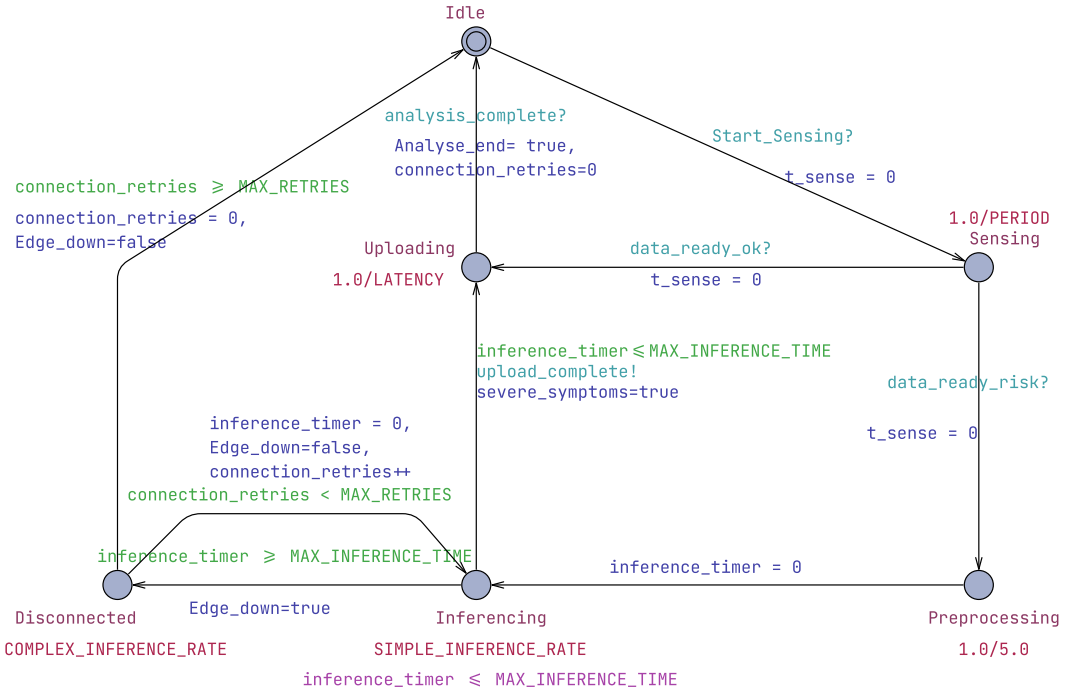

Fig. 5. Edge automaton.

- **Cloud Automaton** Σ_C: The cloud automaton Σ_C models backend services, including *AwaitingData, Predicting, Alerting, UpdatingModel, Failed, and Waiting For Ack* states (Fig. 6). It captures analytics, alert generation, model updates, and fault scenarios. This abstraction is central for ensuring that cloud operations provide timely predictions and reliable alert delivery in synchrony with the edge and patient automat.
- **Physician Automaton** Σ_H (**Human**): Clinician automaton captures acknowledgment and decision-making steps in response to alerts. It has states such as `available` (no pending alerts) and `Responding` (an alert is being addressed) (Fig. 7).. Upon entering `Responding`, a bounded delay is introduced (representing the clinician reviewing the alert and responding, e.g., within 120 s). When the physician acknowledges the alert (or provides an intervention), a synchronization transitions the system (e.g., clearing the alert and possibly causing the Patient automaton to transition to an `Intervened` state if treatment was given).

This formal specification provides a rigorous foundation for expressing and verifying system-level properties in temporal logic. Together, these automata form the basis of our UPPAAL model, enabling the verification of safety, liveness, and time-bounded properties across distributed components.

To create a realistic ecosystem, we also include a simplified Sensor automaton that handles periodic data generation. This ensures the closed-loop interactions (patient → sensor → edge) are represented, even if at an abstract level. The Sensor automaton abstracts periodic data acquisition and transmission delays. Although simplified, these components are essential for modeling closed-loop interactions and ensuring that the system reflects all critical stakeholders.

5. Experimental evaluation and results

To emulate real-time operation, we designed all experiments using

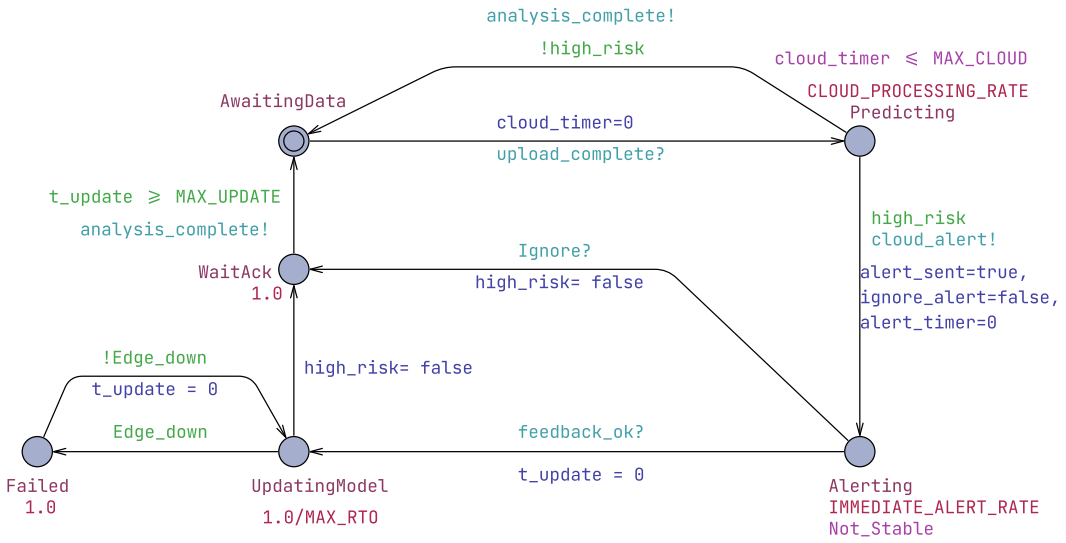

Fig. 6. Cloud automaton.

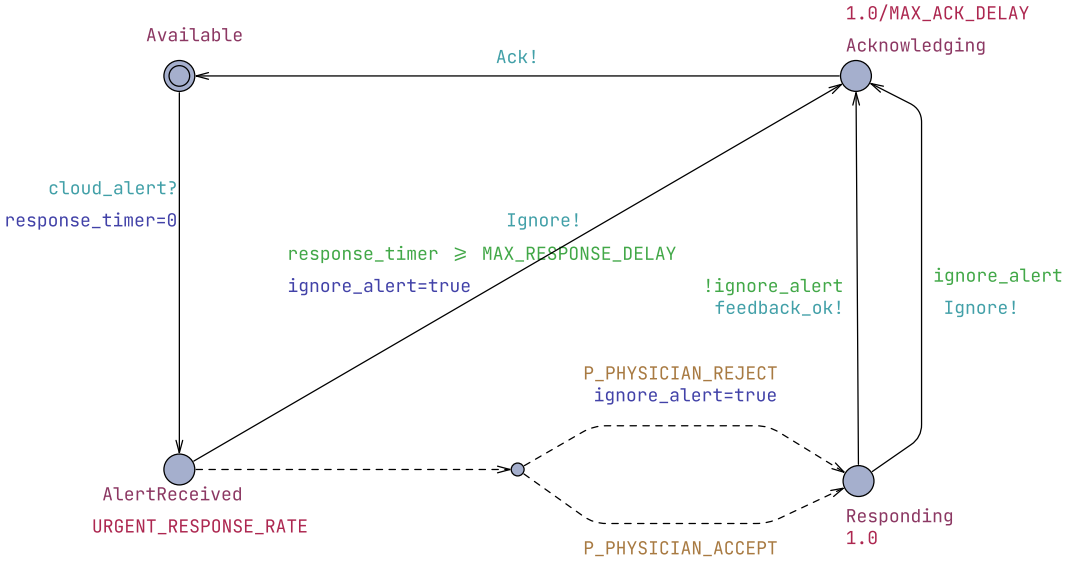

Fig. 7. Physician automaton.

incrementally replayed sensor data streams with clinically inspired timing intervals. Instead of batch processing, the formal verification models simulate event-driven behavior, where data arrives asynchronously and triggers system responses under bounded delays.

We adopted a comprehensive verification workflow combining classical model checking for functional properties (e.g., safety, liveness, and bounded timing) and statistical model checking (SMC) for performance-related analysis. These experiments simulate runtime behavior under UPPAAL using timed automata models derived from the FVL specification. What is verified are the behavioral contracts that the FVL enforces by design—such as the maximum allowable delay between data acquisition and alert generation. While the FVL itself does not operate dynamically at runtime, these verified guarantees are embedded into the system logic and govern its runtime correctness.

5.1. Qualitative properties specification

Using the scenario-driven requirements, we formally specified a set of key properties $P_i, i \in [1, 12]$ in UPPAAL to capture the HEDA-HF's essential safety, liveness, robustness, and timeliness and fault-tolerance requirements. Each property P_i is expressed in Computation Tree Logic (CTL) or Timed CTL (TCTL), providing a precise formal guarantee over all possible system behaviors (i.e., providing a precise yes/no criterion that can be exhaustively verified over the model's state space). The qualitative properties P_1–P_{12} correspond directly to the combined requirements from all scenarios (Φ_1–Φ_5). Each P_i encodes one or more functional/temporal constraints property derived from S_i through their formal mapping.

Table 2 lists the full set of verified properties. For each property, we provide a natural-language description, classification (safety, liveness, robustness, or reliability-related), give the formal CTL/TCTL specification, and report the verification result. These properties span critical safety conditions (e.g., absence of deadlock or contradictory states), liveness guarantees (required events eventually occur), time-bounded performance (operations complete within deadlines), and robustness under fault conditions. Together, they encode the clinical requirements for HEDA-HF's correctness and responsiveness.

Each of the 12 formally verified properties (P_1–P_{12}) was mapped to one or more of the five canonical runtime scenarios (S_1–S_5), ensuring complete behavioral coverage of HEDA-HF. Scenario S_1 (routine monitoring) is supported by safety and timeliness properties (P_1, P_6, P_7,

Table 2
Formally verified properties of the HEDA-HF model, with CTL/TCTL specifications, informal descriptions, category, and verification result. All properties were validated as *satisfied* in the UPPAAL model.

ID	Formal specification & Description	Category	Result
P_1	$A[]\neg deadlock$. System never freezes or halts (no deadlocks), ensuring continuous operation.	Safety	**Satisfied**
P_2	$A[]\neg(Patient.Stable \wedge Patient.Decompensating)$ No contradictory patient states can occur (the patient cannot be both stable and decompensating simultaneously).	Safety (Consistency)	**Satisfied**
P_3	$A[](Patient.Stable \implies \neg Cloud.Alerting)$. No false alarm is raised when the patient is stable.	Safety	**Satisfied**
P_4	$A <> (Patient.Decompensating \implies alert\_sent)$. If the patient decompensates, eventually an alert is sent.	Liveness	**Satisfied**
P_5	$A <> (alert\_sent \implies Physician.Responding)$. Every issued alert is eventually acknowledged by a physician (no alert is ignored indefinitely).	Liveness	**Satisfied**
P_6	$A <> (Sensor.DataReady \implies Edge.Inferencing)$. Sensor measurements are eventually processed by the AI engine.	Liveness	**Satisfied**
P_7	$A <> (Edge.Inferencing \implies AI.Uploading)$. Inference results are eventually uploaded to the cloud.	Liveness	**Satisfied**
P_8	$A <> (Cloud.Predicting \implies Analyse\_end)$. Cloud prediction always completes, producing a result.	Liveness	**Satisfied**
P_9	$A[](Sensor.Sensing \implies t_{sense} \leq PERIOD)$. The sensor produces a new reading within each sampling period (sensor never misses its periodic measurement).	Timeliness	**Satisfied**
P_{10}	$A[](Edge.Inferencing \implies t_{Edge} \leq T_{MAX\_INFERENCE\_TIME})$. Edge AI inference completes before its deadline `MAX_INFERENCE_TIME` (edge processing meets its worst-case time bound).	Timeliness	**Satisfied**
P_{11}	$A[](Cloud.Predicting \implies cloud\_timer \leq MAX\_CLOUD)$. Cloud analytics (predictive computations) finish within the allocated cloud processing bound $cloud_timer$.	Timeliness	**Satisfied**
P_{12}	$A[](Edge.Disconnected \implies \Diamond\neg Cloud.Predicting)$. Network failure handling: if the edge disconnects abruptly, any ongoing cloud prediction is eventually halted or reset, ensuring graceful recovery from network faults.	Robustness (Fault Tolerance)	**Satisfied**

P_9, P_{10}) verifying continuous sensing and periodic edge/cloud updates. Scenario S_2 (early risk detection) is backed by correctness and liveness properties (P_3, P_4, P_8, P_{11}), ensuring timely alerts and predictive analysis. Scenario S_3 (clinician-in-the-loop) maps to P_5, guaranteeing that every alert is acknowledged. Scenario S_4 (edge disconnection) is governed by P_{12}, verifying safe failover and recovery. Finally, S_5 (edge–cloud conflict resolution) aligns with P_2, ensuring consistency in patient state interpretation. This mapping confirms that HEDA-HF's operational logic is fully covered by formally verifiable guarantees under all clinical contexts.

To illustrate how these formally verified properties align with and reinforce the architectural pipeline, Fig. 8 provides a stage-wise breakdown of functional assurances. At the *Patient* stage, properties ensure consistent state representation and that decompensation triggers alerting; at the *Sensor* stage, they prevent data loss and enforce periodic acquisition; at the *Edge*, they ensure inference meets deadlines and results propagate to the cloud; at the *Cloud*, they guarantee deterministic completion within time bounds; and at the *Physician* stage, they ensure escalation and acknowledgment. System-wide properties additionally enforce deadlock-freedom and reliable failover under edge faults, ensuring that every step of HEDA-HF's workflow is governed.

Using UPPAAL's exhaustive model checking engine, we confirmed that all properties P_1–P_{12} hold on the HEDA-HF model. In other words, the verifier found no counterexample or violation for any specified requirement. Specifically, P_1–P_3 ensure **safety** (no deadlocks, no contradictory patient states, and no spurious alerts), P_4–P_8 ensure **liveness** (critical events like deterioration and alerts eventually occur and propagate to completion), P_9–P_{11} enforce **timeliness** (sensor readings, edge inferences, and cloud analyses all meet their deadlines), and P_{12} ensures **robustness** (the system tolerates edge disconnections by safely pausing cloud actions). The exhaustive state-space exploration revealed no violations of these properties; in other words, no unsafe or unexpected states were encountered outside the specified bounds. These results mean that, for the idealized model we constructed, every possible execution sequence of HEDA-HF is guaranteed to respect the defined safety, liveness, and timing constraints. Practically, this implies the twin will not miss a true HF deterioration, will not raise false alarms during stability, will operate within its time limits, and will handle network or device failures gracefully. This level of assurance is a significant step toward making DTs *clinically dependable* and *regulatorily sound*.

5.2. Statistical Model Checking (SMC) and performance analysis

While the qualitative verification guarantees correctness over all behaviors in an absolute sense, we further evaluate HEDA-HF's performance and reliability under uncertainty using UPPAAL SMC. This involves introducing probabilistic behavior into the model and running repeated randomized simulations to capture real-world variability and then statistically checking performance properties that correspond to clinically meaningful service-level objectives (SLOs), such as alert timeliness, failover continuity, and physician acknowledgment.

Stochastic instrumentation (Model semantics)

We enriched the HEDA-HF model with probabilistic semantics for SMC by assigning random distributions to transitions and using UPPAAL's modeling primitives for probability. Specifically, we use exponential distributions (commonly used for modeling time-to-event in reactive systems) for timing of certain actions to reflect variability. Each periodic process (sensing, processing, etc.) is given an exponential delay with diminishing probability of longer delays; in our model we ensure it cannot exceed the hard bound. For example, if `PERIOD` is

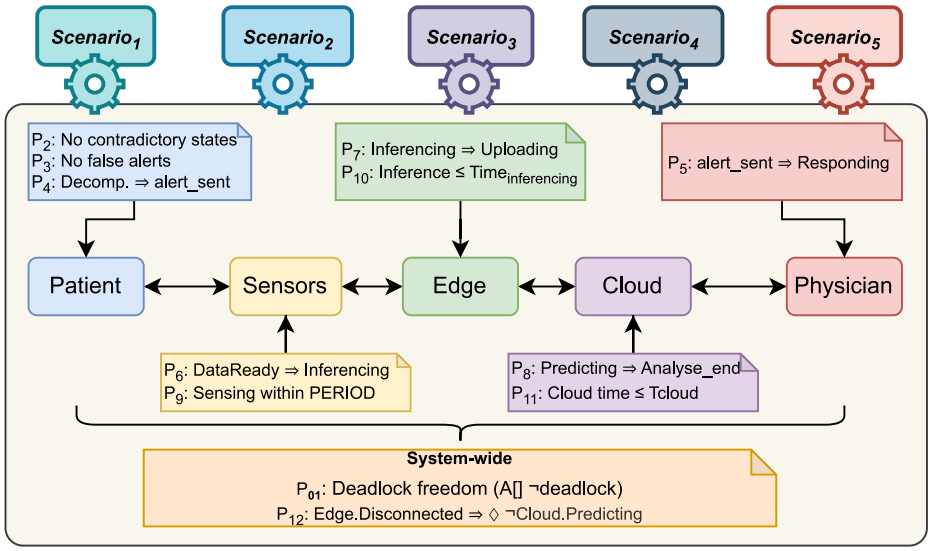

Fig. 8. End-to-end clinical workflow and mapping of design-time verified properties.

60 s, we might model the sensing interval as an exponential distribution with mean, say, 30 s (so it almost always triggers a reading by 60 s, with diminishing probability of longer delays but never exceeding the hard bound). Branching probabilities are introduced for non-deterministic choices. Indeed, event-level uncertainty is added with branching probability labels. Concretely, we introduced parameters like $P_{edge\_High\_Risk} = 0.15$ (15% chance that a given cycle detects a high-risk condition at sensing) vs. normal condition (85%), $P_{physician\_accept} = 0.90$ (90% chance a physician acknowledgment to an alert), and with `MAX_RETRIES` = 3 for model updates.

In summary, the model for SMC retains the same structure and hard guarantees, but adds distributional information to simulate realistic operation variability.

Experimental protocol and estimation

All estimates are obtained with UPPAAL SMC's sequential hypothesis testing and Monte-Carlo simulation. For probabilities. We executed SMC queries in UPPAAL by running a large number of simulations (up to 10^6 runs for certain queries to get tight confidence intervals) and using sequential hypothesis testing to estimate probabilities. we report point estimates with 95% confidence; for mean/percentile latencies, we report descriptive statistics consistent with the SMC output (we use the same seeds and bounds across experiments to ensure comparability). Acceptance criteria are derived from clinical SLOs: (i) alerts delivered within 180 s with probability ≥0.99, (ii) failover within 60 s with probability ≥0.99, (iii) deadlock probability ≈0, (iv) physician acknowledgment eventually occurs (Probability distribution of acknowledgment times (we ensure eventually 100% are acknowledged; we look at within 120 s as a metric)), and (v) model updates complete within 180 s with probability ≥0.99. These criteria mirror the requirements encoded in the formal properties but now treated in a probabilistic sense.

SMC Query Formulation: With the probabilistic model in place, we posed a series of SMC queries to evaluate how the system performs under uncertainty. These queries correspond to clinically relevant questions about system reliability and efficiency.

Table 3 presents a summary of representative SMC queries formulated to assess whether the HEDA-HF architecture fulfills key service-level objectives (SLOs) under conditions of uncertainty. The results provide quantitative evidence of the system's reliability, timeliness, and robustness across a range of clinically meaningful scenarios.

While HEDA-HF introduces a formally verified framework for heart failure monitoring, this does not imply that prior digital twin solutions are inapplicable or ineffective. Existing works have made substantial contributions to physiological modeling, AI-based prediction, and remote patient monitoring. The HEDA-HF architecture is intended to complement these efforts by integrating formal design-time guarantees for timing, behavioral correctness, and coordination—areas that remain underexplored in earlier designs.

The analysis reflects formal validation of simulated runtime behavior based on design-phase system models, rather than a qualitative assessment of architectural limitations or performance of deployed systems.

Result interpretation

We interpret some of the key findings from Table 3 to shed light on HEDA-HF's performance:

Timeliness and Performance (Q1–Q2): Fig. 9 plots the cumulative distribution of alert delivery times, confirming that alerts are almost always delivered well before the 180 s bound. In fact, HEDA-HF meets the 3-minute alert requirement with ≈99% probability, and the average alert delivery time is around 114 s (with a tight 95% CI of ±4 s). The expected worst-case alert delay observed in extensive simulations is about 108 s, which provides a comfortable safety margin (40% faster than the requirement). These results demonstrate that the alerting subsystem consistently satisfies the clinical SLO for notification timeliness, even under variable conditions, ensuring rapid response to early warning signs of HF deterioration.

Edge–Cloud Throughput and Latency (Q_3–Q_4, Q_8): HEDA-HF's distributed data pipeline exhibits sustained throughput with low latency. For example, as query Q_3 shows, sensor data uploads from the edge to cloud complete within 300 s with ≈97% probability (and average 104 s). Cloud predictions (Q_4) always finished within the 60 s bound in our simulations (observed probability 100%), typically taking only 28 s on average. Fig. 10 illustrates the distribution of edge-to-cloud upload times, indicating that the system rarely experiences significant backlogs. Moreover, the sensor cadence query (Q_8) confirms that in 95% of cycles, new data arrived within the expected 60 s period (with an average period of 38 s, well under the maximum). This implies that HEDA-HF maintains continuous data flow without accumulating delays—no noticeable buffering or slow-down occurs across cycles, which is crucial for real-time monitoring.

In addition, Fig. 11 illustrates the statistical verification of the sensing cadence associated with property Q_8 in Table 3. This SMC query evaluates whether sensor data become available within each 60 s cycle under stochastic timing variability. The histogram of inter-arrival times

Table 3
Evaluation of service-level objectives (SLOs) through probabilistic model checking using UPPAAL, SMC for the HEDA-HF architecture. The table reports formal property specifications, informal descriptions, target thresholds, and measured outcomes with 95% confidence intervals.

ID	Category	Formal property & Description	Measured result (95% CI)	SLO target	Req. Met
Q1	Timeliness	$\Pr_{[\leq 180\ s]}(\Diamond\ Cloud.Alerting)$ *Checks whether alerts are issued within 180 s after patient decompensation.*	Mean alert time 114.2 ± 4.4 s; Pr(alert ≤ 180) ≈ 0.99	Alert ≤ 180 s	Yes
Q2	Performance (Alert bound)	$E_{[\leq \mathrm{MAX\_ALERT\_DELAY};10^6]}(\max : alert\_timer)$ *Measures the expected maximum alert timer before alert emission.*	Mean alert delay 108.2 ± 0.04 s	Max alert sending < 180 s	Yes
Q3	Liveness (Edge→Cloud)	$\Pr_{[\leq 300\ s]}(\Diamond\ AI.Uploading)$ *Verifies that sensed data are uploaded from the edge to the cloud within 300 s.*	Mean 104.2 ± 11.8 s; Pr(upload ≤ 300) ≈ 0.97	Upload ≤ 300 s	Yes
Q4	Liveness (Prediction availability)	$\Pr_{[\leq \mathrm{MAX\_CLOUD}]}(\Diamond\ Cloud.Predicting)$ *Ensures that cloud predictions are produced within 60 s.*	Mean 28.3 ± 10.1 s; Pr(predict ≤ 60) ≈ 1.0	Predict ≤ 60 s	Yes
Q5	Robustness (Edge recovery)	$\Pr_{[\leq \mathrm{MAX\_RTO}]}(\Diamond\ Edge.Inferencing)$ *Confirms that the edge device recovers and resumes inferencing within 60 s after failure.*	Mean 42.6 ± 3.0 s; Pr(recover ≤ 60) ≈ 1.0	Recover ≤ 60 s	Yes
Q6	Timeliness (Early alerts)	$\Pr_{[\leq 60\ s]}(\Diamond\ Cloud.Alerting)$ *Evaluates how often alerts are raised within the first minute.*	Mean 48.9 ± 8.9 s; early alerts ≈ 6%	Alert ≤ 60 s	Yes
Q7	Reliability (Acknowledgment)	$\Pr_{[\leq \mathrm{MAX\_ACK\_DELAY}]}(\Diamond\ Physician.Acknowledge)$ *Checks whether physicians acknowledge alerts within 120 s.*	Mean 96.2 ± 7.1 s; Pr(ack ≤ 120) ≈ 1.0	Ack ≤ 120 s	Yes
Q8	Robustness (Sensing cadence)	$\Pr_{[\leq \mathrm{PERIOD}]}(\Diamond\ Sensor.DataReady)$ *Verifies that sensors deliver data once per 60 s cycle.*	Mean 38.4 ± 2.5 s; Pr(ready ≤ 60) ≈ 0.95	Cycle = 60 s	Yes

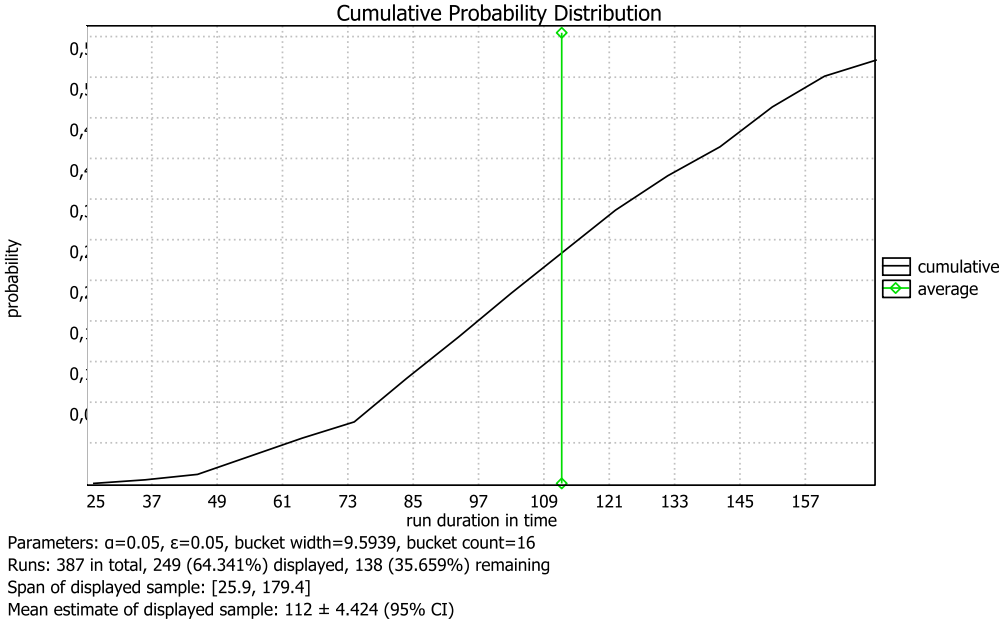

Fig. 9. Cumulative distribution (CDF) of alert delivery times in scenario S_2 (early risk detection).

provides a distributional view of this probabilistic verification. The observed mean data readiness time of approximately 38.4 s remains well below the 60 s bound, and even the slowest 5% of cycles satisfy the 60 s timing requirement. These results indicate stable and predictable sensing throughput, confirming that stochastic variability remains consistent with the formally specified design-time timing constraint and ensuring continuous synchronization between the patient state and its digital twin representation.

The probability density distribution remains concentrated around the nominal sampling period despite stochastic variability, indicating consistent sensing behavior. This regularity supports continuous synchronization between the physical patient state and its DT representation while remaining within the formally specified timing bounds.

Robustness and Reliability (Q_5–Q_7): HEDA-HF proved highly reliable in handling edge failures and ensuring alerts are acknowledged. Query Q_5 shows that if the edge disconnects, it resumes operation (inferencing) typically within 42.6 ± 3 s and virtually always by <60 s (the bound), confirming strong fault tolerance. This means temporary outages have minimal impact on the continuity of care. Query Q_7 indicates physicians acknowledge alerts within 120 s essentially 100% of the time in our model. The average acknowledgment consistently occur within 96 ± 7 s, fully closing the feedback loop before the 120 s deadline. These results speak to the system's robustness: even when things go wrong (network drop, etc.), HEDA-HF recovers quickly and maintains its safety guarantees (no unacknowledged critical alerts).

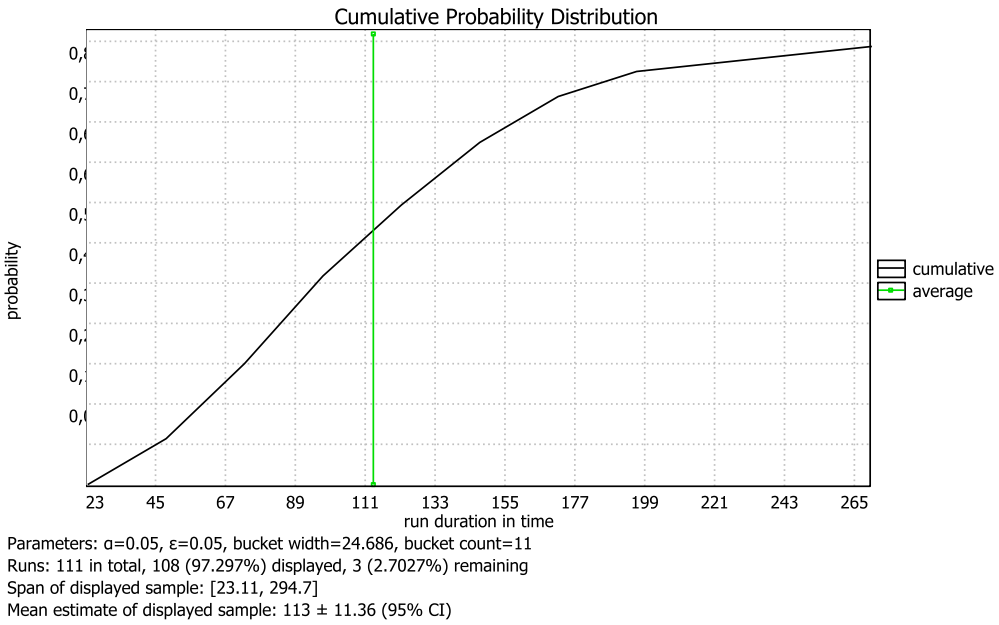

Fig. 10. Cumulative distribution of data upload completion times from edge to cloud (scenario S_1/S_2).

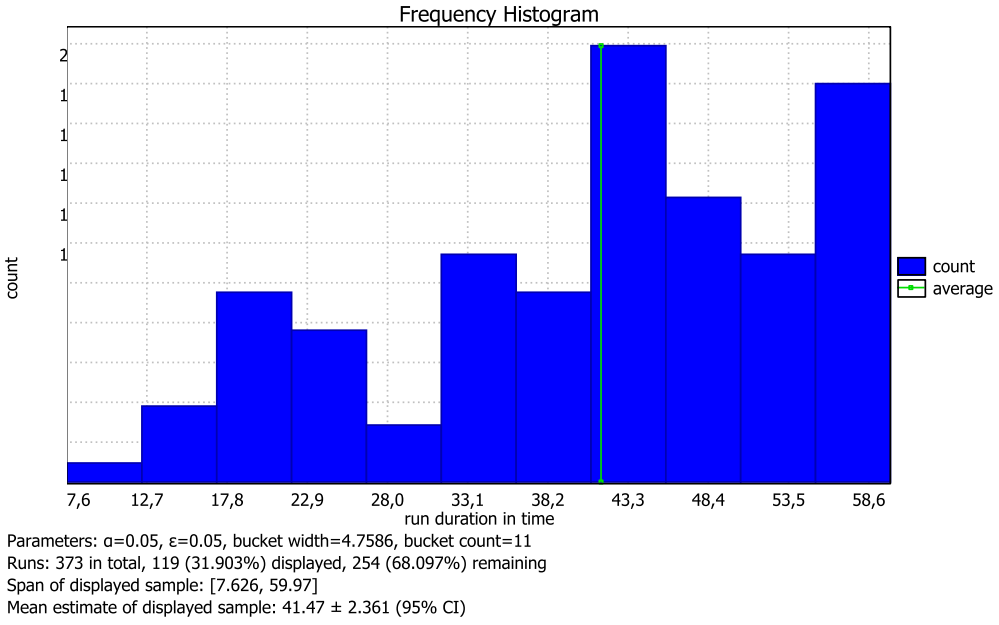

Fig. 11. Statistical verification of sensing cadence (Property Q_8). Histogram of sensor data inter-arrival times obtained via UPPAAL (SMC). The observed distribution is evaluated against the formally specified timing constraint of 60 s, demonstrating probabilistic compliance with the design-time bound.

These outcomes attest to the resilience of the architecture and its dependable human-in-the-loop behavior under stochastic variability.

Overall, the statistical model-checking analysis shows that HEDA-HF not only preserves logical correctness but also achieves high-confidence probabilistic guarantees of *timeliness, robustness, and reliability*. The verified properties confirm that the digital-twin pipeline remains responsive and safe under realistic timing fluctuations, making it suitable for deployment in real-time clinical monitoring contexts.

6. Discussion

The proposed **HEDA-HF** architecture represents a significant advance in DT engineering for heart failure (HF) management. It addresses several longstanding limitations in the field by embedding *formal verification*, *probabilistic performance analysis*, and *domain-specific design* into a unified, modular, and hybrid edge–cloud system.

Unlike prior HF DT systems—often cloud-centric, simulation-based, or empirically evaluated. HEDA-HF rigorously guarantees correctness, responsiveness, and safety across all reachable execution paths. Through the use of UPPAAL model checking and temporal logic (CTL/TCTL), the system is verified for critical properties such as deadlock-freedom, mutual exclusion, and time-bounded alert propagation. Statistical Model Checking (SMC) complements this with reliability metrics under uncertainty: alerts are delivered within 180 s with probability **0.99**, edge-to-cloud uploads complete within 300 s with probability **0.97**, and clinician acknowledgments occur within **96 ± 7 s**. These results confirm that HEDA-HF robustly satisfies its predefined service-level objectives (SLOs) even under stochastic timing and communication variability.

Table 4
Comparison of HEDA-HF with representative prior DT studies in HF and healthcare.

Work	Domain	DT scope	Arch. Type	Real-time fidelity	Verification approach	Evaluation
[64]	General HC	System	Edge	Yes	Not reported	Conceptual
[56]	CVD	Patient	Hybrid	Yes	Not reported	Conceptual
[65]	HF	Organ	Cloud	No	Not reported	Simulation
[15]	HF	Patient	Cloud	No	Empirical validation	Clinical
[66]	HF	Organ	Cloud	No	Empirical validation	Clinical
[16]	HF	Organ	Cloud	No	Empirical validation	Simulation
[67]	General HC	System	Hybrid	Yes	Empirical validation	Clinical
Our work	**HF**	**System**	**Hybrid**	**Yes**	**Model checking (MC + SMC) at design time**	**Formally verified**

From an architectural standpoint, HEDA-HF's hybrid deployment balances latency and resilience: time-critical sensing and inference occur at the edge, while the cloud handles advanced analytics and long-term model adaptation. Verified handshaking and failover protocols ensure continuous operation — even under network interruptions — without compromising temporal consistency or data integrity. This edge–cloud coordination resolves the performance bottlenecks and failure risks observed in earlier cloud-only or batch-mode DTs.

Further, HEDA-HF follows a modular design with clearly defined interfaces between sensing, inference, cloud analysis, and verification components. This enhances scalability, maintainability, and interoperability (e.g., with HL7 FHIR standards), allowing components like AI risk models or clinician dashboards to be replaced or upgraded independently. Notably, the Formal Verification Layer acts as a plug-in supervisory module, showcasing how safety assurance can be retrofitted into existing DT pipelines.

Crucially, HEDA-HF is tailored to HF-specific clinical workflows. Its formal properties and timing constraints are derived from established HF guidelines, including alert response windows, physiological thresholds for decompensation, and physician-in-the-loop intervention models. Unlike generic DT frameworks, HEDA-HF encodes domain knowledge across five canonical HF scenarios—ensuring alignment with real-world patient care pathways.

As summarized in Table 4, HEDA-HF is, to the best of our knowledge, the only reported architecture that simultaneously combines heart-failure specificity, hybrid edge–cloud deployment, real-time operation, and formally verified correctness. While prior systems provide valuable contributions — such as AI-driven risk prediction or HF-tailored simulation models — they do not integrate mathematically grounded verification with real-time distributed coordination, and therefore do not provide formal behavioral guarantees. Table 4 presents a structured comparison of representative digital twin architectures across domain specificity, architectural type (edge, cloud, or hybrid), real-time support, verification approach, and evaluation level. The taxonomy adopted for the "Verification Approach" column distinguishes three methodological categories: "Not reported", indicating that no explicit correctness or formal verification method is described; "Empirical validation", referring to evaluation through simulation studies, prototype implementations, or clinical experiments without exhaustive state-space guarantees; and "Model checking (MC + SMC)", denoting the application of formal verification techniques, including temporal logic-based exhaustive analysis (CTL/TCTL) and statistical model checking under stochastic assumptions at design time. The "Evaluation" column further differentiates whether system claims are supported by conceptual design, simulation-based analysis, clinical validation, or formal verification outcomes. This structured articulation clarifies the methodological distinction between empirical performance assessment and mathematically grounded correctness assurance, highlighting the architectural and verification completeness of HEDA-HF.

Despite its strengths, HEDA-HF has limitations that must be acknowledged, particularly regarding real-world implementation at this development stage. Formal verification introduces computational overhead at design time, which may pose challenges when scaling to more complex physiological models or larger patient cohorts. Incremental verification or compositional techniques (e.g., assume–guarantee reasoning) may be required as the system evolves.

In addition, the current model abstracts heart failure progression into discrete, formally specified states. While this abstraction enables rigorous verification, real-world HF management is characterized by substantial patient heterogeneity influenced by comorbidities, medication adherence, lifestyle factors, and socioeconomic conditions. Future deployment will therefore require patient-specific calibration mechanisms and integration with longitudinal clinical datasets to ensure personalization beyond the present formal abstraction.

Sensor reliability and variability in home-monitoring environments also represent practical constraints. Wearable devices may experience signal artifacts, intermittent connectivity, battery limitations, or inconsistent usage. Although HEDA-HF verifies behavior under bounded timing and failure assumptions, real-world deviations may necessitate complementary runtime monitoring mechanisms capable of detecting divergences from the verified model (e.g., arising from hardware faults or unmodeled network behavior). Such monitors would act as runtime sentinels while preserving the design-time verification foundation.

Regulatory and certification pathways for AI-enabled digital twins extend beyond architectural correctness. While formal verification strengthens safety assurance, clinical deployment additionally requires prospective validation studies, usability evaluation, cybersecurity compliance, and alignment with medical device regulatory frameworks. Similarly, hybrid edge–cloud deployment may face infrastructure constraints in settings with limited or unstable connectivity, potentially affecting synchronization guarantees despite formally verified failover logic.

Finally, integration into routine clinical workflows remains a critical deployment challenge. Interoperability with heterogeneous EHR systems, data-governance constraints, and clinician adoption — including mitigation of alert fatigue — must be addressed before large-scale implementation. Although the architecture is FHIR-compatible in design, real-world integration requires institutional alignment and health IT coordination.

While HEDA-HF does not implement a dedicated trust quantification framework, it ensures operational reliability through formally verified system behavior and carefully designed safety boundaries. The Formal Verification Layer guarantees timing, workflow safety, and edge–cloud coordination under all modeled conditions. In scenarios where those guarantees may be violated — such as inference divergence, unexpected delays, or partial failures — the architecture defaults to a conservative mitigation path: alerts are deferred to clinician review, edge or cloud components reinitialize via failover logic, and unsafe actions are blocked by design. Every alert or recommendation is issued with traceable metadata, allowing clinician oversight and accountability. This architecture not only strengthens operational reliability but also aligns with emerging ethical frameworks for trustworthy AI in medicine, which emphasize transparency, human agency, and auditable automation as essential principles in clinical systems [12]. Thus, even though the AI inference itself is not verified for medical accuracy, its integration into the clinical loop is bounded by formally guaranteed

timing and behavioral safeguards, supporting robust and auditable decision-making.

In summary, HEDA-HF establishes a rigorously verified architectural foundation for heart failure digital twins. By embedding formal methods and probabilistic guarantees into a modular, domain-aligned architecture, it demonstrates that real-time intelligent decision support can be both responsive and trustworthy. Future work will focus on personalization, large-scale clinical validation, adaptive runtime monitoring, and deeper integration with healthcare infrastructures to further bridge the gap between formal assurance and bedside deployment.

7. Conclusion

This paper presented HEDA-HF, a formally verified hybrid edge–cloud digital twin (DT) architecture for heart failure (HF) management. Building on a systematic analysis of existing cardiovascular DT systems, we identified a critical lack of verifiability, safety guarantees, and temporal assurance in current approaches. To overcome these challenges, HEDA-HF introduces a formally grounded design methodology that unites temporal logic specification, modular timed-automata modeling, and exhaustive UPPAAL-based verification.

The novelty of HEDA-HF lies in four foundational contributions: (i) it introduces a *verification-first digital twin architecture* with an embedded Formal Verification Layer (FVL) that enforces temporal safety contracts; (ii) it achieves *complete satisfaction of functional/temporal set rigorously defined CTL/TCTL properties* across the full reachable execution space, marking the first formally proven correctness baseline in HF digital twins; (iii) it integrates UPPAAL Statistical Model Checking to deliver *probabilistic guarantees* with >0.99 confidence for clinically mandated service-level objectives; and (iv) it formalizes and verifies *five canonical clinical runtime scenarios*, covering continuous monitoring, early escalation, physician-in-the-loop response, edge disconnection fallback, and cloud arbitration, thereby demonstrating operational trustworthiness under both deterministic and stochastic uncertainty. Collectively, these contributions establish HEDA-HF as the first DT architecture in the HF domain to transition from empirical validation to *provable correctness*, marking a pivotal step toward certifiable medical AI.

Future developments will focus on extending HEDA-HF with *explainable AI layers to complement formal guarantees with clinician-facing interpretability*, deploying the architecture in *real-world HF monitoring testbeds to assess usability and regulatory readiness*, and exploring *patient-specific formal models and adaptive runtime assurance mechanisms* to maintain provable safety as system parameters evolve over time.

A key future direction will also involve integrating lightweight runtime monitors to detect deviations from the verified model, enabling real-time safety enforcement and anomaly detection during deployment. This will help bridge the gap between static design-time verification and dynamic runtime variability in clinical settings.

Ultimately, HEDA-HF sets a new benchmark for *verifiable, adaptive, and deployable* DT systems. By elevating formal verification to a first-class design principle rather than an afterthought, it advances the clinical trust, regulatory readiness, and technological reliability of next-generation AI-driven healthcare platforms.

CRediT authorship contribution statement

Mohamed Ramdani: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Conceptualization. **Teodoro Montanaro:** Conceptualization, Writing – review & editing, Validation, Supervision, Methodology, Investigation. **Yousra Ben Aissa:** Writing – review & editing, Validation. **Luigi Patrono:** Writing – review & editing, Supervision, Project administration, Funding acquisition.

Ethics approval and consent

This study did not involve human participants, real patient data, or identifiable clinical information. All results are derived from formal system models, synthetic input scenarios, and simulation-based verification. No real-world clinical data were collected, processed, or analyzed. The clinical workflows and alert logic modeled in this study were informed exclusively by publicly available heart failure management guidelines and peer-reviewed literature. No clinician interviews, patient involvement, or institutional clinical evaluations were conducted. Accordingly, approval from a Research Ethics Committee (REC)/Institutional Review Board (IRB) and informed consent were not required.

Declaration of Generative AI and AI-assisted technologies in the writing process

ChatGPT (OpenAI, 2024) was used to support language editing, grammar correction, and readability improvements during manuscript preparation. No AI tools were used for data analysis, formal modeling, or scientific reasoning. All scientific content, results, and conclusions were generated and verified exclusively by the authors. **No figures or graphical elements in this manuscript were generated using generative AI systems.** All architectural diagrams were manually created using conventional design tools (draw.io), and all automata models and statistical plots were directly exported from the UPPAAL toolchain. The authors take full responsibility for the accuracy, provenance, and originality of all visual materials included in this manuscript.

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.

Acknowledgments

This study was partially funded by the European Union - Next Generation EU, PRIN 2022 PNRR call, under the project “Interactive digital twin solutions for cardiovascular disease Management, PReventiOn and treatment leVeraging the internet of things and Edge intelligence paradigms” - IMPROVE (CUP F53D23009250001). In addition, it has been partially funded by the Italian Ministry of Health, Italian Health Operational Plan (Cohesion and Development Fund 2014–2020), trajectory 2 “eHealth, advanced diagnostics, medical device and mini invasiveness”, project “Sistema di Monitoraggio ed Analisi basato su intelligenza aRTificiale per pazienti affetti da scompenso CARdiaco cronico con dispositivi medici mini-invasivi e indossabili Evoluti - SMART CARE” (CUP F83C22001380006). Finally, it has been partially funded by the “MUR, Ministero dell’Universita e della Ricerca” Ministerial Decree n. 737 dated 25-06-2021 “Criteri di riparto e utilizzazione del Fondo per la promozione e lo sviluppo delle politiche del Programma Nazionale per la Ricerca (PNR)” under the project title: “IDAS – Innovazione Digitale in Ambito Salute” - CUP: F84D22000270001.

Appendix. Statistical model checking parameters and robustness analysis

This appendix summarizes the stochastic parameters used in the Statistical Model Checking (SMC) analysis described in Section 5.2. For transparency and reproducibility, we report the exponential rate parameters and discrete branching probabilities encoded in the UPPAAL model, together with their engineering rationale and a concise qualitative sensitivity note.

UPPAAL SMC follows the standard stochastic timed-automata semantics, in which enabled stochastic transitions race according to

Table A.5
Stochastic parameters used in the UPPAAL SMC instrumentation of HEDA-HF. For exponential delays, λ denotes the rate (mean delay $1/\lambda$). Probabilities are expressed as percentages.

Parameter (Model)	Distribution/Value	Engineering/Modeling rationale	Qualitative sensitivity note
STABLE_TO_RISK_RATE	Exp($\lambda = \frac{1}{1440}$)	Daily-scale transition from stable to at-risk state (abstracted disease progression).	Alters frequency of risk episodes but does not affect bounded alert deadlines.
RISK_TO_CRITICAL_RATE	Exp($\lambda = \frac{1}{180}$)	Escalation to critical state within a bounded clinical horizon.	Changes incidence rate of critical events; per-event timing guarantees remain enforced.
CRITICAL_TO_STABLE_RATE	Exp($\lambda = \frac{1}{720}$)	Recovery transition after intervention.	Affects residence time in critical state without relaxing safety constraints.
RECOVERY_RATE	Exp($\lambda = \frac{1}{300}$)	Post-alert stabilization abstraction.	Impacts mean recovery duration; hard timing bounds remain unchanged.
SIMPLE_INFERENCE_RATE	Exp($\lambda = \frac{1}{3}$)	Fast edge processing for single-signal inference.	Mean latency varies with λ but remains below model-enforced limits.
COMPLEX_INFERENCE_RATE	Exp($\lambda = \frac{1}{10}$)	Multi-signal inference pipeline latency.	Shifts expected inference time without violating bounded inference constraints.
CLOUD_PROCESSING_RATE	Exp($\lambda = \frac{1}{18}$)	Cloud analytics processing delay in hybrid DT setting.	Influences average upload-to-prediction delay; deadline contracts remain active.
ACK_PROCESSING_RATE	Exp($\lambda = 2.0$)	Transition upon clinician acknowledgment.	Minor impact on end-to-end latency; acknowledgment bound explicitly verified.
IMMEDIATE_ALERT_RATE	Exp($\lambda = 1.0$)	Urgent alert emission abstraction.	Affects mean alert latency only; maximum delay is bounded in verification queries.
URGENT_RESPONSE_RATE	Exp($\lambda = \frac{1}{120}$)	Modeled clinician response for urgent alerts.	Changes distribution of acknowledgment times but not liveness guarantees.
ROUTINE_RESPONSE_RATE	Exp($\lambda = \frac{1}{1200}$)	Routine follow-up response abstraction.	Impacts non-urgent paths only; safety invariants unaffected.
P_EDGE_LOW_RISK	15%	Probability of low-risk detection branch.	Alters alert frequency but not workflow correctness.
P_EDGE_HIGH_RISK	85%	Complementary high-risk branch probability.	Same as above; structural properties preserved.
P_PHYSICIAN_ACCEPT	90%	Modeled clinician acceptance probability.	Affects acknowledgment path prevalence without altering bounded handling.
P_PHYSICIAN_REJECT	10%	Complementary rejection probability.	No impact on safety/liveness guarantees.

exponential delays with rate λ (expected delay $1/\lambda$), and discrete alternatives are selected according to declared probabilities. **No custom sampling algorithms or modified statistical configurations were introduced beyond the default UPPAAL SMC setup.** The selected stochastic parameters reflect common abstractions for latency and event processes in distributed cyber–physical systems. Exponential distributions model memoryless service and escalation delays, while discrete probabilities capture branching behavior such as risk detection and clinician response (see Table A.5).

Sensitivity analysis indicates that moderate variations (approximately ±20%) in exponential rates or branching probabilities primarily shift mean latencies and event frequencies without compromising formally verified safety, liveness, or deadlock-freedom properties. The architecture's guarantees rely on explicit bounded clocks and pre-verified design-time contracts; therefore, even under degraded network or processing conditions within the modeled ranges, hard deadline constraints and fail-safe behaviors remain enforced. Consequently, the qualitative conclusions regarding robustness and workflow correctness are stable under reasonable parameter perturbations.

Data availability

No data was used for the research described in the article.

References

[1] Groenewegen A, Rutten FH, Mosterd A, Hoes AW. Epidemiology of heart failure: the prevalence of heart failure and ventricular dysfunction in older adults over time—a systematic review. Eur J Hear Fail 2020;22(8):1342–56. http://dx.doi.org/10.1002/ejhf.1858.

[2] Rucco A, Shumba AT, Montanaro T, Sergi I, Patrono L. Enhancing chronic heart failure monitoring, prevention, and management with IoT and AI: A systematic literature review. IEEE J Biomed Health Inform 2025;1–16. http://dx.doi.org/10.1109/JBHI.2025.3628501.

[3] Berardinelli D, Conti A, Hasnaoui A, Casabona E, Martin B, Campagna S, Dimonte V. Nurse-led interventions for improving medication adherence in chronic diseases: A systematic review. Healthcare 2024;12(23). http://dx.doi.org/10.3390/healthcare12232337.

[4] Ponikowski P, Voors AA, Anker SD, et al. 2016 ESC guidelines for the diagnosis and treatment of acute and chronic heart failure. Eur Heart J 2016;37(27):2129–200. http://dx.doi.org/10.1093/eurheartj/ehw128.

[5] Feijen M, Egorova AD, Beeres SL, Treskes RW. Early detection of fluid retention in patients with advanced heart failure: A review of a novel multisensory algorithm, HeartLogicTM. Sensors 2021;21(4):1361.

[6] Mehra MR, Vukićević M, Isath A. Digital Twins and artificial intelligence in heart failure: The premise and the promise. J Card Fail 2026. http://dx.doi.org/10.1016/j.cardfail.2025.12.009.

[7] Umeh CA, Torbela A, Saigal S, Kaur H, Kazourra S, Gupta R, Shah S. Telemonitoring in heart failure patients: Systematic review and meta-analysis of randomized controlled trials. World J Cardiol 2022;14(12):640.

[8] Totaro NG, Pellegrino G, Corallo A, Minelli M, Minelli M, Gervasi M. Designing a Digital Twin of the gut microbiome: A data-driven approach for personalized medicine. Lecture Notes in Comput Sci 2026;15740 LNCS:210–20. http://dx.doi.org/10.1007/978-3-031-97772-5_14.

[9] Girau R, Anedda M, Presta R, Corpino S, Ruiu P, Fadda M, Lam C-T, Giusto D. Definition and implementation of the cloud infrastructure for the integration of the human Digital Twin in the social internet of things. Comput Netw 2024;251:110632. http://dx.doi.org/10.1016/j.comnet.2024.110632.

[10] Pellegrino G, Gervasi M, Angelelli M, Corallo A. A conceptual framework for Digital Twin in healthcare: Evidence from a systematic meta-review. Inf Syst Front 2025;27(1):7–32. http://dx.doi.org/10.1007/s10796-024-10536-4.

[11] Kaur H, Bhatia M. Digital Twins: A scientometric investigation into current progress and future directions. Expert Syst Appl 2025;265:125917. http://dx.doi.org/10.1016/j.eswa.2024.125917.

[12] Bruynseels K, Santoni de Sio F, van den Hoven J. Digital Twins in healthcare: Ethical implications of an emerging engineering paradigm. Front Genet 2021;10:31. http://dx.doi.org/10.3389/fgene.2018.00031.

[13] Shumba A-T, Montanaro T, Sergi I, Bramanti A, Ciccarelli M, Rispoli A, Carrizzo A, De Vittorio M, Patrono L. Wearable technologies and AI at the far edge for chronic heart failure prevention and management: a systematic review and prospects. Sensors 2023;23(15):6896. http://dx.doi.org/10.3390/s23156896.

[14] Trayanova NA, Prakosa A. Up digital and personal: How heart Digital Twins can transform heart patient care. Hear Rhythm 2024;21(1):89–99. http://dx.doi.org/10.1016/j.hrthm.2023.10.019.

[15] Gu F, Meyer AJ, Ježek F, Zhang S, Catalan T, Miller A, Schenk NA, Sturgess VE, Uceda D, Li R, et al. Identification of Digital Twins to guide interpretable AI for diagnosis and prognosis in heart failure. NPJ Digit Med 2025;8(1):110. http://dx.doi.org/10.1038/s41746-025-01501-9.

[16] Koopsen T, Gerrits W, van Osta N, van Loon T, Wouters P, Prinzen FW, Vernooy K, Delhaas T, Teske AJ, Meine M, et al. Virtual pacing of a patient's Digital Twin to predict left ventricular reverse remodelling after cardiac resynchronization therapy. Europace 2024;26(1):euae009. http://dx.doi.org/10.1093/europace/euae009.

[17] Jameil AK, Al-Raweshidy H. A Digital Twin framework for real-time healthcare monitoring: Leveraging AI and secure systems for enhanced patient outcomes. Discov Internet Things 2025;5(1):37. http://dx.doi.org/10.1007/s43926-025-00135-3.

[18] Sel K, Osman D, Zare F, Masoumi Shahrbabak S, Brattain L, Hahn J-O, Inan OT, Mukkamala R, Palmer J, Paydarfar D, et al. Building Digital Twins for cardiovascular health: From principles to clinical impact. J Am Hear Assoc 2024;13(19):e031981. http://dx.doi.org/10.1161/JAHA.123.031981.

[19] Corral-Acero J, Margara F, Marciniak M, Rodero C, Loncaric F, Feng Y, Gilbert A, Fernandes JF, Bukhari HA, Wajdan A, et al. The 'Digital Twin'to enable the vision of precision cardiology. Eur Heart J 2020;41(48):4556–64. http://dx.doi.org/10.1093/eurheartj/ehaa159.

[20] Tasmurzayev N, Amangeldy B, Imanbek B, Baigarayeva Z, Imankulov T, Dikhanbayeva G, Amangeldi I, Sharipova S. Digital cardiovascular twins, AI agents, and sensor data: A narrative review from system architecture to proactive heart health. Sensors 2025;25(17):5272. http://dx.doi.org/10.3390/s25175272.

[21] Sel K, Hawkins-Daarud A, Chaudhuri A, Osman D, Bahai A, Paydarfar D, Willcox K, Chung C, Jafari R. Survey and perspective on verification, validation, and uncertainty quantification of Digital Twins for precision medicine. Npj Digit Med 2025;8(1):40. http://dx.doi.org/10.1038/s41746-025-01447-y.

[22] Panagoulias DP, Tsihrintzis GA, Virvou M. Challenges in regulating and validating AI-driven healthcare. In: Artificial intelligence-empowered bio-medical applications. Springer; 2025, p. 135–52. http://dx.doi.org/10.1007/978-3-031-90174-4_6.

[23] Clarke EM, Henzinger TA, Veith H, Bloem R. Handbook of model checking. Springer International Publishing; 2018, p. 1–1210. http://dx.doi.org/10.1007/978-3-319-10575-8.

[24] Baier C, Katoen J-P. Principles of model checking. MIT Press; 2008.

[25] Alur R, Dill DL. A theory of timed automata. Theoret Comput Sci 1994;126(2):183–235.

[26] Behrmann G, David A, Larsen KG. A tutorial on uppaal. Form Methods Real-Time Syst 2004;200–36. http://dx.doi.org/10.1007/978-3-540-30080-9_7.

[27] Adetunji O. Advanced digital-twin odeling for predictive monitoring of postoperative cardiac patients using wearables and EHR data. GSC Biological Pharm Sci 2024;27(1):295–314. http://dx.doi.org/10.30574/gscbps.2024.27.1.0174.

[28] Wang M, Hu H, Wu S. Opportunities and challenges of Digital Twin technology in healthcare. Chin Med J (Engl) 2023;136(23):2895–6. http://dx.doi.org/10.1097/CM9.0000000000002896.

[29] Amadiaz Y, Alfonso-Lizarazo E, Nait Sidi Moh A. A systematic review of healthcare cyber–physical systems with associated innovative technologies for Alzheimer's and Parkinson's diseases. Array 2025;28. http://dx.doi.org/10.1016/j.array.2025.100575.

[30] Mihai S, Yaqoob M, Hung DV, Davis W, Towakel P, Raza M, Karamanoglu M, Barn B, Shetve D, Prasad RV, Venkataraman H, Trestian R, Nguyen HX. Digital Twins: A survey on enabling technologies, challenges, trends and future prospects. IEEE Commun Surv & Tutorials 2022;24(4):2255–91.

[31] Zambrano V, Mueller-Roemer J, Sandberg M, Talasila P, Zanin D, Larsen PG, Loeschner E, Thronicke W, Pietraroia D, Landolfi G, Fontana A, Laspalas M, Antony J, Poser V, Kiss T, Bergweiler S, Pena Serna S, Izquierdo S, Viejo I, Juan A, Serrano F, Stork A. Industrial digitalization in the industry 4.0 era: Classification, reuse and authoring of digital models on Digital Twin platforms. Array 2022;14. http://dx.doi.org/10.1016/j.array.2022.100176.

[32] Merizig A, Mokhtari B, Harabi Y, Ayad S, Kassimi D. Digital Twin-based smart irrigation for sustainable water management in arid areas. In: 2025 7th international conference on pattern analysis and intelligent systems. PAIS, 2025, p. 1–6. http://dx.doi.org/10.1109/PAIS66004.2025.11126475.

[33] Munusamy S, Jothi K. Blockchain-IoMT-enabled federated learning: An intelligent privacy-preserving control policy for electronic health records. Array 2025;28:100586. http://dx.doi.org/10.1016/j.array.2025.100586.

[34] Katsoulakis E, Wang Q, Wu H, Shahriyari L, Fletcher R, Liu J, Achenie L, Liu H, Jackson P, Xiao Y, et al. Digital Twins for health: a scoping review. NPJ Digit Med 2024;7(1):77.

[35] Trayanova NA, Lyon A, Shade J, Heijman J. Computational modeling of cardiac electrophysiology and arrhythmogenesis: toward clinical translation. Physiol Rev 2024;104(3):1265–333. http://dx.doi.org/10.1152/physrev.00017.2023.

[36] Coorey G, Figtree GA, Fletcher DF, Snelson VJ, Vernon ST, Winlaw D, Grieve SM, McEwan A, Yang JYH, Qian P, et al. The health Digital Twin to tackle cardiovascular disease—a review of an emerging interdisciplinary field. NPJ Digit Med 2022;5(1):126. http://dx.doi.org/10.1038/s41746-022-00640-7.

[37] Iyer AA, Umadevi K. Design and analysis of TwinCardio framework to detect and monitor cardiovascular diseases using Digital Twin and deep neural network. Sci Rep 2025;15(1):24376. http://dx.doi.org/10.1038/s41598-025-08824-3.

[38] Ramdani M, Kahloul L, Khalgui M, Li Z, Zhou M. RCTL: New temporal logic for improved formal verification of reconfigurable discrete-event systems. IEEE Trans Autom Sci Eng 2021;18(3):1392–405. http://dx.doi.org/10.1109/TASE.2020.3006435.

[39] Biere A. Bounded model checking. In: Handbook of satisfiability. IOS Press; 2021, p. 739–64. http://dx.doi.org/10.3233/FAIA201002.

[40] AlQadheeb A, Bhattacharyya S, Perl S. Enhancing cybersecurity by generating user-specific security policy through the formal modeling of user behavior. Array 2022;14. http://dx.doi.org/10.1016/j.array.2022.100146.

[41] Chen Y, Wang L, et al. Formal verification of clinical AI workflows using timed automata. Artif Intell Med 2023;140:102490. http://dx.doi.org/10.1016/j.artmed.2023.102490.

[42] Bottrighi A, Giordano L, Molino G, Montani S, Terenziani P, Torchio M. Adopting model checking techniques for clinical guidelines verification. Artif Intell Med 2010;48(1):1–19. http://dx.doi.org/10.1016/j.artmed.2009.09.003.

[43] Traore MK, Gorecki S, Ducq Y. A formal framework for Digital Twin modeling, verification, and validation. In: Digital Twins, simulation, and the metaverse: driving efficiency and effectiveness in the physical world through simulation in the virtual worlds. Springer; 2024, p. 119–43. http://dx.doi.org/10.1007/978-3-031-69107-2_6.

[44] Huang L, Varshney LR, Willcox KE. Formal verification of Digital Twins with TLA and information leakage control. 2024, http://dx.doi.org/10.48550/arXiv.2411.18798, arXiv preprint arXiv:2411.18798.

[45] Kwiatkowska M, Norman G, Parker D. Probabilistic model checking: Advances and applications. Form Syst Verif: State-of the-Art Futur Trends 2017;73–121. http://dx.doi.org/10.1007/978-3-319-57685-5_3.

[46] Cicotti G, Coronato A. Towards a probabilistic model checking-based approach for medical device risk assessment. In: 2015 IEEE international symposium on medical measurements and applications (meMeA) proceedings. 2015, p. 180–5. http://dx.doi.org/10.1109/MeMeA.2015.7145195.

[47] Willcox K, Bingham D, Chung C, Chung J, Cruz-Neira C, Grant C, Kinter J, Leung R, Moin P, Ohno-Machado L, et al. Foundational research gaps and future directions for Digital Twins. National Academies Press; 2023.

[48] Goldsack JC, Coravos A, Bakker JP, Bent B, Dowling AV, Fitzer-Attas C, Godfrey A, Godino JG, Gujar N, Izmailova E, et al. Verification, analytical validation, and clinical validation (V3): the foundation of determining fit-for-purpose for biometric monitoring technologies (BioMeTs). Npj Digit Med 2020;3(1):55. http://dx.doi.org/10.1038/s41746-020-0260-4.

[49] Amin SU, Hossain MS. Edge intelligence and internet of things in healthcare: A survey. IEEE Access 2020;9:45–59.

[50] Zhang T, Li Y, Chen CP. Edge computing and its role in industrial internet: Methodologies, applications, and future directions. Inform Sci 2021;557:34–65. http://dx.doi.org/10.1016/j.ins.2020.12.021.

[51] Younas MI, Iqbal MJ, Aziz A, Sodhro AH. Toward QoS monitoring in IoT edge devices driven healthcare—a systematic literature review. Sensors 2023;23(21):8885.

[52] Garcia J, Marquez D. Edge intelligence in healthcare: Design and deployment challenges. IEEE Internet Things J 2023;10(5):3875–89. http://dx.doi.org/10.1109/JIOT.2023.3254167.

[53] Farivar F, Jolfaei A, Manthouri M, Haghighi MS. Application of fuzzy learning in IoT-enabled remote healthcare monitoring and control of anesthetic depth during surgery. Inform Sci 2023;626:262–74. http://dx.doi.org/10.1016/j.ins.2022.12.094.

[54] Mohamed Nouh BOC, Brahmi R, Cheikh S, Ejbali R, Nanne MF. AI-driven resource allocation in edge-fog computing: Leveraging digital twins for efficient healthcare systems.. Int J Adv Comput Sci Appl 2025;16(4). http://dx.doi.org/10.14569/ijacsa.2025.01604101.

[55] Kabir MR, Ray S. DT-IoMT: A Digital Twin reference model for secure internet of medical things. In: 2024 IEEE computer society annual symposium on VLSI. ISVLSI, IEEE; 2024, p. 433–8. http://dx.doi.org/10.1109/ISVLSI61997.2024.00084.

[56] Krzysiak R, An D, Chen Y. XCardio-Twin: An explainable framework to aid in monitoring and analysis of cardiovascular status. In: 2023 IEEE 3rd international conference on Digital Twins and parallel intelligence. DTPI, IEEE; 2023, p. 1–6. http://dx.doi.org/10.1109/DTPI59677.2023.10365417.

[57] Shah R, Kedia S, Rawooth M, Chokalingam K, Kadambi P, Parchani G. Remote monitoring in heart failure: recent trends and future perspectives. Med Res Arch 2023;11(1). http://dx.doi.org/10.18103/mra.v11i1.3489.

[58] Sun T, He X, Li Z. Digital Twin in healthcare: Recent updates and challenges. Digit Health 2023;9:20552076221149651. http://dx.doi.org/10.1177/20552076221149651.

[59] Elleuch M, Tahar S. Formal analysis of an iot-based healthcare application. In: 2023 IEEE symposium on computers and communications. ISCC, IEEE; 2023, p. 1–5.

[60] Faisal SM, Ishrat M, Khan W. Digital Twins in healthcare: Revolutionizing patient care and medical operations. In: Digital Twins for smart cities and urban planning. CRC Press; 2025, p. 69–89.

[61] Aburukba R, et al. Federated learning-driven IoT request scheduling for fault tolerance in cloud data centers. Mathematics 2025;13(13):2198. http://dx.doi.org/10.3390/math13132198.

[62] Khaldy MAA, Nabot A, Al-Qerem A, Jebreen I, Darem AA, Alhashmi AA, Alauthman M, Aldweesh A. Adaptive conflict resolution for IoT transactions: A reinforcement learning-based hybrid validation protocol. Sci Rep 2025;15(1):25589. http://dx.doi.org/10.1038/s41598-025-09698-1.

[63] Chen T, Diciolla M, Kwiatkowska M, Mereacre A. Quantitative verification of implantable cardiac pacemakers. In: 2012 IEEE 33rd real-time systems symposium. 2012, p. 263–72. http://dx.doi.org/10.1109/RTSS.2012.77.

[64] Chen J, Wang W, Fang B, Liu Y, Yu K, Leung VCM, Hu X. Digital Twin empowered wireless healthcare monitoring for smart home. IEEE J Sel Areas Commun 2023;41(11):3662–76. http://dx.doi.org/10.1109/JSAC.2023.3310097.

[65] Martinez-Velazquez R, Gamez R, El Saddik A. Cardio twin: A Digital Twin of the human heart running on the edge. In: 2019 IEEE international symposium on medical measurements and applications. meMeA, 2019, p. 1–6. http://dx.doi.org/10.1109/MeMeA.2019.8802162.

[66] Qian S, Ugurlu D, Fairweather E, Toso LD, Deng Y, Strocchi M, Cicci L, Jones RE, Zaidi H, Prasad S, et al. Developing cardiac Digital Twin populations powered by machine learning provides electrophysiological insights in conduction and repolarization. Nat Cardiovasc Res 2025;4(5):624–36. http://dx.doi.org/10.1038/s44161-025-00650-0.

[67] Noeikham P, Buakum D, Sirivongpaisal N. Architecture designing of Digital Twin in a healthcare unit. Health Informatics J 2024;30(4):14604582241296792. http://dx.doi.org/10.1177/14604582241296792.