Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 04 Issue: 03 | Mar-2026
ISSN (O) 3107-6696

Formal Modeling and Verification of an Autonomous Irrigation System Using Petri Nets

Vaishnav Krishna P

Electronics Department, Ming Chi University of Technology (MCUT), Taiwan

---

Abstract - Automated irrigation systems are critical for efficient water and nutrient management in modern agriculture. However, most existing approaches rely on heuristic methods without formal verification, leaving the system prone to deadlocks or inconsistent operation. This paper addresses this research gap by presenting a formal modeling and verification of an automatic irrigation system using Petri Nets. The system autonomously monitors soil moisture and nutrient levels to perform irrigation and fertilization without human intervention. The workflow including sensing, decision-making, actuation, logging, and notification was modeled and simulated using PIPE 4.3.0 and WoPeD 3.8.0, providing both formal analysis and intuitive workflow visualization. Verification of reachability, boundedness, liveness, safeness, and deadlock-freeness confirms reliable system operation, highlighting the relevance of Petri Net-based modeling for robust, scalable autonomous irrigation systems.

Keywords: Autonomous Irrigation System, Petri Nets, Formal Verification, Reachability Analysis,, Deadlock Detection, Boundedness and Liveness, RPA Tools (PIPE, WoPeD)

Introduction

AGRICULTURE is a significant consumer of global fresh-water resources, and efficient irrigation is essential for sustainable crop management and environmental protection. Traditional irrigation practices often lead to water wastage, uneven distribution, and inefficient nutrient application. Re cent advances in smart agriculture emphasize the use of IoT sensors, data analytics, and automated control to improve efficiency and reduce human intervention [1]–[5]. However, many existing approaches rely on heuristic control logic or machine learning models that lack formal correctness guarantees in concurrent operational environments [6], [7]

Formal methods, including model checking and Petri Net analysis, provide mathematical frameworks capable of mod-eling and verifying complex system behaviors under con-currency, deadlocks, and asynchronous events [8]–[10]. Petri Nets, in particular, have been successfully applied to model industrial automation, workflow systems, and distributed IoT processes due to their inherent ability to represent states, transitions, and synchronization explicitly [11], [12].

Automatic irrigation systems can benefit greatly from for- mal modeling, as they often involve interleaved decisions based on multiple environmental inputs (e.g., moisture, nutrients, weather forecasts) and require reliable coordination of actuators such as pumps and valves [13], [14]. Despite this potential, there remains a research gap in formally ver- ifying the correctness properties — such as boundedness, liveness, reachability, safeness, and deadlock-freeness — of autonomous irrigation workflows [2], [3], [15].

This paper bridges that gap by constructing a formal Petri Net model of an autonomous irrigation system and verify- ing its operational properties using state-of-the-art Petri Net tools (PIPE 4.3.0 and WoPeD 3.8.0). The proposed model advances reliability and provides a verified foundation for future large-scale smart agricultural deployments.

1.1 *Objectives*

The main objectives of this research are:

- To model the workflow of an autonomous irrigation system using Petri Nets.
- To formally analyze and verify system properties, ensur- ing correct and deadlock-free operation under multiple conditions.
- To demonstrate the application of formal verification tools for accurate system validation and visualization.

1.2 *Scope and Significance*

This study focuses on autonomously managing irrigation and fertilization based on multi-sensor thresholds and verified Petri Net behavior. By ensuring mathematically guaranteed system correctness, this work contributes to improved reliability, resource efficiency, and decision support in precision agriculture.

1.3 *Paper Organization*

The remainder of the paper is structured as follows: Section 2 reviews related work, Section 3 describes the Petri Net modeling approach, Section 4 presents verification results using formal tools, Section 5 discusses analysis outcomes, and Section 6 concludes with future research directions.

2. Related Work

2.1 Petri Nets for IoT and Smart Systems

Petri Nets are widely adopted for modeling IoT-enabled monitoring and distributed systems because of their precise semantics for concurrency and asynchronous behavior [8]. In smart environments, researchers have leveraged Petri Nets to model and verify control logic, system states, and event dependencies in workflows where correctness and reliability are critical [16], [17]. Extensions such as High-Level Petri Nets and Algebraic Petri Nets further enhance expressiveness for complex IoT system analysis [11].

2.2 Petri Nets in Hybrid and Intelligent Systems

Formal verification using Petri Nets has been explored extensively in cyber-physical systems (CPS), where real-time constraints and safe coordination of physical and logical elements are essential. Techniques for boundedness, reachability, and liveness analysis using Petri Net structures are central to ensuring CPS correctness [19]. In addition, distributed verifi- cation frameworks based on Petri Net formalisms have been proposed to address scalability challenges in large systems

2.3 Formal Verification in Cyber-Physical Systems

Petri Nets have been applied in **embedded and cyber-physical systems (CPS)** to ensure correctness, liveness, and safety in real-time and distributed environments. High-level Petri Nets allow verification of multiple concurrent IoT processes, detecting deadlocks and conflicts before deployment [12], [13], [14].

2.4 Petri Nets in Agriculture and Smart Farming

Workflow nets, a class of Petri Nets, provide powerful for- mal frameworks for modeling business process workflows, enabling reachability and soundness analysis using well-defined criteria [20]. Furthermore, verification of distributed system behaviors through Petri Net models has been studied as a method for ensuring correctness in distributed applications [21].

2.5 Tools and Techniques for Petri Net Verification

Multiple tools and techniques exist to support the formal verification of Petri Net models. Tools like TAPAAL and Romeo provide simulation and model checking for timed and parametric Petri Nets, enabling automated verification of reachability, safety, and boundedness properties [22], [23]. Formal analysis frameworks also focus on optimizing verification processes to mitigate state-space explosion problems in large systems.

2.6 Research Gap and Motivation

While the above literature demonstrates extensive research in Petri Net modeling and verification across CPS, IoT, workflow, and distributed systems, few studies focus specifically on *autonomous irrigation systems* with formal verifica- tion of system properties such as safety, boundedness, and deadlock-freeness under concurrent operational conditions. This paper addresses this gap by combining formal Petri Net analysis with workflow simulation tools such as PIPE and WoPeD.

3. Methodology

3.1 Workflow of Irrigation System

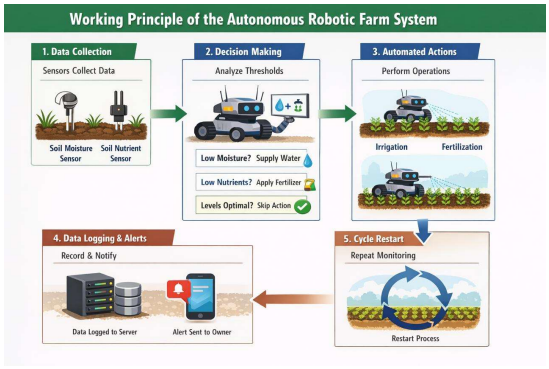

Figure 1: Workflow of the automatic irrigation system.

The proposed system is an automatic irrigation and nutrient management system, designed to monitor the soil and regulate farming processes automatically. It operates as a closed-loop control system without human intervention. Soil moisture and nutrient sensors collect environmental data at the beginning of each monitoring cycle. The system compares the values with predefined threshold values based on crop requirements.

If moisture is low, irrigation is activated; if nutrients are low, fertilization is activated. If both values are normal, no action is taken. All readings and actions are logged on a server, and

Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 04 Issue: 03 | Mar-2026
ISSN (O) 3107-6696

notifications are sent to the farmer's mobile application. The system then returns to monitoring mode.

3.2 System Architecture

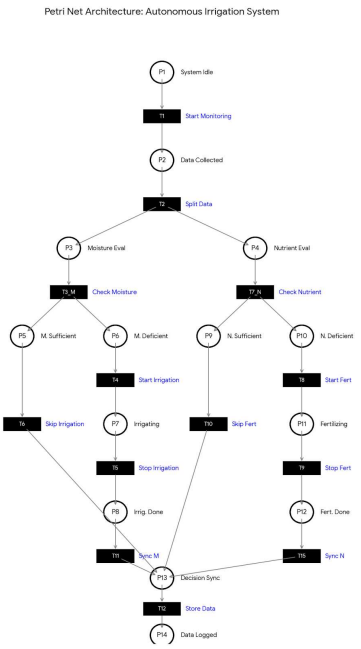

Fig. 2. System architecture of the automatic irrigation and nutrient management system

The architecture of the proposed automatic irrigation system is designed to integrate sensors, controllers, actuators, and the data logging system into a closed-loop framework. Soil moisture and nutrient sensors continuously collect environmental data, which is processed by the control unit. Based on the comparison with predefined thresholds, the system activates irrigation or fertilization as needed. All actions and readings are logged, and notifications are sent to the farmer's mobile application.

Figure 2 shows the overall system architecture, highlighting the flow of data from sensors to controllers, actuators, server, and user interface.

3.3 Algorithm of working

The operation of the automatic irrigation system can be summarized as follows:

1) Initialize the system and define threshold values for soil moisture and nutrients.
2) Start the system.
3) Obtain sensor readings for soil moisture and nutrient levels.
4) Compare the readings with threshold values.
5) If moisture is below the threshold, activate irrigation. If nutrients are below the threshold, activate fertilization.
6) If readings are within acceptable limits, no action is taken.
7) Record readings and operations in the server/database.
8) Send notifications to the farmer via mobile application.
9) Return the system to monitoring mode.

3.4 Modelling with Petrinets

3. 4.1 Definition of Petri Net

The behavior of the proposed automatic irrigation system is modeled using a Petri Net represented as a four-tuple:

PN = (P, T, F, M_0)

where:

- **P** represents the finite set of places describing system states.
- **T** represents the finite set of transitions representing system events.
- **F** represents the flow relation between places and transitions.
- **M_0** represents the initial marking (initial distribution of tokens).

For the proposed model:

- **P = {P_1, P_2, P_3, …, P_{15}}**
- **T = {T_1, T_2, T_3, …, T_{15}}**
- **F ⊆ (P × T) ∪ (T × P)**
- **M_0 = {1, 0, 0, …, 0}**

The initial marking indicates that the system begins in the **idle state** with a token in **P1.**

3.4.2 Transitions and Places

The set of places & Transition represents the operational states of the irrigation system.

Place	Description	Transition	Description
P1	System Idle	T1	Start Monitoring

P2	Sensor data collected	T2	Split sensor data
P3	Moisture evaluation ready	T3	Check Moisture
P4	Nutrient evaluation ready	T4	Start Irrigation
P5	Moisture sufficient	T5	Stop Irrigation
P6	Moisture deficient	T6	Skip Irrigation
P7	Irrigation in progress	T7	Check Nutrient
P8	Irrigation completed	T8	Start Fertilization
P9	Nutrient sufficient	T9	Stop Fertilization
P10	Nutrient deficient	T10	Skip Fertilization
P11	Fertilization in progress	T11	Synchronize branches
P12	Fertilization completed	T12	Store data
P13	Decision synchronization	T13	Send Notification
P14	Data logged to server	T14	Restart Cycle
P15	Notification sent	T15	Synchronize branches

3.4.3 Token Flow Analysis

The token flow represents the operational workflow of the automatic irrigation system. Tokens indicate the current system state and move as transitions fire.

Initialization: A token starts in P1 (System Idle). T1 moves it to P2 (Sensor data collected), then T2 splits it into two parallel branches: moisture (P3) and nutrient (P4).

Moisture Branch:

- P3 → T3 → P5/P6
- If sufficient (P5), T6 → P13.
- If deficient (P6), T4 → P7 → T5 → P8 → T11 → P13.

Nutrient Branch:

- P4 → T7 → P9/P10
- If sufficient (P9), T10 → P13.
- If deficient (P10), T8 → P11 → T9 → P12 → T15 → P13.

Logging & Notification: P13 → T12 → P14 → T13 → P15 → T14 → P1, completing the cycle.

3.4.4 Tools Used for Simulation

The Petri Net model of the automatic irrigation system was simulated using the tools PIPE (Platform Independent Petri Net Editor) version 4.3.0 and WoPeD (Workflow Petri Net Designer) version 3.8.0.

A) **WoPeD:** This tool provided an **intuitive and user-friendly interface** for modeling the system workflow, including the **parallel and synchronized branches** of the irrigation and fertilization processes.

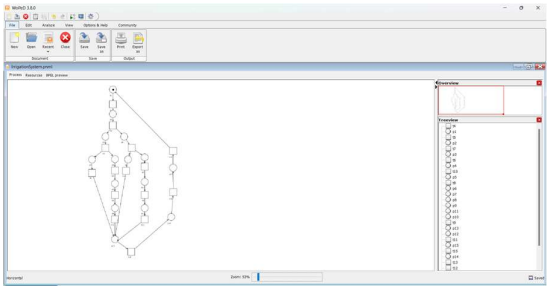

Fig. 3. Automatic irrigation system modeled using WoPeD

A) **PIPE:** This tool was primarily used for the **formal analysis and verification** of system properties, including reachability, boundedness, and liveness.

Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 04 Issue: 03 | Mar-2026
ISSN (O) 3107-6696

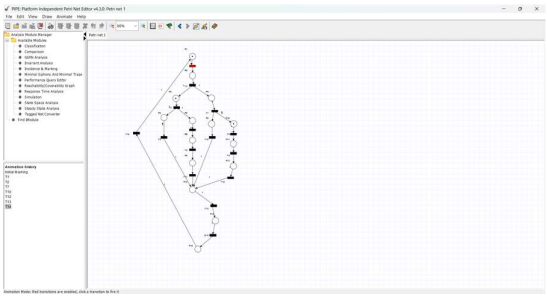

Fig. 4. Automatic irrigation system modeled using PIPE

4. Results

4.1 Checking for Reachability

The reachability of the proposed automatic irrigation system was verified for all operational cases. Each case was analyzed by firing the corresponding sequence of transitions, and it was confirmed that the system returns to the initial state while all relevant states are reachable.

4.1.1 Water Deficiency Case

Firing Sequence: T1 → T2 → T3 → T4 → T5 → T11 → T12 → T13 → T14
In this scenario, the system detects insufficient soil moisture, activates irrigation, completes the process, logs data, sends notification, and returns to the idle monitoring state.

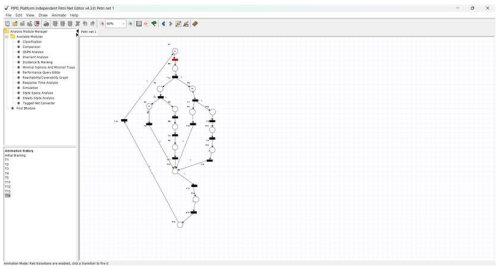

Fig. 5. Firing sequence for the Water Deficiency scenario

4.1.2 Water Sufficiency Case

Firing Sequence: T1 → T2 → T3 → T6 → T12 → T13 → T14
Here, the moisture level is adequate, so the irrigation step is skipped. The system logs data, sends notification, and returns to the initial state.

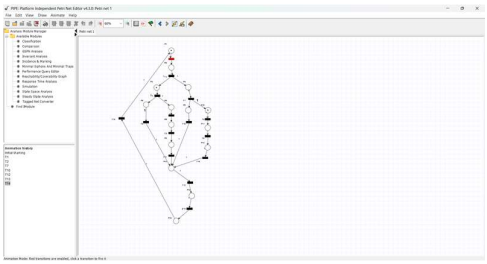

Fig. 6. Firing sequence for the Nutrient Sufficiency scenario

4.1.3 Nutrient Sufficiency Case

Firing Sequence: T1 → T2 → T7 → T10 → T12 → T13 → T14
With sufficient nutrients, fertilization is skipped. Data logging and notification occur before the system returns to the idle state.

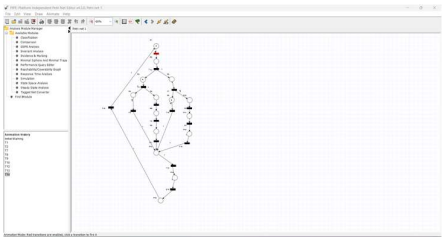

Fig. 7. Firing sequence for the Nutrient Deficiency scenario

4.1.4 Nutrient Deficiency Case

Firing Sequence: T1 → T2 → T7 → T8 → T9 → T15 → T12 → T13 → T14
When nutrients are insufficient, the system activates and completes fertilization, synchronizes branches, logs data, sends notification, and returns to the monitoring state.

Observation: In all cases, the Petri Net successfully returns to the initial state, ensuring all system states are reachable without deadlocks.

4.2 Reachability Graph Analysis

The reachability graph of the automatic irrigation system shows all possible states of the system and how it moves from one state to another through transitions. Each node in the graph represents a unique system state, and each directed edge represents a transition firing.

By examining the reachability graph, we can confirm that all important states, such as idle, irrigation, fertilization, and notification, are reachable from the initial state. This ensures that the system operates correctly and avoids deadlocks.

Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 04 Issue: 03 | Mar-2026
ISSN (O) 3107-6696

Figure 8 illustrates the reachability graph for the proposed system, showing paths for different scenarios including water deficiency, water sufficiency, nutrient deficiency, and nutrient sufficiency. The graph helps validate that the system returns to idle after completing each task.

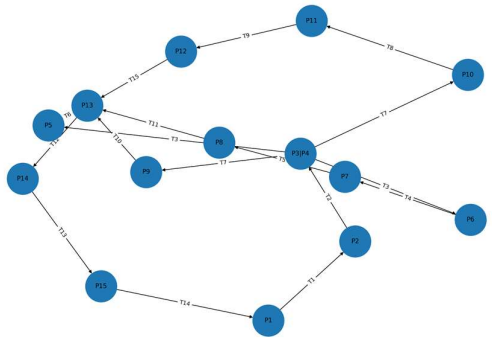

Fig. 8. Reachability graph of the automatic irrigation system

4.3 Checking for Deadlock

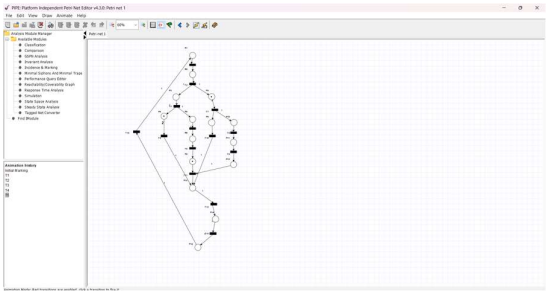

Fig. 9. Deadlock detected in the initial Petri Net design

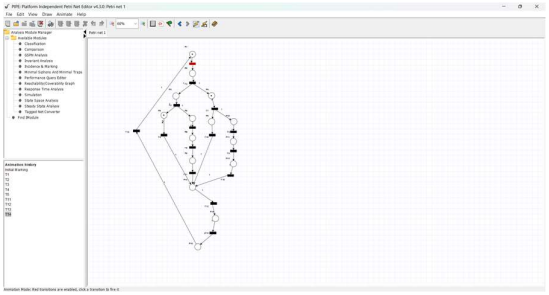

Fig. 10. Deadlock resolved by introducing transition T15 for synchronization

In the reachability and verification stage, each operational scenario was created through manual modeling by carefully increasing tokens according to the conditions required by the system. At first, there was a deadlock in the sequence T1 -> T2 -> T3 -> T4 -> T5, which was caused by the blocking of the sequence since transition T11 needed a token from P12, which was absent at this stage. To resolve this, another transition, T15, was incorporated to ensure synchronization of the parallel paths, thereby allowing token movement. After this, all firing sequences were verified, which supported the fact that the system resets to the initial state without deadlocks and all states become reachable in the Petri Net model.

4.4 Checking for Boundedness

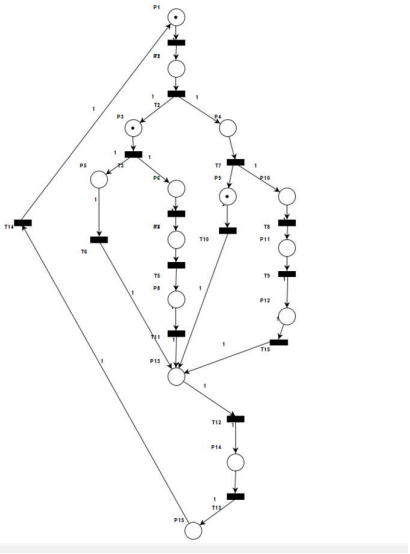

Fig. 11. Petri Net modeled using PIPE 4.3.0 for boundedness verification

The Petri Net of the automatic irrigation system is bounded,as verified using all firing sequences.

For each operational Scenario

Water deficiency (T1→T2→T3→T4→T5→T11→T12→T13→T14)

Water sufficiency (T1→T2→T3→T6→T12→T13→T14)

Nutrient sufficiency (T1→T2→T7→T10→T12→T13→T14)

Nutrient deficiency (T1→T2→T7→T8→T9→T15→T12→T13→T14)

it was observed that no place accumulated more than one token at any point during execution. This indicates that the system is naturally resource-limited, ensuring that no state can overflow with tokens or produce uncontrolled behavior. Boundedness in this context guarantees that each action, such as irrigation or fertilization, occurs in a controlled manner and that the system can maintain stability throughout repeated monitoring cycles. Moreover, the bounded nature of the Petri Net simplifies formal verification, as the maximum number of tokens in each place is known and predictable, allowing safe and reliable execution under all defined operational scenarios.

4.5 Checking for Safeness

Each place in the model was verified to hold at most one token during all firing sequences. This confirms that the Petri Net is safe, aligning with the system requirement that each operation irrigation or fertilization occurs at most once per monitoring cycle. Safeness ensures correct resource usage and prevents simultaneous or repeated triggering of actions.

4.6 Checking for Liveliness

The liveliness of the proposed automatic irrigation system was verified to ensure that all transitions can eventually fire in every scenario, preventing any part of the Petri Net from becoming permanently disabled. This guarantees continuous monitoring, irrigation, and fertilization operations without deadlocks or starvation.

Figure 12 shows the system modeled in WoPeD 3.8.0 for liveliness verification. Each transition in the Petri Net can fire along at least one execution path, confirming that the system is fully live.

To further validate liveliness, a Transition Firing Histogram was generated to show the number of times each transition fires across all workflow execution paths (see Figure 13). This histogram confirms that every transition, including irriga- tion and fertilization operations, is executed at least once under some scenario. Transitions that occur in all scenarios, such as T1 , T2 , T12 , T13 , and T14 , have the highest firing counts, while condition-specific transitions, like T4 , T5 , T6 , T8 , T9 , T10 , T11 , and T15, appear according to scenario requirements.

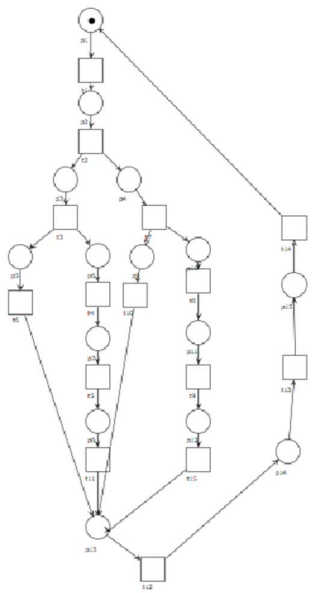

Fig. 12. Petri Net modeled using WoPeD 3.8.0 for liveliness verification

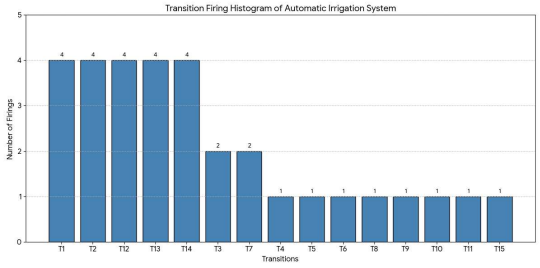

Fig. 13. Transition Firing Histogram showing frequency of execution for all transitions, validating liveliness

The analysis confirms that all transitions are live. No transition becomes permanently disabled, and the system can continuously execute all necessary workflows. The histogram and reachability analysis together demonstrate that the auto- matic irrigation system maintains liveliness in all operational scenarios.

Result: The analysis confirms that all transitions, including irrigation and fertilization operations, are live. No transition becomes permanently disabled, and the system can continuously execute all necessary workflows. The reachability tree in Figure 13 further illustrates that every state is reachable and every transition can eventually fire, validating the liveliness property of the system.

5. Conclusion

In this study, a fully automatic irrigation system was modeled and analyzed using Petri Nets to ensure reliable and continuous farm operation. The system was formally verified for reachability, boundedness, liveness, safeness, and deadlock-freeness using the firing sequences corresponding to various scenarios, including water deficiency, water sufficiency, nutrient sufficiency, and nutrient deficiency. Deadlocks detected during initial modeling were successfully resolved by introducing an additional synchronization transition (T15), ensuring uninterrupted system operation. The Petri Net models were implemented and simulated using PIPE version 4.3.0 and WoPeD version 3.8.0, demonstrating both formal verification and workflow visualization. Overall, the study confirms that Petri Net-based modeling is an effective approach for designing, verifying, and optimizing automated irrigation systems, providing a robust framework for future enhancements and real-world deployment.

6. Future Work

Future research can extend the proposed system in several directions to enhance functionality, scalability, and real-time responsiveness:

Colored Petri Nets (CPNs): Incorporating CPNs will allow tokens to carry data values such as soil moisture, nutrient concentration, or weather conditions. This en- ables modeling of variable irrigation volumes and adaptive fertilization schedules, improving the precision of automated operations.

Multi-crop and Multi-field Systems: The model can be scaled to support multiple crop types with different water and nutrient requirements, as well as multiple fields or zones. CPNs can help coordinate parallel operations across these zones while preserving deadlock-freeness and liveness.

Integration with IoT and Cloud Platforms: Real-time sensor networks, edge computing, and cloud-based analytics can be integrated to allow dynamic decision-making and predictive irrigation strategies. This would enable automated adjustments based on changing environmental conditions, historical data trends, or weather forecasts.

Performance and Reliability Analysis: Using simu- lation and formal methods, future work can include quantitative analysis of throughput, latency, and resource utilization. For example, evaluating the time taken from sensing to actuation under various load conditions, ensur- ing the system meets real-time requirements.

Machine Learning Hybrid Models: Combining Petri Net verification with machine learning can allow predictive irrigation decisions while preserving formal guarantees. ML models can recommend optimal irriga- tion/fertilization schedules, while the Petri Net ensures the system remains deadlock-free and safe.

Energy and Resource Optimization: Future extensions can consider constraints such as limited water supply, energy-efficient pump usage, or fertilizer consumption, integrating cost-aware or sustainability metrics into the Petri Net model. By pursuing these enhancements, the proposed framework can evolve into a comprehensive, reliable, and scalable plat- form for next-generation precision agriculture, supporting autonomous, adaptive, and resource-efficient irrigation manage- ment.

7. References

[1] A. A., "Research on automatic irrigation control: State of the art and recent results," Agricultural Water Management, vol. 114, pp. 59–66, 2025.

[2] A. B., "Towards an intelligent and automatic irrigation system based on Internet of Things with authentication feature in VANET," Journal of Information Security and Applications, vol. 88, p. 103927, 2025.

[3] A. C., "Smart irrigation systems in agriculture: An overview," Computers and Electronics in Agriculture, vol. 239, Part B, p. 111008, 2025.

[4] J. Alvarez and R. Singh, "Smart agriculture: IoT and machine learning for irrigation management," IEEE Access, vol. 9, pp. 12345–12360, 2021.

[5] A. Patel and S. Mehta, "A comprehensive review of automated irrigation systems in precision agriculture," Computers and Electronics in Agriculture, vol. 195, p. 107463, 2023.

[6] A. D., "Modeling and simulation of an irrigation control system using production flow schema/object-oriented Petri nets," in Proc. INDUSCON, 2025.

[7] A. E., "A prototype modeling of smart irrigation system using Event-B," SN Computer Science, 2025.

[8] T. Murata, "Petri nets: Properties, analysis, and applications," Proceedings of the IEEE, vol. 77, no. 4, pp. 541–580, 1989.

[9] J. Sifakis, "Formal methods: Achievements and challenges," Science of Computer Programming, vol. 55, pp. 11–19, 2005.

[10] A. Cau and F. Dross, "Formal verification methods for cyber-physical systems: A survey," in IEEE Annual Conference on Industrial Electronics, 2018, pp. 1123–1130.

[11] A. Fortas et al., "High-level Petri nets based approach for designing and analyzing IoT systems," Computers & Electrical Engineering, 2025.

[12] W. M. P. van der Aalst, "The application of Petri nets to workflow management," in IEEE International Conference on Systems, Man and Cybernetics, 1998, pp. 1227–1234.

[13] L. Gao, Y. Zhao, and R. Wang, "Adaptive IoT-based irrigation control for water-efficient farming," IEEE Internet of Things Journal, vol. 7, pp. 4512–4524, 2020.

[14] H. Li, Y. Chen, and X. Zhang, "IoT-adaptive irrigation management with multi-sensor fusion," IEEE Sensors Journal, vol. 22, pp. 15678–15688, 2022.

[15] B. G. Yazgac et al., "Petri nets based procedure of hardware/software codesign of an urban agriculture monitoring system," Agro-Geoinformatics, 2019.

[16] C.-Y. Yang et al., "A novel IoT-enabled system for real-time monitoring home appliances using Petri nets," Journal of Canadian Electrical and Computer Engineering, 2025.

[17] C.-Y. Yang et al., "Petri net modeling and analysis of an IoT-enabled system for real-time monitoring of eggplants," Systems Engineering, 2024.

[18] J. López, A. Santana-Alonso, and M. Díaz-Cacho Medina, "Formal verification for task description languages: A Petri net approach," Sensors, vol. 19, no. 22, p. 4965, 2019.

[19] R. Wiśniewski, M. Wojnakowski, and Z. Li, "Design and verification of Petri-Net-Based cyber-physical systems oriented toward implementation in field-programmable gate arrays—A case study example," Energies, vol. 16, p. 67, 2023.

[20] W. M. P. van der Aalst, "Verification of workflow nets," in Application and Theory of Petri Nets, Lecture Notes in Computer Science, vol. 1248, 1997.

[21] F. Kordon, "Modeling and verifying distributed systems with Petri nets," in IEEE Workshop on Applications of Petri Nets, 2012.

[22] "TAPAAL model checker," Available: https://en.wikipedia.org/wiki/TAPAAL_Model_Checker. Accessed: 2026.

[23] "Romeo model checker," Available: https://en.wikipedia.org/wiki/Romeo_Model_Checker. Accessed: 2026.

Author Information

Vaishnav Krishna P received his B.Tech degree in Artificial Intelligence and Machine Learning from Presidency University, Bangalore, and is currently pursuing his Master's degree at MCUT. His research focuses on artificial intelligence, machine learning, and automation systems, with applications in smart irrigation, predictive analytics, and AI prompt engineering. He has experience in modeling and analyzing workflow-based systems, emphasizing efficiency, reliability, and behavioral correctness, bridging the gap between theoretical research and practical implementations.