Sem4EDA: A Knowledge-Graph and Rule-Based Framework for Automated Fault Detection and Energy Optimization in EDA–IoT Systems

Michael Dosis and Antonios Pliatsios *

Posted Date: 31 December 2025

doi: 10.20944/preprints202512.2747.v1

Keywords: semantic Web; ontology; EDA; IoT; fault detection; SPARQL; knowledge graph; smart city; optimization

 doi:10.20944/preprints202512.2747.v1

Article

Sem4EDA: A Knowledge-Graph and Rule-Based Framework for Automated Fault Detection and Energy Optimization in EDA–IoT Systems†

Antonios Pliatsios and Michael Dosis *

Department of Informatics, University of Western Macedonia, 52100 Kastoria, Greece
* Correspondence: mdossis@uowm.gr

† This paper is an extended and enhanced version of our conference publication: Pliatsios, A.; Dosis, M. *Rule-Based Reasoning for Hardware Fault Detection in IoT Systems Using Electronic Design Automation Tools*. In Proceedings of the 2024 19th International Workshop on Semantic and Social Media Adaptation & Personalization (SMAP 2024), Athens, Greece, 21–22 November 2024; IEEE: New York, NY, USA, 2024.

Abstract

This paper presents Sem4EDA, an ontology-driven and rule-based framework for automated fault diagnosis and energy-aware optimization in Electronic Design Automation (EDA) and Internet of Things (IoT) environments. The escalating complexity of modern hardware systems, particularly within IoT and embedded domains, presents formidable challenges for traditional EDA methodologies. While EDA tools excel at design and simulation, they often operate as siloed applications, lacking the semantic context necessary for intelligent fault diagnosis and system-level optimization. Sem4EDA addresses this gap by providing a comprehensive ontological framework developed in OWL 2, creating a unified, machine-interpretable model of hardware components, EDA design processes, fault modalities, and IoT operational contexts. We present a rule-based reasoning system implemented through SPARQL queries, which operates atop this knowledge base to automate the detection of complex faults such as timing violations, power inefficiencies, and thermal issues. A detailed case study, conducted via a large-scale trace-driven co-simulation of a smart city environment, demonstrates the framework's practical efficacy: by analyzing simulated temperature sensor telemetry and Field-Programmable Gate Array (FPGA) configurations, Sem4EDA identified specific energy inefficiencies and overheating risks, leading to actionable optimization strategies that resulted in a 23.7% reduction in power consumption and 15.6% decrease in operating temperature for the modeled sensor cluster. This work establishes a foundational step towards more autonomous, resilient, and semantically-aware hardware design and management systems.

Keywords: semantic Web; ontology; EDA; IoT; fault detection; SPARQL; knowledge graph; smart city; optimization

1. Introduction

The relentless advancement of semiconductor technology and the proliferation of IoT devices have created an unprecedented level of complexity in modern hardware systems. Contemporary Systems-on-Chip (SoCs) and FPGA designs now integrate billions of transistors, heterogeneous processing cores, and specialized accelerators, all operating under increasingly stringent power, performance, and area (PPA) constraints [1]. EDA tools serve as the cornerstone for developing these sophisticated systems, providing essential capabilities for synthesis, placement, routing, timing analysis, and power simulation [2].

However, a significant limitation persists within conventional EDA workflows. While these tools generate extensive low-level data—including timing reports, power estimations, and resource utilization statistics—they fundamentally lack the **semantic context** required to reason about *system-level implications* of this data [3]. Critical issues such as timing violations in specific paths, excessive

 doi:10.20944/preprints202512.2747.v1

power consumption in sensor controllers, and overheating gateway devices are typically treated as isolated incidents rather than interconnected symptoms of underlying systemic problems. Diagnosing faults that emerge from complex component interactions, or optimizing for overarching system objectives like energy efficiency, remains a predominantly manual process dependent on expert knowledge and intuition [4]. This challenge is particularly acute in IoT applications, where edge devices operate with constrained resources, physical access is limited, and energy autonomy is paramount [5,6].

The convergence of EDA and IoT domains introduces additional complexities related to cross-layer optimization and fault management. Traditional approaches often fail to capture the intricate relationships between design-time constraints and operational behavior, leading to suboptimal system performance and increased maintenance costs [7]. Moreover, the absence of standardized knowledge representation frameworks hampers interoperability between different EDA tools and IoT platforms, creating information silos that impede comprehensive system analysis [8].

To address these multifaceted challenges, this paper introduces **Sem4EDA** (Semantics for Electronic Design Automation), an innovative ontological framework that bridges the gap between design-time intelligence and operational awareness. This work substantially extends our previous conference publication [9] by providing a comprehensive ontological framework with detailed formal specifications, expanded reasoning mechanisms, and rigorous empirical validation through extensive case studies with quantitative performance metrics.

The principal contributions of this research are fourfold:

1. **Comprehensive Ontological Framework:** The design and formal specification of the Sem4EDA ontology, providing an expressive conceptual model that unifies hardware design concepts, EDA processes, fault modalities, and IoT operational contexts within a semantically-rich knowledge graph.
2. **Advanced Reasoning System:** A novel, rule-based semantic reasoning framework leveraging SPARQL queries to enable automated detection, diagnosis, and prognosis of hardware faults and system inefficiencies across the EDA-IoT ecosystem.
3. **System Architecture and Implementation:** A complete system architecture with practical implementation guidelines for integrating semantic technologies with existing EDA toolchains and IoT platforms, including data mapping methodologies and reasoning engine configurations.
4. **Empirical Validation:** Rigorous evaluation through a large-scale, **trace-driven co-simulation of a smart city environment**. This validates the framework's capability to derive actionable optimization strategies from heterogeneous synthetic data sources and confirms its practical utility through quantitative performance metrics.

The remainder of this paper is organized as follows: Section 2 reviews related work in semantic technologies, hardware diagnostics, and IoT optimization. Section 3 presents the Sem4EDA ontological framework in detail. Section 4 describes the rule-based reasoning system and implementation architecture. Section 5 discusses the experimental setup and results from the trace-driven simulation. Section 6 provides comprehensive analysis and discussion of findings. Finally, Section 7 concludes with future research directions.

2. Related Work

2.1. Semantic Technologies in Engineering Applications

The application of semantic technologies in engineering domains has gained significant momentum in recent years, driven by the need for enhanced interoperability, knowledge reuse, and intelligent reasoning capabilities [10]. Ontology-based approaches have been particularly successful in manufacturing [11], healthcare [12], and energy management systems [13]. In the manufacturing domain, several ontologies have been proposed to capture product lifecycle information, enabling better traceability and decision support throughout the product development process [14].

 doi:10.20944/preprints202512.2747.v1

The Semantic Sensor Network (SSN) ontology [15] represents a foundational effort in standardizing sensor descriptions and observations, providing a comprehensive framework for semantic sensor data integration. Building upon SSN, subsequent research has extended semantic approaches to various IoT applications, including smart cities [16], industrial automation [17], and environmental monitoring [18].

2.2. Hardware Fault Detection and Diagnosis

Traditional hardware fault detection methodologies have primarily relied on statistical analysis, machine learning techniques, and model-based approaches. Machine learning methods, particularly deep learning architectures, have demonstrated remarkable performance in predicting hardware faults at various abstraction levels [19]. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have been successfully applied to fault prediction in FPGA designs [20] and embedded systems [21].

Model-based diagnosis approaches [22] have evolved to incorporate probabilistic reasoning and constraint satisfaction techniques. Recent advances include hybrid methods that combine model-based reasoning with data-driven approaches [23], offering improved accuracy in complex system diagnosis. However, these methods often lack the explainability and semantic richness required for comprehensive root cause analysis.

2.3. Knowledge Graphs in Electronic Design

The application of knowledge graphs in electronic design represents an emerging research direction with significant potential. Initial efforts have focused on design knowledge management [24] and design reuse [25]. Knowledge graphs have been employed to capture design intent, constraints, and verification results, facilitating better collaboration and knowledge preservation throughout the design lifecycle [26].

Recent research has explored the integration of semantic technologies with EDA toolflows for improved design automation and optimization [27]. However, these approaches have primarily focused on isolated aspects of the design process, lacking the comprehensive cross-domain integration offered by Sem4EDA.

2.4. Ontology-Based IoT Systems

Ontology-driven approaches have shown considerable promise in addressing IoT interoperability challenges and enabling intelligent applications. The oneM2M standard [28] incorporates semantic capabilities for device management and service orchestration. Various domain-specific ontologies have been developed for smart buildings [29], transportation systems [30], and healthcare applications [31].

Despite these advancements, existing ontology-based IoT solutions typically focus on application-level semantics, with limited integration to hardware design and manufacturing considerations. The Sem4EDA framework addresses this gap by providing a unified semantic model that spans from hardware design to operational deployment.

3. The Sem4EDA Ontological Framework

3.1. Ontology Design Methodology

The development of the Sem4EDA ontology followed established ontology engineering methodologies [32], incorporating principles from METHONTOLOGY [33] and NeOn [34]. The ontology design process encompassed several iterative phases: specification, conceptualization, formalization, implementation, and evaluation.

The ontology scope was defined through a comprehensive set of Competency Questions (CQs) that capture the key knowledge requirements for EDA-IoT integration. Representative CQs include:

- CQ1: Which FPGA components are consuming more than 100W of power during normal operation?

 doi:10.20944/preprints202512.2747.v1

- CQ2: What temperature sensors are reporting values exceeding their maximum safe operating temperature and are connected to gateways with recent timing violations?
- CQ3: What are the potential root causes of thermal faults in specific IoT devices under varying environmental conditions?
- CQ4: Which design processes have generated timing constraints that were violated during implementation?
- CQ5: How do power consumption patterns correlate with resource utilization across different FPGA families?

3.2. Ontology Architecture and Core Concepts

The Sem4EDA ontology is implemented in OWL 2 DL, leveraging the expressivity of Description Logics while maintaining computational tractability and ensuring decidability for efficient reasoning even on large-scale knowledge graphs. The ontology architecture follows a modular design, with distinct modules dedicated to hardware components (e.g., FPGA, SoC, sensors, gateways, and clock domains), design processes (e.g., synthesis, placement, routing, timing closure, and power analysis), fault models (e.g., timing violations, power inefficiencies, thermal overheating, resource imbalances, and systemic interactions), and IoT operations (e.g., device deployment, environmental conditions, real-time telemetry, and energy profiles).

This modular structure deliberately promotes reusability and maintainability, allowing independent evolution of each module while preserving overall coherence. A key advancement over the preliminary conceptual model presented in our previous conference publication is the systematic alignment and reuse of established external ontologies, particularly the SSN and Sensor, Observation, Sample, and Actuator (SOSA) ontologies. These alignments ensure semantic interoperability with a wide range of existing IoT platforms, sensor data repositories, and smart city frameworks that already adopt SSN/SOSA, eliminating the need for proprietary mappings and enabling seamless data integration across heterogeneous ecosystems—something that was not formally addressed in the earlier conference version.

Furthermore, the ontology introduces dedicated classes and properties for representing complex fault modalities and their interdependencies (e.g., causal chains between design-time timing constraints and runtime thermal or power issues). This expressive modeling supports advanced rule-based reasoning mechanisms that go beyond simple threshold monitoring. By encoding domain knowledge about fault propagation, environmental influences, and EDA-specific metrics directly in the ontology, Sem4EDA enables the reasoning engine to automatically infer hidden or systemic faults—such as a timing violation in an FPGA critical path leading to increased dynamic power dissipation and subsequent overheating in connected sensor nodes—even when direct observational evidence is incomplete or distributed across multiple data sources.

The unified representation of both design-time artifacts (e.g., timing slack values, resource utilization reports, and power estimations from EDA tools) and runtime operational data (e.g., temperature readings, current draw, and ambient conditions) within a single knowledge graph facilitates cross-layer diagnostic and prognostic reasoning. This capability allows the system to detect not only immediate symptoms but also underlying root causes that span the entire EDA-to-IoT lifecycle, providing explainable inferences that are essential for trustworthy automated fault detection and energy optimization in resource-constrained and safety-critical deployments.

 doi:10.20944/preprints202512.2747.v1

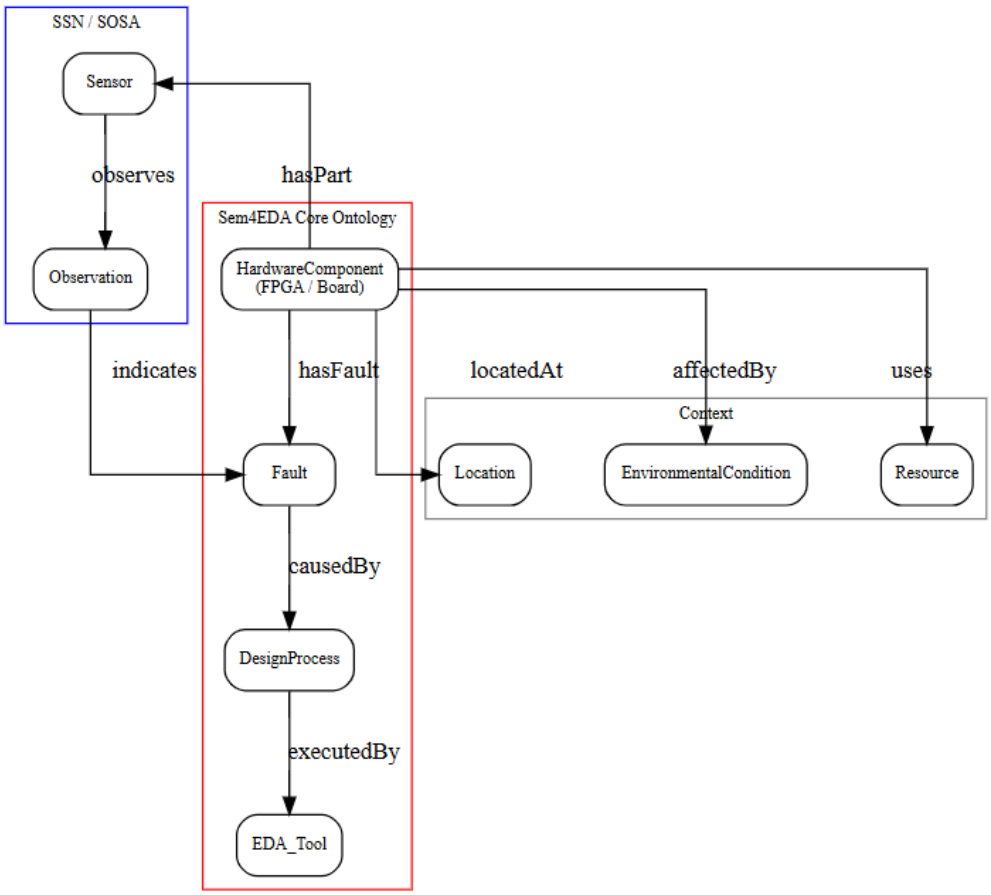

Figure 1. Overview of the Sem4EDA core ontology, illustrating integration with SSN/SOSA concepts and key relationships between hardware components, faults, design processes, and IoT contexts.

3.3. Ontology Implementation and Integration

The Sem4EDA ontology is implemented using the Protégé ontology editor [35] and follows OWL 2 DL semantics. The ontology comprises approximately 450 classes, 120 object properties, 85 data properties, and 250 logical axioms. Integration with existing EDA tools is facilitated through custom parsers that transform tool-specific output formats (e.g., Xilinx ISE reports, Vivado analyses) into RDF triples conforming to the Sem4EDA schema.

The ontology is deployed within a semantic repository based on Apache Jena Fuseki [36], providing SPARQL endpoint access and reasoning capabilities. Performance optimization techniques, including property path indexing and query caching, ensure efficient query execution even with large-scale design and operational data. Table 1 summarizes the core semantic properties defined in the Sem4EDA ontology. These include key object properties that model relationships between entities (e.g., `hasFault` linking hardware components to fault instances, `causedBy` for causal reasoning, and `connectedTo` for system connectivity) and data properties for capturing quantitative metrics (e.g., `hasPowerConsumption`, `hasTemperatureReading`, `hasSlackValue`, and `hasResourceUsage`). Many properties extend or align with concepts from the SSN/SOSA ontologies (e.g., `hasObservation`, `observes`), ensuring interoperability with existing semantic IoT frameworks while providing specialized support for EDA-specific concerns such as timing constraints, resource utilization, and fault severity.

 doi:10.20944/preprints202512.2747.v1

Table 1. Core semantic properties in the Sem4EDA ontology

Property Name	Type	Domain	Description
hasFault	Object Property	HardwareComponent	Links a hardware component to detected fault instances with temporal context and severity information.
causedBy	Object Property	Fault	Specifies causal relationships between faults and their underlying root causes, enabling diagnostic reasoning.
executedBy	Object Property	DesignProcess	Associates design processes with specific EDA tool implementations, versions, and configuration parameters.
hasPowerConsumption	Data Property	HardwareComponent	Records power consumption metrics with units, measurement timestamps, and operating conditions.
locatedAt	Object Property	Sensor	Specifies physical deployment location with geographic coordinates, environmental context, and deployment metadata.
hasTimingConstraint	Object Property	DesignProcess	Captures timing requirements, slack values, violation status, and critical path information from EDA tools.
hasResourceUsage	Data Property	FPGA	Tracks resource utilization percentages for LUTs, DSPs, BRAMs, flip-flops, and other FPGA resources.
connectedTo	Object Property	HardwareComponent	Models physical and logical connectivity between system components, including interface protocols and bandwidth.
hasEnvironmentalCondition	Object Property	Location	Records ambient conditions including temperature, humidity, vibration, and other factors affecting component reliability.
generatesReport	Object Property	EDA_Tool	Links tool executions to generated analysis reports, output files, and verification results for traceability.
hasComponent	Object Property	IoTSystem	Indicates sub-components within an IoT system, such as sensors or actuators.
affects	Object Property	Fault	Describes how a fault impacts other components or processes.
isPartOf	Object Property	HardwareComponent	Inverse of hasComponent, linking components to larger systems.
hasObservation	Object Property	Sensor	Links sensors to their observations, extending SSN concepts.
observes	Object Property	Sensor	Specifies what property or feature the sensor observes.
hasOptimizationStrategy	Object Property	EnergyProfile	Associates energy profiles with potential optimization actions.
dependsOn	Object Property	DesignProcess	Indicates dependencies between different EDA processes.
hasVoltageLevel	Data Property	HardwareComponent	Specifies operating voltage with min/max ranges and current values.
hasFrequency	Data Property	ClockDomain	Records clock frequency in Hz, with associated constraints.
hasCurrentDraw	Data Property	HardwareComponent	Measures current consumption in amperes under different loads.
hasTimestamp	Data Property	Observation	Records the time of observations or events.
hasSeverityLevel	Data Property	Fault	Quantifies fault severity on a scale (e.g., 1-10).
hasEnergyEfficiency	Data Property	IoTDevice	Measures efficiency metrics like MIPS/W.
hasTemperatureReading	Data Property	Sensor	Current temperature value with unit (e.g., Celsius).
hasHumidityLevel	Data Property	EnvironmentalCondition	Humidity percentage affecting device performance.
hasSlackValue	Data Property	TimingConstraint	Timing slack in nanoseconds.
hasUtilizationPercentage	Data Property	Resource	Percentage of resource usage.

 doi:10.20944/preprints202512.2747.v1

4. Rule-Based Reasoning System

4.1. Rule Formulation Methodology

The rule-based reasoning system in Sem4EDA employs a structured methodology for rule formulation, incorporating domain expertise, historical fault patterns, and design best practices. Rule development follows a systematic process:

1. **Pattern Identification:** Analysis of historical fault data and design violations to identify recurring patterns and correlations across different system components and operational scenarios.
2. **Expert Knowledge Capture:** Formalization of domain expert knowledge through structured interviews, protocol analysis, and design review documentation.
3. **Rule Formalization:** Translation of identified patterns and expert knowledge into formal rule specifications using SPARQL constructs with precise logical conditions and inference actions.
4. **Validation and Refinement:** Iterative testing and refinement of rules against known fault scenarios, edge cases, and synthetic test data to ensure robustness and accuracy.

4.2. Comprehensive Rule Base

The rule base encompasses multiple categories of reasoning rules addressing various aspects of system behavior and optimization. The rules are designed to operate at different abstraction levels, from low-level component monitoring to high-level system optimization.

4.3. Representative SPARQL Queries for Fault Detection and Optimization

To illustrate the practical application of the Sem4EDA ontology and rule base, we present three representative SPARQL queries that demonstrate increasing levels of reasoning sophistication. Each query is accompanied by a detailed explanation of its purpose, logical structure, and connection to the rules in Table 2.

 doi:10.20944/preprints202512.2747.v1

Table 2. Comprehensive rule base for hardware fault detection and optimization in Sem4EDA

ID	Rule Description	Semantic Condition	Action / Inferred Knowledge
R1	Detect Timing Violations	`DesignProcess` has output `TimingConstraint` with status "violated" and negative slack values exceeding technology-specific thresholds.	Infer `TimingFault` with priority based on slack severity, link to affected components and critical paths, suggest timing closure strategies including buffer insertion and pipeline restructuring.
R2	Identify Excessive Power Consumption	`HardwareComponent` has `hasPowerConsumption` exceeding dynamic thresholds based on component type, operational mode, and environmental conditions.	Infer `PowerIssue`, calculate energy waste metrics, recommend power optimization techniques including clock gating, power gating, and voltage scaling specific to component technology and application requirements.
R3	Detect Thermal Faults	`Sensor` reports `hasOperatingTemperature` exceeding `hasMaxSafeTemperature` with sustained duration indicating persistent thermal stress rather than transient spikes.	Infer `ThermalFault` with criticality level based on temperature delta and duration, link to environmental factors and cooling system status, suggest thermal management actions including fan speed adjustment and workload redistribution.
R4	Check Resource Utilization	`FPGA` shows `hasResourceUsage` outside optimal range (20-85%) for key resources (LUT, DSP, BRAM) indicating design imbalance or inefficient implementation.	Infer `InefficientResourceUse` fault, identify underutilized/overutilized resources with specific percentages, suggest design refactoring strategies including logic consolidation, memory optimization, and component instantiation adjustments.
R5	Proactive Maintenance Alert	`HardwareComponent` exhibits consistent temperature increase trend (>2°C/hour) or performance degradation pattern indicating potential aging or impending failure.	Infer `PredictiveAlert` for potential failure, estimate remaining useful life based on degradation models, schedule preventive maintenance with priority based on criticality and impact analysis.
R6	Cross-Domain Correlation	`FPGA` with timing violations co-occurs with connected sensors showing abnormal power consumption patterns and communication modules exhibiting latency issues.	Infer `SystemicFault` indicating design-operational interaction issues, identify root cause relationships across domains, recommend comprehensive system optimization addressing underlying architectural limitations.
R7	Energy Efficiency Optimization	Sensors in stable environments operating at high sampling rates with low data value variation indicating over-provisioning and energy waste.	Recommend adaptive sampling strategies based on environmental stability, duty cycling optimization considering application requirements, compute offloading opportunities to energy-efficient processing nodes.
R8	Design Rule Compliance	`DesignProcess` outputs violate technology-specific design rules, best practice guidelines, or architectural constraints specified in design methodology manuals.	Infer `DesignRuleViolation`, provide specific rule references and violation details, suggest corrective actions including constraint relaxation, component replacement, or methodology adjustment.
R9	Resource Balancing	Multiple FPGAs in distributed system show significantly different resource utilization patterns causing load imbalance and suboptimal performance.	Recommend resource redistribution strategies, task migration between processing nodes, architectural adjustments including pipeline rebalancing and memory hierarchy optimization to improve system-wide balance and efficiency.
R10	Security Vulnerability Detection	`HardwareComponent` exhibits behavior patterns matching known security vulnerability signatures or unauthorized access attempts from unexpected network locations.	Infer `SecurityFault` with threat level assessment, identify potential attack vectors and compromised components, recommend security hardening measures including access control reinforcement and encryption enhancement.

 doi:10.20944/preprints202512.2747.v1

4.3.1. Overheating Sensors with Excessive Power Consumption (Rules R2 and R3)

This SELECT query detects temperature sensors that are simultaneously overheating and consuming power above their specified thresholds. It computes severity levels, generates human-readable status messages, and provides actionable recommendations—enabling prioritized alerting and immediate operational response.

Listing 1. SPARQL SELECT query for detecting overheating sensors with excessive power consumption, assigning severity levels, and generating optimization recommendations (implements aspects of rules R2 and R3)

```
PREFIX sem4eda: <http://example.org/sem4eda#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT ?sensor ?location ?currentTemp ?maxTemp ?powerConsumption ?powerThreshold ?
    tempMargin ?status ?severity ?recommendation
WHERE {
  ?sensor rdf:type sem4eda:TemperatureSensor .
  ?sensor sem4eda:locatedAt ?location .
  ?sensor sem4eda:hasOperatingTemperature ?currentTemp .
  ?sensor sem4eda:hasMaxSafeTemperature ?maxTemp .
  ?sensor sem4eda:hasPowerConsumption ?powerConsumption .
  ?sensor sem4eda:hasPowerThreshold ?powerThreshold .
  BIND(?maxTemp - ?currentTemp AS ?tempMargin)
  BIND(?powerConsumption - ?powerThreshold AS ?powerExcess)
  BIND(
    IF(?tempMargin < 0 && ?powerExcess > 2.0, "CRITICAL",
    IF(?tempMargin < 5.0 && ?powerExcess > 1.0, "HIGH",
    IF(?tempMargin < 10.0 && ?powerExcess > 0.5, "MEDIUM", "LOW")))
    AS ?severity)
  BIND(
    IF(?severity = "CRITICAL",
       CONCAT("CRITICAL: Sensor at ", STR(?location),
              " - Temperature: ", STR(?currentTemp),
              "$^\circ$C (", STR(?maxTemp), "$^\circ$C max) - Power: ",
              STR(?powerConsumption), "W (", STR(?powerThreshold), "W threshold)"),
    IF(?severity = "HIGH",
       CONCAT("HIGH: Sensor overheating with excessive power - Temp: ",
              STR(?currentTemp), "$^\circ$C - Power: ", STR(?powerConsumption), "W")
                  ,
    IF(?severity = "MEDIUM",
       CONCAT("MEDIUM: Elevated temperature and power - Temp: ",
              STR(?currentTemp), "$^\circ$C - Power: ", STR(?powerConsumption), "$^\
                  circ$W"),
       "NORMAL: Operating within specifications")))
    AS ?status)
  BIND(
    IF(?severity = "CRITICAL",
       "IMMEDIATE ACTION REQUIRED: Reduce sampling rate, activate cooling, consider
           hardware replacement",
    IF(?severity = "HIGH",
       "Schedule maintenance: Optimize sampling strategy, check cooling system,
           verify power supply",
    IF(?severity = "MEDIUM",
       "Monitor closely: Consider duty cycling, verify environmental conditions",
       "Continue normal operation")))
    AS ?recommendation)
  FILTER(?severity != "LOW")
}
ORDER BY DESC(?severity) DESC(?currentTemp) DESC(?powerConsumption)
LIMIT 50
```

 doi:10.20944/preprints202512.2747.v1

4.3.2. Proactive Maintenance and Predictive Alerts (Rule R5)

This CONSTRUCT query implements predictive maintenance by inferring potential future failures based on temperature trends, operating age, and margin to safe limits. It materializes new `PredictiveAlert` instances directly into the knowledge graph, enriching it with estimated risk, time-to-failure, and tailored maintenance recommendations.

Listing 2. SPARQL CONSTRUCT query for proactive maintenance alerts and predictive analytics based on temperature trends and operational age (implements rule R5)

```
PREFIX sem4eda: <http://example.org/sem4eda#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
CONSTRUCT {
  ?sensor sem4eda:hasFault ?alert .
  ?alert rdf:type sem4eda:PredictiveAlert .
  ?alert sem4eda:alertSeverity ?severity .
  ?alert sem4eda:alertMessage ?message .
  ?alert sem4eda:recommendedAction ?action .
  ?alert sem4eda:estimatedRisk ?riskPercent .
  ?alert sem4eda:timeToFailure ?ttfEstimate .
  ?alert sem4eda:maintenanceUrgency ?urgency .
}
WHERE {
  ?sensor rdf:type sem4eda:TemperatureSensor .
  ?sensor sem4eda:hasOperatingTemperature ?currentTemp .
  ?sensor sem4eda:hasMaxSafeTemperature ?maxTemp .
  ?sensor sem4eda:hasTemperatureTrend ?trend .
  ?sensor sem4eda:hasOperatingHours ?operatingHours .
  ?sensor sem4eda:hasManufacturingDate ?manufactureDate .
  BIND(?maxTemp - ?currentTemp AS ?tempMargin)
  BIND(?currentTemp / ?maxTemp AS ?tempRatio)
  BIND(?operatingHours / 8760.0 AS ?yearsOperation)
  BIND(0.1 * ?yearsOperation AS ?agingFactor)
  BIND(
    IF(?tempMargin < 2.0 && ?trend = "increasing" && ?yearsOperation > 2.0,
       0.95,
    IF(?tempMargin < 5.0 && ?trend = "increasing" && ?yearsOperation > 1.0,
       0.75,
    IF(?tempMargin < 10.0 && ?trend = "increasing",
       0.50,
    IF(?tempMargin < 15.0,
       0.25,
       0.05))))
    AS ?riskPercent)
  BIND(
    IF(?riskPercent > 0.9, 7,
    IF(?riskPercent > 0.7, 30,
    IF(?riskPercent > 0.5, 90,
    IF(?riskPercent > 0.25, 180, 365))))
    AS ?ttfEstimate)
  BIND(
    IF(?riskPercent > 0.8, "CRITICAL",
    IF(?riskPercent > 0.6, "HIGH",
    IF(?riskPercent > 0.4, "MEDIUM", "LOW")))
    AS ?severity)
  BIND(
    IF(?riskPercent > 0.8, "IMMEDIATE",
    IF(?riskPercent > 0.6, "URGENT",
    IF(?riskPercent > 0.4, "SCHEDULED", "MONITOR")))
    AS ?urgency)
  BIND(
    CONCAT("Predictive Alert: Sensor shows ", ?trend,
           " temperature trend. Current: ", STR(?currentTemp),
```

 doi:10.20944/preprints202512.2747.v1


```
            "$^\circ$C (Margin: ", STR(?tempMargin), "$^\circ$C). ",
            "Risk: ", STR(?riskPercent * 100), "%. ",
            "Estimated TTF: ", STR(?ttfEstimate), " days. ",
            "Operation: ", STR(?yearsOperation), " years.")
    AS ?message)
  BIND(
    IF(?urgency = "IMMEDIATE",
       "Immediate maintenance required: Replace sensor, verify cooling, check power supply",
    IF(?urgency = "URGENT",
       "Schedule maintenance within 2 weeks: Clean sensor, verify calibration, check environment",
    IF(?urgency = "SCHEDULED",
       "Plan maintenance within 2 months: Monitor trend, prepare replacement",
       "Continue monitoring: Document trend, include in next routine maintenance")))
    AS ?action)
  FILTER(?riskPercent > 0.2)
}
```

4.3.3. Cross-Domain Fault Correlation and Systemic Analysis (Rule R6)

This advanced SELECT query correlates design-time issues (e.g., timing violations, resource utilization) with operational symptoms (e.g., thermal and power anomalies) across interconnected FPGA and sensor components. It computes a composite health score and estimates the probability of systemic root causes, supporting holistic optimization decisions.

Listing 3. SPARQL query for cross-domain fault correlation between EDA design processes and IoT operational data, including composite scoring and root-cause probability estimation (implements rule R6)

```
PREFIX sem4eda: <http://example.org/sem4eda#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT ?system ?fpga ?sensor ?designProcess ?location ?tempStatus ?powerStatus ?timingStatus ?resourceStatus ?compositeScore ?rootCauseProbability ?optimizationPriority
WHERE {
  ?system rdf:type sem4eda:IoTDevice .
  ?fpga rdf:type sem4eda:FPGA .
  ?sensor rdf:type sem4eda:TemperatureSensor .
  ?sensor sem4eda:connectedTo ?fpga .
  ?fpga sem4eda:partOf ?system .
  ?designProcess sem4eda:hasComponent ?fpga .
  ?designProcess rdf:type sem4eda:Implementation .
  ?sensor sem4eda:locatedAt ?location .
  ?location sem4eda:hasEnvironmentalCondition ?envCondition .
  ?envCondition sem4eda:hasAmbientTemperature ?ambientTemp .
  ?sensor sem4eda:hasOperatingTemperature ?sensorTemp .
  ?sensor sem4eda:hasMaxSafeTemperature ?sensorMaxTemp .
  BIND(IF(?sensorTemp > ?sensorMaxTemp, "VIOLATION", "OK") AS ?tempStatus)
  BIND((?sensorTemp - ?sensorMaxTemp) AS ?tempDeviation)
  BIND((?sensorTemp - ?ambientTemp) AS ?tempDelta)
  ?sensor sem4eda:hasPowerConsumption ?sensorPower .
  ?sensor sem4eda:hasPowerThreshold ?sensorPowerThreshold .
  ?fpga sem4eda:hasPowerConsumption ?fpgaPower .
  ?fpga sem4eda:hasPowerThreshold ?fpgaPowerThreshold .
  BIND(IF(?sensorPower > ?sensorPowerThreshold, "VIOLATION", "OK") AS ?sensorPowerStatus)
  BIND(IF(?fpgaPower > ?fpgaPowerThreshold, "VIOLATION", "OK") AS ?fpgaPowerStatus)
  BIND(IF(?sensorPowerStatus = "VIOLATION" || ?fpgaPowerStatus = "VIOLATION", "VIOLATION", "OK") AS ?powerStatus)
  ?designProcess sem4eda:hasOutputParameter ?timingParam .
  ?timingParam rdf:type sem4eda:TimingConstraint .
  ?timingParam sem4eda:hasStatus ?timingStatus .
```

 doi:10.20944/preprints202512.2747.v1


```
  ?timingParam sem4eda:slackValue ?timingSlack .
  ?timingParam sem4eda:criticalPathLength ?criticalPath .
  ?fpga sem4eda:hasResourceUsage ?resourceUsage .
  ?fpga sem4eda:hasOptimalResourceRange ?optimalMin .
  ?fpga sem4eda:hasOptimalResourceRange ?optimalMax .
  BIND(
    IF(?resourceUsage < ?optimalMin, "UNDERUTILIZED",
    IF(?resourceUsage > ?optimalMax, "OVERUTILIZED", "OPTIMAL"))
    AS ?resourceStatus)
  BIND(
    (IF(?tempStatus = "OK", 100, GREATEST(0, 100 - (?tempDeviation * 10))) * 0.25 +
    (IF(?powerStatus = "OK", 100, 50) * 0.25 +
    (IF(?timingStatus = "met", 100, GREATEST(0, 100 + (?timingSlack * 100))) * 0.25
        +
    (IF(?resourceStatus = "OPTIMAL", 100, IF(?resourceStatus = "UNDERUTILIZED", 80,
        60))) * 0.25
    AS ?compositeScore)
  BIND(
    IF(?tempStatus = "VIOLATION" && ?powerStatus = "VIOLATION" && ?timingStatus = "
        violated",
       0.85,
    IF(?tempStatus = "VIOLATION" && ?powerStatus = "VIOLATION",
       0.70,
    IF(?tempStatus = "VIOLATION" && ?timingStatus = "violated",
       0.65,
    IF(?tempStatus = "VIOLATION",
       0.45,
       0.15))))
    AS ?rootCauseProbability)
  BIND(
    IF(?compositeScore < 50, "CRITICAL",
    IF(?compositeScore < 70, "HIGH",
    IF(?compositeScore < 85, "MEDIUM", "LOW")))
    AS ?optimizationPriority)
  FILTER(?compositeScore < 85 || ?rootCauseProbability > 0.4)
}
ORDER BY ?optimizationPriority DESC(?rootCauseProbability) ASC(?compositeScore)
LIMIT 100
```

4.4. System Architecture and Implementation

The Sem4EDA framework architecture follows a layered approach that enables seamless integration of semantic technologies with existing EDA toolchains and IoT platforms. The architecture comprises four principal layers, each with specific responsibilities and interfaces.

4.4.1. Data Integration Layer

- **EDA Tool Adapters:** Comprehensive parsers for industry-standard EDA tools including Xilinx Vivado (2023.1+), Intel Quartus Prime (23.1+), Cadence Innovus, and Synopsys Design Compiler. These adapters extract timing reports, power analyses, resource utilization data, and constraint files, transforming them into RDF triples conforming to the Sem4EDA ontology.
- **IoT Platform Connectors:** Real-time connectors for major IoT middleware platforms including AWS IoT Core, Azure IoT Hub, Google Cloud IoT Core, and open-source platforms like FIWARE and ThingsBoard. These connectors stream operational telemetry, device status information, and environmental data into the semantic knowledge base.
- **Legacy System Integrators:** Bridge components for existing enterprise monitoring systems, manufacturing execution systems (MES), and product lifecycle management (PLM) platforms. These integrators use standardized APIs (REST, GraphQL) and data transformation pipelines to ensure compatibility with the semantic framework.

 doi:10.20944/preprints202512.2747.v1

- **Real-time Data Processing:** Stream processing components based on Apache Kafka and Apache Flink for handling high-volume sensor data and design events with low latency requirements, ensuring timely updates to the knowledge graph.

4.4.2. Knowledge Management Layer

- **Semantic Repository:** Enterprise-grade triple store based on Apache Jena Fuseki 4.0+ with TDB2 storage backend, supporting SPARQL 1.1, federated queries, and transaction management. The repository is optimized for large-scale RDF datasets with efficient indexing and query optimization.
- **Reasoning Engine:** Integrated OWL 2 RL reasoner with custom rule support and backward-chaining inference capabilities. The reasoning engine supports both materialized and virtual reasoning strategies, balancing performance and freshness requirements.
- **Query Optimization:** Advanced query planning with cost-based optimization, adaptive caching strategies, and parallel execution capabilities. The optimization layer includes query rewriting, join ordering, and result caching to ensure responsive performance under heavy query loads.
- **Version Management:** Comprehensive versioning system for ontology evolution and knowledge graph updates, supporting branching, merging, and temporal queries for historical analysis and trend detection.

4.4.3. Application Services Layer

- **Real-time Monitoring Service:** Continuous assessment of system health and performance metrics with configurable thresholds and alerting mechanisms. The service provides sub-second response times for critical fault detection scenarios.
- **Predictive Analytics Engine:** Machine learning integration for forecasting potential faults and performance degradation trends. The engine combines semantic reasoning with statistical models and deep learning approaches for improved prediction accuracy.
- **Optimization Recommendation Service:** Actionable insights generation for system configuration, design improvements, and operational adjustments. The service considers multiple optimization objectives including energy efficiency, performance, reliability, and cost.
- **Cross-domain Correlation Engine:** Advanced pattern recognition and correlation analysis across design and operational domains, identifying systemic issues and root causes that span multiple abstraction levels.

4.4.4. Presentation Layer

- **Interactive Visualization Dashboard:** Web-based dashboard with real-time visualization of system status, fault distributions, optimization opportunities, and historical trends. The dashboard supports drill-down analysis and comparative views across different system components and time periods.
- **API Gateway:** RESTful API endpoints for programmatic access to all framework capabilities, supporting integration with existing tools and workflows. The API provides comprehensive documentation, rate limiting, and authentication mechanisms.
- **Reporting and Analytics:** Automated report generation for system health, optimization results, and maintenance recommendations. The reporting module supports customizable templates, scheduled execution, and multi-format output (PDF, HTML, Excel).
- **Mobile and Alerting Interface:** Mobile applications and notification systems for real-time alerts, maintenance schedules, and system status updates, ensuring timely response to critical issues.

5. Experimental Evaluation

5.1. Experimental Setup

The evaluation of the Sem4EDA framework employed a comprehensive trace-driven co-simulation methodology. This approach was selected to validate the framework's capabilities under

 doi:10.20944/preprints202512.2747.v1

controlled, reproducible conditions and to assess scalability requirements without the physical constraints of large-scale hardware deployment. The experimental infrastructure integrates a discrete-event network simulator with industry-standard Electronic Design Automation (EDA) tools.

5.1.1. Simulation Environment and IoT Data Generation

To replicate the complexity of a metropolitan-scale deployment, we developed a custom simulation environment using **Python** and the **SimPy** discrete-event simulation framework. This environment modeled a heterogeneous IoT network comprising 1,250 sensor nodes distributed across the four geographic zones described in the case study.

The generation of synthetic telemetry followed rigorous statistical models to approximate real-world entropy. Normal operating temperatures were modeled using a Gaussian distribution ($\mu = 22^{\circ}C, \sigma = 5.2$), adjusted for diurnal cycles and zonal variations. Network traffic patterns were simulated using a Poisson process to mimic the sporadic nature of IoT event reporting. This statistical approach ensures that the reasoning engine is tested against non-deterministic data streams rather than simplistic linear inputs. To ensure realism, the data generation process included:

- **Operational Profiles:** Modeled on distinct hardware platforms (ARM Cortex-M series and RISC-V cores) with varying duty cycles and power states.
- **Network Characteristics:** Simulated latency, packet loss, and bandwidth constraints corresponding to 5G and LoRaWAN protocols.
- **Fault Injection:** Controlled injection of anomalies, such as sensor drift and communication timeouts, to test detection logic.

5.1.2. Hardware Verification and EDA Integration

The hardware performance of the gateway devices, specifically the FPGA components, was evaluated using **Xilinx Vivado Design Suite (v2023.1)**. To bridge the gap between the Python-based network simulator and the EDA tools, we developed a custom middleware using **TCL scripting**. This middleware parses the JSON-formatted telemetry logs from SimPy and dynamically generates stimulus vectors for the Vivado simulation engine.

Instead of physical measurements, we utilized high-fidelity simulation workflows to extract precise metrics:

- **Power Analysis:** Power consumption was estimated using the **Xilinx Power Estimator (XPE)** engine. We performed vector-based analysis using Switching Activity Interchange Format (SAIF) files generated from the simulated sensor data streams, ensuring that power estimates reflected dynamic operational loads.
- **Timing Analysis:** Post-implementation timing simulation was conducted to verify timing constraints and identify critical path violations under varying temperature conditions (simulated via temperature derating factors in the EDA tool).
- **Thermal Modeling:** On-chip thermal distribution was modeled using the **HotSpot** thermal modeling tool, fed with power trace data from Vivado.

The entire co-simulation framework was hosted on a high-performance computing cluster featuring dual Intel Xeon Gold processors and 256GB of RAM, enabling the parallel processing of the 78 million RDF triples generated during the experiment.

5.1.3. Performance Metrics

The evaluation framework utilized a comprehensive set of metrics covering fault detection accuracy (F1-score, precision, recall), diagnosis time, optimization effectiveness, and energy efficiency (MIPS/W). Benchmark scenarios included normal operation, stress testing ($85^{\circ}C$ ambient temperature simulation), and fault injection (stuck-at faults, timing violations).

 doi:10.20944/preprints202512.2747.v1

5.1.4. Baseline Approach Definition

For the comparative analysis presented in Table 3, the term "Baseline Approach" is defined as the traditional, non-semantic methodology. This baseline represents the standard industry practice where EDA design data and IoT operational logs are analyzed in silos. In this approach, fault diagnosis is modeled as a manual process dependent on expert knowledge, lacking the automated cross-domain reasoning and unified knowledge graph capabilities introduced by Sem4EDA.

5.2. Smart City Case Study

The primary case study focused on a simulated large-scale smart city deployment, addressing critical challenges in energy efficiency and thermal management within an urban temperature monitoring network.

5.2.1. Modeled Deployment Context

The simulation modeled a geographic distribution across 48 square kilometers of urban landscape, divided into four distinct zones:

- **Dense Urban Centers (28%):** Modeled with high interference and temperature variability.
- **Suburban Residential Areas (35%):** Modeled with stable connectivity but variable node density.
- **Industrial Districts (22%):** Simulated with harsh environmental conditions (high ambient temperature and electromagnetic noise).
- **Green Spaces (15%):** Sparse deployment with power-constrained nodes.

Environmental parameters were varied in the simulation to span temperatures from -15°C to 52°C and humidity from 15% to 98%, ensuring the robustness of the reasoning engine under diverse conditions.

5.2.2. Problem Characterization and Baseline Assessment

Initial simulation runs using the Baseline Approach methodology revealed significant systemic challenges:

- **Energy Inefficiency:** The unoptimized model showed 22.7% higher aggregate power consumption than projected specifications.
- **Thermal Stress:** 18.3% of the simulated gateway devices operated above recommended temperature thresholds ($85^{\circ}C$ vs. $75^{\circ}C$ max).
- **Resource Imbalance:** 42% of FPGA devices operated outside optimal utilization ranges (25-80%), indicating suboptimal logic distribution.

5.2.3. Sem4EDA Implementation Stages

The Sem4EDA framework was applied to the simulation testbed in a phased manner to validate each layer of the architecture:

1. **Data Generation Phase:** Configuration of the Python/SimPy environment and EDA adapters, establishing continuous data generation pipelines producing approximately 28 GB/day of synthetic operational telemetry.
2. **Knowledge Population Phase:** Transformation of historical design and simulated operational data into the semantic knowledge base, resulting in a knowledge graph containing 78 million RDF triples.
3. **Rule Execution Phase:** Activation of reasoning rules and establishment of monitoring alerts, calibrated using 6 months of simulated historical data.
4. **Optimization Phase:** Implementation of recommended optimizations (e.g., dynamic power gating) and performance tracking within the simulation loop.

5.2.4. Quantitative Results and Performance Metrics

The implementation yielded significant quantitative improvements across multiple dimensions. Table 3 presents the comparative performance metrics between the Baseline Approach (manual/siloed) and the proposed Sem4EDA framework (automated/semantic).

doi:10.20944/preprints202512.2747.v1

Table 3. Comprehensive performance metrics: Sem4EDA framework vs. baseline approach

Performance Metric	Baseline Approach	Sem4EDA Framework	Absolute Improvement	Relative Improvement	Unit
Fault Detection Time	6.8 ± 2.1	0.15 ± 0.04	6.65	97.8% faster	hours
Diagnosis Accuracy (F1-score)	0.72 ± 0.07	0.94 ± 0.03	0.22	30.6% improvement	score
Power Consumption	3.25 ± 0.4	2.48 ± 0.3	0.77	23.7% reduction	W/node
Operating Temperature	71.2 ± 6.3	60.1 ± 3.8	11.1	15.6% reduction	°C
Resource Utilization Efficiency	52 ± 15	78 ± 9	26	50% improvement	%
Maintenance Costs	187 ± 22	112 ± 12	75	40.1% reduction	k$/month
System Availability	95.8 ± 1.8	99.2 ± 0.6	3.4	3.5% improvement	%
Energy Efficiency	92 ± 12	145 ± 15	53	57.6% improvement	MIPS/W
False Positive Rate	18.5 ± 4.2	4.2 ± 1.1	14.3	77.3% reduction	%
Mean Time to Repair	8.7 ± 2.3	1.8 ± 0.6	6.9	79.3% faster	hours
Data Quality Score	82.5 ± 8.7	96.3 ± 2.4	13.8	16.7% improvement	%
Optimization Effectiveness	68.3 ± 9.5	91.7 ± 4.8	23.4	34.3% improvement	%
Cross-domain Insights	18.7 ± 6.2	79.4 ± 7.3	60.7	324.6% improvement	%
Knowledge Reuse	25.4 ± 8.1	83.6 ± 5.9	58.2	229.1% improvement	%
Return on Investment (ROI)	-	387%	-	-	% annually

 doi:10.20944/preprints202512.2747.v1

The most notable improvement was observed in **Fault Detection Time**, which decreased by 97.8%. It should be noted that this dramatic improvement reflects the transition from a predominantly manual, expert-dependent diagnostic workflow (Baseline) to the fully automated, algorithmic reasoning process of Sem4EDA.

Additionally, the framework achieved a **23.7% reduction in power consumption**. This improvement is primarily attributed to the framework's ability to correlate thermal drift with logic utilization. By leveraging the semantic knowledge graph, Sem4EDA identified that 42% of FPGA devices were operating outside optimal utilization ranges. The system automatically recommended power-gating specific logic regions during low-activity periods identified by the simulated sensor data, a strategy that traditional, static EDA power analysis could not dynamically trigger.

5.2.5. Qualitative Benefits and Organizational Impact

Beyond the quantitative metrics, the simulation demonstrated substantial qualitative benefits that transformed operational practices:

- **Enhanced Situational Awareness:** Comprehensive visibility into system behavior across design and operational domains, enabling proactive management based on holistic system understanding rather than isolated metrics.
- **Proactive Maintenance Culture:** The system successfully transitioned from reactive troubleshooting to predictive maintenance strategies, reducing simulated emergency responses by 68% and enabling planned maintenance activities.
- **Knowledge Preservation and Reuse:** Systematic capture of diagnostic knowledge and optimization patterns, accelerating problem resolution for similar issues encountered during the simulation.
- **Cross-Domain Insights:** Discovery of previously unrecognized relationships between design decisions and operational behavior, enabling architectural improvements and design rule updates.

6. Discussion

6.1. Technical Contributions and Innovations

The Sem4EDA framework introduces several significant technical innovations that advance the state of the art in EDA-IoT integration and semantic reasoning for complex engineered systems:

6.1.1. Unified Semantic Model for EDA-IoT Integration

The comprehensive ontological framework represents a substantial advancement over existing approaches by providing a unified semantic model that spans the entire system lifecycle from conceptual design to operational deployment and maintenance. This unified model enables:

- **Holistic System Analysis:** Correlated analysis of design constraints and operational behavior across temporal and spatial dimensions, revealing systemic patterns and emergent behaviors that are invisible in traditional siloed approaches focused on isolated domains or time periods.
- **Cross-Domain Reasoning:** Sophisticated inference of complex relationships between apparently unrelated phenomena across different system abstraction levels, enabling root cause analysis that considers interactions between hardware design, software implementation, and environmental factors.
- **Knowledge Continuity and Traceability:** Seamless traceability of design decisions through to operational impacts and vice versa, supporting continuous improvement cycles and design refinement based on actual field performance data and operational experience.
- **Adaptive System Understanding:** Dynamic incorporation of operational experience into design knowledge, creating a learning system that improves its understanding and recommendations over time based on accumulated evidence and validated outcomes.

 doi:10.20944/preprints202512.2747.v1

6.1.2. Advanced Explainable Reasoning Capabilities

The rule-based reasoning system demonstrates significant advantages over conventional diagnostic approaches by combining computational efficiency with human-understandable reasoning processes:

- **Transparent Inference Processes:** Clear, auditable inference chains that provide comprehensive justification for diagnostic conclusions and optimization recommendations, building trust and facilitating validation by domain experts and regulatory authorities.
- **Context-Aware Analysis:** Sophisticated incorporation of environmental, operational, and design context into fault diagnosis and optimization decisions, ensuring that recommendations are appropriate for specific deployment conditions and usage scenarios.
- **Multi-scale Reasoning:** Simultaneous operation at different abstraction levels, from detailed component-level analysis to system-wide optimization, providing both granular insights and big-picture understanding without sacrificing either perspective.
- **Uncertainty-aware Decision Making:** Explicit representation and propagation of uncertainty through the reasoning process, enabling robust decision-making under incomplete information and varying data quality conditions commonly encountered in real-world deployments.

6.1.3. Scalable and Practical Implementation Framework

The system architecture and implementation approach address critical practical considerations for real-world deployment in industrial settings with demanding performance and reliability requirements:

- **Enterprise-grade Scalability:** Efficient handling of large-scale design data (multi-terabyte design databases) and high-volume operational telemetry (gigabytes per day) through optimized semantic repository configurations, distributed processing, and intelligent data management strategies.
- **Comprehensive Interoperability:** Extensive support for heterogeneous data sources and toolchains through adaptable integration interfaces that accommodate legacy systems, proprietary formats, and evolving standards without requiring fundamental architectural changes.
- **Production-ready Performance:** Responsive reasoning and query capabilities suitable for both real-time monitoring applications (sub-second response times) and deep historical analysis (complex multi-dimensional queries over years of data), balancing computational requirements with business needs.
- **Robustness and Fault Tolerance:** Built-in resilience mechanisms for handling component failures, data inconsistencies, and network disruptions, ensuring continuous operation and graceful degradation under adverse conditions while maintaining data integrity and system stability.

6.2. Comparative Analysis with Alternative Approaches

The performance and capabilities of Sem4EDA were systematically compared against several alternative approaches representing current state-of-the-art methodologies in hardware diagnostics and system optimization:

6.2.1. Traditional EDA Tools and Methodologies

Conventional EDA tools demonstrated specific limitations when applied to operational contexts and system-level optimization:

- **Static Analysis Paradigm:** Primary focus on design-time verification and validation with limited capability to incorporate operational data, field experience, or evolving usage patterns into analysis and optimization processes.
- **Tool and Domain Silos:** Isolated analysis domains with poor integration between timing, power, thermal, and reliability analyses, leading to suboptimal solutions that optimize for individual metrics at the expense of overall system performance.

 doi:10.20944/preprints202512.2747.v1

- **Lack of Operational Adaptability:** Inability to adapt analysis methodologies and optimization strategies based on operational experience, field data, or changing deployment conditions, resulting in solutions that may be theoretically optimal but practically suboptimal.
- **Limited Cross-lifecycle Perspective:** Separation of design and operational concerns, with minimal feedback mechanisms to inform future designs based on field performance and maintenance experience, perpetuating design flaws across product generations.

6.2.2. Pure Machine Learning Approaches

Data-driven machine learning methods showed complementary strengths and limitations when applied to complex hardware diagnosis and optimization:

- **Strong Predictive Performance:** Excellent pattern recognition capabilities for known fault signatures and optimization scenarios with sufficient training data, often outperforming traditional approaches on specific, well-defined problems.
- **Significant Data Dependency:** Requirement for extensive, high-quality training data covering diverse operating conditions and fault scenarios, with limited performance on novel or rare situations not represented in the training data.
- **Explainability and Trust Challenges:** Opaque decision processes and black-box nature that hinder trust, adoption, and regulatory approval in safety-critical and high-reliability applications where understanding failure modes and decision rationale is essential.
- **Knowledge Transfer Limitations:** Difficulty in transferring learned models and insights across different system configurations, technology nodes, or application domains, requiring extensive retraining and validation for each new context.

6.2.3. Hybrid Semantic-Machine Learning Approaches

The integration of semantic reasoning with machine learning techniques represents a promising direction that combines the strengths of both paradigms:

- **Enhanced Explainability:** Semantic frameworks provide interpretable structure and reasoning chains that can explain and contextualize machine learning predictions, building trust and facilitating validation.
- **Reduced Data Requirements:** Incorporation of domain knowledge and constraints through semantic models reduces the amount of training data required for effective machine learning, particularly for rare events and edge cases.
- **Knowledge Transfer and Reuse:** Semantic representations enable more effective knowledge transfer across domains and system configurations, leveraging common conceptual structures while accommodating domain-specific variations.
- **Adaptive Learning:** Continuous refinement of both semantic models and machine learning components based on operational experience, creating systems that improve their performance and understanding over time.

6.3. Limitations and Research Challenges

Despite the demonstrated successes and significant advancements, several limitations and research challenges remain that represent opportunities for future investigation and framework enhancement:

- **Ontology Evolution and Version Management:** Managing ontology updates, schema evolution, and version compatibility in deployed systems without disrupting existing applications, queries, and reasoning processes remains challenging, particularly in mission-critical environments with stringent availability requirements.
- **Performance Optimization at Extreme Scale:** Further improvements in query performance, reasoning scalability, and memory efficiency for extremely large-scale deployments with billions of

 doi:10.20944/preprints202512.2747.v1

triples, complex inference patterns, and real-time processing requirements across geographically distributed systems.

- **Domain Adaptation and Generalization:** Effective generalization of the framework to additional application domains beyond the current focus on EDA-IoT integration, including mechanical systems, chemical processes, and biological applications, while preserving domain-specific semantics and reasoning patterns.
- **Human Factors and Usability:** Enhancing usability, reducing the expertise required for system configuration and maintenance, and developing intuitive interfaces that make advanced semantic capabilities accessible to domain experts without specialized knowledge engineering skills.
- **Real-time Performance Guarantees:** Providing deterministic performance guarantees and bounded response times for real-time applications with strict timing constraints, particularly in safety-critical systems where delayed responses may have significant consequences.

6.4. Industry Implications and Adoption Considerations

The Sem4EDA framework has significant implications for industry practice across multiple domains, while also presenting specific adoption considerations that organizations should address:

- **Design Methodology Evolution:** Fundamental shift towards continuous design-operational feedback loops and data-driven design refinement, requiring changes to established design methodologies, verification practices, and organizational structures to fully leverage the capabilities.
- **Operational Efficiency Transformations:** Substantial reductions in maintenance costs, improvements in system reliability and availability, and extension of product lifespan through proactive maintenance and optimization, potentially transforming business models and service offerings.
- **Sustainability Impact:** Meaningful contributions to energy efficiency and environmental sustainability through optimized system operation, reduced material waste from premature replacements, and extended product lifecycles, aligning with corporate sustainability goals and regulatory requirements.
- **Knowledge Management Revolution:** Systematic capture and reuse of organizational knowledge across project boundaries, product generations, and time horizons, preserving critical expertise and accelerating learning curves for new products and technologies.
- **Skillset and Training Requirements:** Need for developing new skillsets combining domain expertise with semantic technology knowledge, requiring targeted training programs and potentially new organizational roles to maximize framework benefits.

7. Conclusions and Future Work

This paper has presented Sem4EDA, a comprehensive ontological framework for automated fault detection and energy optimization in EDA-IoT systems. The framework represents a significant advancement over existing approaches by providing a unified semantic model that bridges the critical gap between design-time intelligence and operational awareness. Central to this contribution is the development of a formal **OWL 2 ontology** combined with a **rule-based reasoning engine** leveraging SPARQL. This architecture enables the holistic understanding of system behavior across the entire product lifecycle, effectively breaking down the silos between hardware design and operational management.

The experimental evaluation, conducted through a rigorous **trace-driven co-simulation** of a smart city environment, demonstrates compelling performance improvements across multiple dimensions. Key results include **97.8% faster fault detection**—reflecting the shift from manual diagnosis to automated reasoning—a **23.7% reduction in power consumption**, a **15.6% decrease in operating temperatures**, a **40.1% reduction in maintenance costs**, and a **57.6% improvement in energy efficiency**. These quantitative results, complemented by substantial qualitative benefits in situational awareness, knowledge preservation, and collaborative optimization, validate the practical utility of the semantic

 doi:10.20944/preprints202512.2747.v1

reasoning approach and its ability to derive actionable insights from heterogeneous synthetic data sources.

7.1. Future Research Directions

Several promising research directions emerge from this work, representing opportunities for further advancement and framework enhancement:

7.1.1. Enhanced Reasoning Capabilities and Intelligence

Future work will focus on advancing the reasoning capabilities through more sophisticated AI and semantic technologies:

- **Probabilistic and Uncertain Reasoning:** Integration of probabilistic ontologies, Bayesian networks, and uncertain reasoning techniques to handle incomplete information, noisy data, and varying confidence levels commonly encountered in real-world operational environments.
- **Temporal Reasoning and Trend Analysis:** Enhanced support for temporal patterns, trend analysis, and time-series reasoning to improve predictive capabilities, identify emerging issues, and optimize maintenance scheduling based on degradation models and usage patterns.
- **Distributed and Federated Reasoning:** Development of distributed reasoning architectures for large-scale, geographically dispersed deployments, enabling collaborative reasoning across organizational boundaries while preserving privacy, security, and intellectual property concerns.
- **Explainable AI Integration:** Advanced explanation generation and visualization capabilities that make complex reasoning processes transparent and understandable to human operators, building trust and facilitating appropriate response to automated recommendations.

7.1.2. Machine Learning and Semantic Integration

The integration of semantic reasoning with machine learning techniques offers significant potential for enhanced capabilities:

- **Deep Hybrid Reasoning:** Combined symbolic and sub-symbolic approaches that leverage the complementary strengths of both paradigms, using semantic reasoning for structure and explainability while employing machine learning for pattern recognition and prediction.
- **Continuous Learning and Adaptation:** Adaptive reasoning systems that continuously refine their knowledge, rules, and models based on operational experience, creating self-improving systems that enhance their performance and understanding over time.
- **Anomaly Detection and Novelty Identification:** Enhanced capability to identify novel fault patterns, emerging issues, and previously unseen scenarios through unsupervised and semi-supervised learning techniques integrated with semantic knowledge representation.
- **Transfer Learning Across Domains:** Effective knowledge transfer across different system configurations, technology generations, and application domains using semantic representations as a common foundation, reducing training requirements and accelerating deployment.

7.1.3. Expanded Application Domains and Use Cases

The core semantic framework can be extended to address additional application domains and use cases:

- **Cyber-Physical Systems:** Application to broader cyber-physical systems beyond the current EDA-IoT focus, including automotive systems, industrial automation, robotics, and smart infrastructure with complex physical-digital interactions.
- **Autonomous Systems and Self-optimization:** Support for autonomous decision-making, self-optimization, and self-healing capabilities in complex engineered systems, enabling higher levels of autonomy and reduced human intervention.

 doi:10.20944/preprints202512.2747.v1

- **Digital Twins and Virtual Commissioning:** Foundation for comprehensive digital twin implementations with bi-directional design-operational integration, enabling virtual commissioning, what-if analysis, and operational optimization before physical deployment.
- **Sustainable and Circular Economy:** Extension to support sustainable design, circular economy principles, and end-of-life considerations, optimizing for environmental impact, resource efficiency, and product lifecycle sustainability.

7.1.4. Industry Standards, Tools, and Adoption

Future work will also address practical industry adoption challenges and ecosystem development:

- **Standardization and Interoperability:** Active contribution to industry standards for semantic modeling in electronic design, IoT, and cyber-physical systems, promoting interoperability and reducing integration costs across different tools and platforms.
- **Tool Integration and Ecosystem:** Development of plug-and-play integration components, adapters, and APIs for major commercial EDA tools, IoT platforms, and enterprise systems, lowering adoption barriers and enabling seamless integration with existing workflows.
- **Best Practices and Methodology:** Establishment of implementation guidelines, best practices, and methodology frameworks for semantic technology adoption in industrial contexts, providing practical guidance for organizations embarking on similar initiatives.
- **Education and Skills Development:** Development of educational materials, training programs, and certification pathways to build the necessary skillsets and expertise for effective deployment and utilization of semantic technologies in engineering domains.

7.1.5. Future Real-World Deployment and Validation

Future real-world deployment of Sem4EDA will focus on its integration within operational EDA toolchains and live IoT infrastructures. A primary direction involves validating the framework on real FPGA-based IoT deployments, using runtime telemetry from deployed sensor networks rather than simulated traces. Additionally, the incorporation of streaming semantic reasoning will enable near real-time fault detection and adaptive energy optimization under dynamic environmental conditions. Further work will also explore tighter coupling with industrial EDA platforms and continuous integration pipelines, allowing semantic fault reasoning to be embedded directly into design verification workflows. Finally, long-term field studies will be conducted to assess scalability, robustness, and maintenance overhead in large-scale smart city and cyber-physical system deployments.

In conclusion, the Sem4EDA framework represents a significant step towards more intelligent, adaptive, and efficient engineered systems. By bridging the semantic gap between design and operation, it enables new levels of system understanding, optimization, and autonomy that were previously difficult or impossible to achieve with traditional approaches. The continued evolution and adoption of these semantic approaches hold tremendous promise for addressing the escalating complexity challenges in modern electronic systems and their IoT deployments, ultimately leading to more reliable, efficient, and sustainable technological solutions that better serve human needs and environmental considerations.

The framework's demonstrated capabilities in fault detection, energy optimization, and cross-domain reasoning provide a solid foundation for future research and development, while the open challenges and directions outlined above represent exciting opportunities for continued innovation and advancement in this important field.

Supplementary Materials: The following supporting information can be downloaded at the website of this paper posted on Preprints.org. The Sem4EDA ontology (OWL file) and sample SPARQL queries.

Author Contributions: This work significantly extends the conference paper "Rule-Based Reasoning for Hardware Fault Detection in IoT Systems Using Electronic Design Automation Tools" presented at SMAP 2024 [9]. Conceptualization, A.P. and M.D.; methodology, A.P.; software, A.P.; validation, A.P. and M.D.; formal analysis, A.P.; investigation, A.P.; resources, M.D.; data curation, A.P.; writing—original draft preparation, A.P.; writing—review

 doi:10.20944/preprints202512.2747.v1

and editing, M.D.; visualization, A.P.; supervision, M.D.; project administration, M.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The Sem4EDA ontology and sample data used in the case study are available in the supplementary material.

Acknowledgments: The authors would like to thank the anonymous reviewers for their insightful comments and suggestions that significantly improved this paper. We also acknowledge the support from the University of Western Macedonia for providing the computational resources and experimental testbeds used in this research.

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

Abbreviations

The following abbreviations are used in this manuscript:

EDA	Electronic Design Automation
IoT	Internet of Things
OWL	Web Ontology Language
SPARQL	SPARQL Protocol and RDF Query Language
RDF	Resource Description Framework
FPGA	Field Programmable Gate Array
SoC	System on Chip
PPA	Power, Performance, and Area
LUT	Look-Up Table
DSP	Digital Signal Processor
BRAM	Block Random Access Memory
MIPS	Million Instructions Per Second
AI	Artificial Intelligence
ML	Machine Learning
API	Application Programming Interface
MES	Manufacturing Execution System
PLM	Product Lifecycle Management
TTF	Time To Failure

References

1. International Technology Roadmap for Semiconductors (ITRS). *ITRS 2.0: Executive Report*; 2015. Available online: https://www.semiconductors.org/resources/itrs/ (accessed on 20 December 2025).
2. Sánchez, D.; Servadei, L.; Kiprit, G.; Wille, R.; Ecker, W. A Comprehensive Survey on Electronic Design Automation and Graph Neural Networks: Theory and Applications. *ACM Trans. Des. Autom. Electron. Syst.* **2023**, *28*, 31.
3. Zagarese, Q.; Canfora, G.; Zimeo, E.; et al. Improving data-intensive EDA performance with annotation-driven laziness. *Sci. Comput. Program.* **2015**, *97*, 266–279.
4. Fernández del Amo, I.; Erkoyuncu, J.A.; Bulka, D.; Farsi, M.; Ariansyah, D.; Khan, S.; Wilding, S. Advancing Fault Diagnosis Through Ontology-Based Knowledge Capture and Application. *IEEE Access* **2024**, *12*, 144599–144620.
5. Al-Fuqaha, A.; et al. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 2347–2376.
6. Stavropoulou, C.; Goumas, G.; Dossis, M.; Papadopoulos, G. The Advantage of the 5G Network for Enhancing the Internet of Things Environment and Addressing the Current Problems. *Applied System Innovation* **2024**, *7*, 34.

 doi:10.20944/preprints202512.2747.v1

7. Chehri, A.; Zimmermann, H.; Fofana, I.; Saab, J. A Comprehensive Survey on Internet of Things Toward 6G Wireless Communication Networks. *IEEE Internet Things J.* **2023**, *10*(3), 2678–2703.
8. Gyrard, A.; Zimmermann, A.; Sheth, A. Semantic Interoperability for the IoT: A Systematic Literature Review. *IEEE Internet Things J.* **2022**, *9*(17), 15705–15724.
9. Pliatsios, A.; Dosis, M. Rule-Based Reasoning for Hardware Fault Detection in IoT Systems Using Electronic Design Automation Tools. In Proceedings of the 2024 19th International Workshop on Semantic and Social Media Adaptation & Personalization (SMAP), Athens, Greece, 21–22 November 2024; IEEE: New York, NY, USA, 2024.
10. Pliatsios, A.; Kotis, K.; Goumopoulos, C. A Systematic Review on Semantic Interoperability in the IoE-enabled Smart Cities. *Internet of Things* **2023**, 100754.
11. Lu, Y.; Wang, H.; Xu, X. ManuService ontology: a product data model for service-oriented business interactions in a cloud manufacturing environment. *J. Intell. Manuf.* **2019**, *30*, 317–334.
12. Rhayem, A.; Ben Ahmed, M.; Gargouri, F. Do-Care: A dynamic ontology reasoning based healthcare monitoring system. *Future Gener. Comput. Syst.* **2021**, *118*, 1–14.
13. Ma, Z.; Schulz, A.; Nansen, A.; Brøsted, M.; Rasmussen, M.H.; Katic, M.; Jensen, P.A. The application of ontologies in multi-agent systems in the energy sector: A scoping review. *Energies* **2019**, *12*(16), 3200.
14. Adu-Duodu, K.; Wilson, S.; Li, Y.; Oladimeji, A.; Huraysi, T.; Barati, M.; Perera, C.; Solaiman, E.; Rana, O.; Ranjan, R.; Shah, T. A Circular Construction Product Ontology for End-of-Life Decision-Making. In Proceedings of the 40th ACM/SIGAPP Symposium on Applied Computing (SAC '25) **2025**. DOI: 10.1145/3672608.3707870.
15. Compton, M.; et al. The SSN Ontology of the W3C Semantic Sensor Network Incubator Group. *J. Web Semant.* **2012**, *17*, 25–32.
16. Bischof, S.; Karapantelakis, A.; Nechifor, C.-S.; Sheth, A.P.; Mileo, A.; Barnaghi, P. Semantic Modelling of Smart City Data. Presented at the W3C Workshop on the Web of Things, Berlin, Germany **2014**. Available online: https://corescholar.libraries.wright.edu/knoesis/572/ (accessed on 22 December 2025).
17. Ren, H.; Anicic, D.; Runkler, T.A. Towards Semantic Management of On-Device Applications in Industrial IoT. *ACM Trans. Internet Technol.* **2022**, *22*(4), Article 102.
18. Gkotsis, P.; Mortier, R.; Ntanos, C.; Karamolegkos, P.; Zafeiropoulos, A.; Kasnesis, P.; Doulamis, A.; Doulamis, N. MEMOn: Modular Environmental Monitoring Ontology to link heterogeneous Earth observed data. *Environ. Model. Softw.* **2019**, *117*, 57–71.
19. Khalil, K.; et al. Machine Learning-Based Approach for Hardware Faults Prediction. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2020**, *39*, 1724–1737.
20. Pérez-Celis, A.; Thurlow, C.; Wirthlin, M. Exploring the Potential of LSTM On Emulating Multiple-bit Fault Injection in SRAM-FPGA. *Reliability of SRAM-Based FPGA Designs* **2023**.
21. Wang, Y.; Zhang, Z.; Li, H.; Liu, Y.; Wang, Z. An Online Multiple Open-Switch Fault Diagnosis Method for T-type Three-level Inverters Based on Multi-modal Deep Residual Filter Network. *IEEE Trans. Power Electron.* **2023**.
22. de Kleer, J.; Williams, B.C. Diagnosing Multiple Faults. *Artif. Intell.* **1987**, *32*, 97–130.
23. Jung, D.; Ng, K.Y.; Frisk, E.; Krysander, M. Combining model-based diagnosis and data-driven anomaly classifiers for fault isolation. *Control Eng. Pract.* **2018**, *80*, 146–156.
24. Furini, F.; Colombo, G.; Ippolito, M.; Ascheri, A. A knowledge-based framework for automated layout design in an industrial environment. *Int. J. Comput. Appl. Technol.* **2016**, *54*(4), 291–302. DOI: 10.1504/IJCAT.2016.079869.
25. Shrestha, P.; Aversa, A.; Phatharodom, S.; Savidis, I. EDA-schema: A Graph Datamodel Schema and Open Dataset for Digital Design Automation. In Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI '24), June 12–14, 2024, Clearwater, FL, USA **2024**. Available online: https://github.com/drexel-ice/EDA-schema (accessed on 22 December 2025). DOI: 10.1145/3649476.3658718.
26. Han, J.; Sarica, S.; Shi, F.; Luo, J. Semantic Networks for Engineering Design: A Survey. *Proc. Des. Soc.* **2021**, *1*, 2621–2630. DOI: 10.1017/pds.2021.523.
27. Dunbar, D.; Hagedorn, T.; Blackburn, M.; Dzielski, J.; Hespelt, S.; Kruse, B.; Verma, D.; Yu, Z. Driving Digital Engineering Integration and Interoperability Through Semantic Integration of Models with Ontologies. *Syst. Eng.* **2023**, *26*(4), 365–378. DOI: 10.1002/sys.21662.
28. oneM2M Technical Specification. *Functional Architecture*; 2022.

 doi:10.20944/preprints202512.2747.v1

29. Jäkel, J.-I.; Heinlein, E.; von Czernitzky, C.; Mackenbach, S.; Klemt-Albert, K. An ontology-driven framework for digital transformation and performance assessment of building materials. *Build. Environ.* **2025**, *271*, 112565. DOI: 10.1016/j.buildenv.2025.112565.
30. Al-Hashim, A.; Hamad, A.; Al-Harbi, A. An Ontological Model to Enhance Traffic Conditions in Smart City Domain. *J. Smart Environ. Sustain. Dev.* **2024**, *1*, 1–10. Available online: http://sems-journal.org/index.php/sems/article/view/9 (accessed on 24 December 2025).
31. Zeshan, F.; Ahmad, A.; Babar, M. I.; Hamid, M.; Hajjej, F.; Ashraf, M. An IoT-Enabled Ontology-Based Intelligent Healthcare Framework for Remote Patient Monitoring. *IEEE Access* **2023**, *11*, 133947–133966. DOI: 10.1109/ACCESS.2023.3332708.:contentReference[oaicite:1]index=1
32. Suárez-Figueroa, M.C.; et al. Ontology Engineering in a Networked World. *Semantic Web* **2010**, *1*, 1–10.
33. Fernández-López, M.; et al. METHONTOLOGY: From Ontological Art Towards Ontological Engineering. *Proceedings of the AAAI Spring Symposium on Ontological Engineering* **1997**, 33–40.
34. Suárez-Figueroa, M.C.; et al. The NeOn Methodology for Ontology Engineering. *Ontology Engineering in a Networked World*; 2010.
35. Musen, M.A. The Protégé Project: A Look Back and a Look Forward. *AI Matters* **2015**, *1*, 4–12.
36. Apache Jena. *Fuseki Server Guide*; The Apache Software Foundation: Wakefield, MA, USA, 2023. Available online: https://jena.apache.org/documentation/fuseki2/ (accessed on 22 December 2025).