LEARNPLC: A JAVAFX-BASED EDUCATIONAL PLATFORM FOR LADDER LOGIC PROGRAMMING WITH REAL-TIME CIRCUIT SIMULATION

Ina PAPADHOPULLI and Martin ÇARO
Department of Computer Engineering, Faculty of Information Technology, Polytechnic University of Tirana, Albania
Author for correspondence: ipapadhopulli@fti.edu.al
IP, 0000-0003-2040-6860; MC, 0009-0003-7513-6797

ABSTRACT

Efficient education in programmable logic controller (PLC) programming requires tools that integrate theoretical knowledge with practical applications. This paper presents **LearnPLC**, a comprehensive educational platform developed in JavaFX designed to provide an intuitive environment for ladder logic programming and real-time circuit simulation. The platform employs a directed graph model for circuit representation and integrates real-time signal propagation algorithms to replicate the behaviour of industrial PLC. It supports standard PLC components, including contacts, coils, timers (TON/TOF), and counters (CTU/CTD), while delivering immediate visual feedback during the simulation. By combining technical precision with a user-friendly interface, LearnPLC addresses the common challenges in PLC education and enhances hands-on learning. Results proved the platform's effectiveness in teaching automation concepts through realistic and interactive circuit behaviour.

1. INTRODUCTION

Programmable Logic Controller (PLC) programming education is crucial for preparing skilled automation engineers and addressing the growing skills gap in industrial automation. However, many educational institutions face challenges in providing effective PLC training due to the limited availability of accessible, modern tools that deliver realistic learning experiences without the complexity typically associated with industrial-grade solutions (https://www.siemens.com/tia-portal).

To develop practical skills, students need hands-on experience in ladder logic programming, circuit construction and real-time simulation. However, educational environments demand specialized tools that prioritize learning effectiveness, immediate feedback, and progressive skill

development rather than extensive feature sets designed for industrial production environments (Petruzella, 2016).

LearnPLC addresses these challenges through two key innovations:

1. **Directed- graph-based circuit representation:** Ladder logic circuits are modelled as graphs, with nodes representing the PLC elements and edges representing the electrical connections. This approach enables efficient simulation algorithms while maintaining consistency with the established electrical circuit principles.
2. **Real-time simulation engine:** The platform incorporates a simulation engine that provides immediate visual feedback, including the visualization of signal propagation. This feature helps students understand the dynamic behaviour of PLC circuits without the need for physical circuit hardware.

The platform supports standard PLC elements, such as contacts, coils, timers, and counters, enabling an accurate representation of fundamental automation concepts. The main educational objectives addressed include circuit construction, real-time simulation, and visual state monitoring, which are critical components of effective automation learning.

To evaluate the platform's effectiveness, we analyse it with respect to three key factors:

- **Functional completeness** relative to educational requirements
- **User experience and accessibility** for educational contexts
- **Implementation architecture and educational design**

The remainder of the paper is structured as follows: Section 2 provides an overview of related work in PLC education, existing simulation platforms, and the theoretical foundations underlying our approach. Section 3 describes the proposed platform architecture and its implementation details. Section 4 presents the technical evaluation and analysis of the study. Section 5 discusses the educational effectiveness and benefits of the proposed method. Finally, Section 6 concludes with the key findings and outlines the directions for future research.

2. RELATED WORK

The literature on PLC education and simulation platforms highlights several approaches to address the challenges of automation education. In this study, we examined existing educational tools, research-oriented solutions, and the theoretical foundations that inform our platform design.

2.1 Educational PLC Simulation Platforms and the Need for a New System

Current PLC programming education faces persistent challenges that existing simulation platforms do not adequately address. A review of the available educational tools revealed recurring limitations, which motivated the development of LearnPLC.

Limitations of Existing Platforms

PLCLogix simplifies programming interfaces but lacks the real-time visual feedback necessary for understanding dynamic circuit behaviour (PLCLogix, https://www.plclogix.com). Factory I/O provides advanced 3D visualization but introduces unnecessary complexity that can overwhelm beginners (Vargas *et al.* 2023). LogixPro and Automation Studio offer sophisticated simulation features but require costly licenses and complex setup procedures to use. Commercial industrial platforms, such as Siemens TIA Portal and Schneider Unity Pro, deliver authentic professional experiences but pose substantial barriers owing to their high costs, demanding installation requirements, and extensive feature sets (Siemens TIA Portal, https://www.siemens.com/tia-portal). Open-source solutions, such as OpenPLC, demonstrate feasibility but lack educationally tailored features and often require significant technical expertise (Alves *et al.* 2014).

Critical Educational Gaps

1. **Accessibility barriers:** High licensing costs and complex installation procedures limit widespread adoption of these technologies.
2. **Limited real-time feedback:** Many platforms do not provide immediate visualization of circuit behaviour.
3. **Inappropriate interface design:** Interfaces are often developed for professionals rather than tailored to students learning needs.
4. **Insufficient progressive learning:** Most platforms present the full range of features at once without supporting gradual skill development.
5. **Lack of educational features:** Industrial completeness is frequently prioritized over pedagogical effectiveness.

Educational Requirements

Effective PLC education requires i) immediate visual feedback to clarify cause-and-effect relationships, ii) progressive complexity management from basic to advanced concepts, iii) safe environments for experimentation, iv) accessible platforms with minimal setup and free availability, and v) a student-focused interface design that prioritizes learning effectiveness. These requirements underscore the need for LearnPLC, a purpose-built educational platform designed to combine technical accuracy and industrial relevance with accessibility, usability, and robust pedagogical support.

2.2 Technical Foundations and Software Architecture

Modern educational software development benefits from established software-engineering practices. The Model-View-Controller (MVC) architectural pattern supports the creation of maintainable and extensible educational platforms (https://docs.oracle.com/javafx/). JavaFX provides a robust foundation for developing rich cross-platform educational applications with sophisticated and responsive user interfaces.

Graph-based algorithms provide efficient foundations for circuit simulations and analysis. The challenge lies in adapting these theoretical foundations into practical educational tools that balance computational efficiency and pedagogical effectiveness (Cormen *et al*. 2022).

Research has further demonstrated the educational value of innovative approaches to PLC instruction. Pavković and Živanović (2011) highlighted the effectiveness of virtual environments for PLC programming education, while academic initiatives, such as OpenPLC (Alves *et al*. 2014), have shown the feasibility of creating comprehensive PLC solutions tailored to educational needs. These projects underscore the importance of open, accessible approaches to automation education while identifying opportunities for the development of specialized educational tools.

2.3 Software Engineering Approaches to Educational Tools

Modern educational software increasingly integrates user-centered design principles with contemporary software engineering. The use of Model–View–Controller (MVC) architectural patterns enable the development of maintainable, extensible, and modular learning platforms (https://docs.oracle.com/javafx/). JavaFX provides a robust foundation for

building rich, cross-platform educational applications with sophisticated and responsive user interfaces (Cormen *et al*. 2022).

Incorporating real-time simulation capabilities into educational platforms requires careful architectural design, with attention to performance considerations, modularity, and scalability. These considerations ensure system responsiveness while maintaining pedagogical effectiveness.

3. PROPOSED EDUCATIONAL PLATFORM ARCHITECTURE

This section introduces the LearnPLC educational platform, outlining its system architecture, circuit representation model, and real-time simulation engine. The platform integrates established software engineering principles with domain-specific requirements for PLC education to create a robust and pedagogically effective learning environment.

3.1 System Architecture Design

LearnPLC employs a modular architecture that separates concerns while maintaining a strong educational focus. The system is organized into several core functional modules, each responsible for a distinct aspect of the platform's operation. This modular design not only enhances maintainability and scalability but also facilitates targeted improvements to individual components without compromising the overall system.

Core Application Modules

User Interface Module

The LearnPLCApp functions as the main application controller, coordinating user interactions and managing the overall application state. The *StartupMenu* provides the initial user experience and project management capabilities, while the *ThemeManager* supports visual customization to accommodate different learning preferences.

Circuit Management Module

At the core of the platform is a directed-graph-based circuit model, implemented through a set of interconnected classes:

- ***CircuitDigraph*** – Implements the directed graph representation of ladder logic circuits.

- ***CircuitNode*** – Abstract base class for all circuit elements with specialized implementations.
- ***CircuitEdge*** – Manages connections between circuit elements and tracks signal states.
- ***ConnectionManager*** – Validates circuit topology and ensures connection integrity.

Element Library Module

Provides a comprehensive set of PLC element implementations:

- ***ElementNode*** – Input elements (contacts) with user-controllable states.
- ***OutputNode, SetOutputNode, ResetOutputNode*** – Various output element types.
- ***TimerNode*** – Implements ON-delay and OFF-delay timing with real-time execution.
- ***CounterNode*** – Provides up/down counting functionality with preset values.
- ***ConnectorNode*** – Junction points for parallel circuit paths.
- ***SourceNode*** – Represents the power source for circuit initiation.

Functional Support Module

Contains specialized components that define core application functionality:

- ***ElementManager*** – Enables drag-and-drop functionality and organizes the component palette.
- ***TableManager*** – Provides real-time monitoring and state display for all circuit elements.
- ***UndoRedoManager*** – Manages system states to enable safe experimentation.
- ***UniqueIDGenerator*** – Ensures proper element identification and referencing.

Project Management Module

Supports educational workflow requirements:

- ***FileManager*** – JSON-based project serialization optimized for sharing and collaboration.
- ***ApplicationSnapshot*** – Manages state persistence for undo/redo functionality and session recovery.

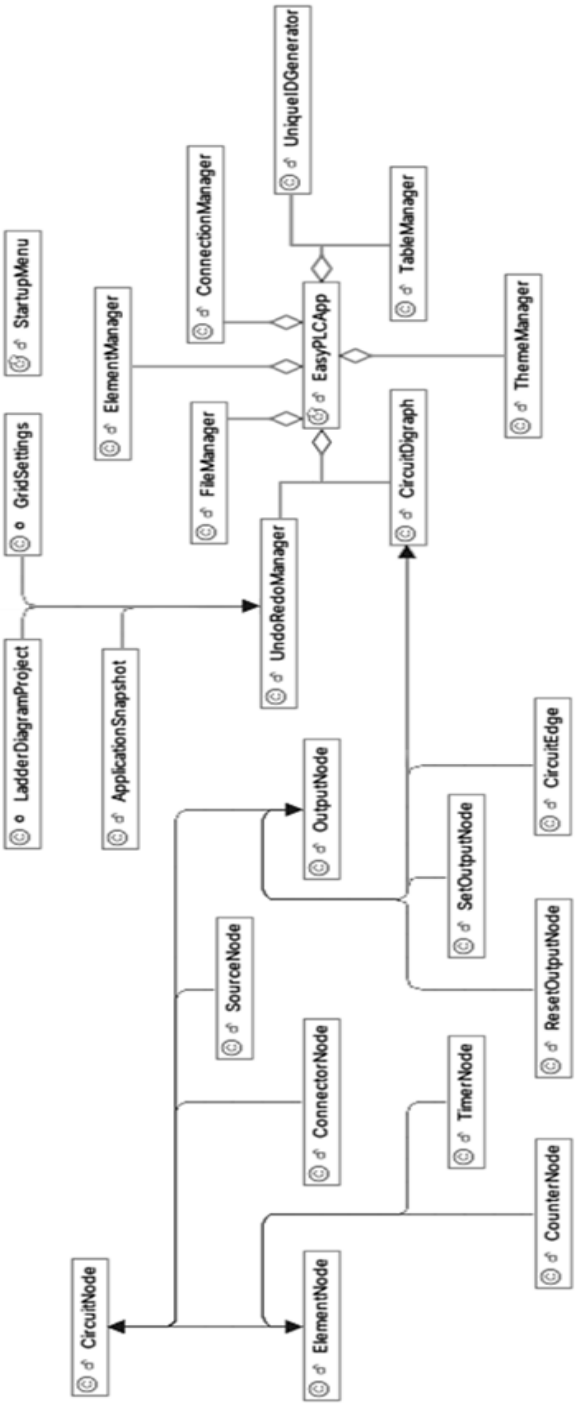

Fig. 1: Modular architecture of LearnPLCApp, illustrating user interface, circuit management, element library, functional support, and project management modules.

3.2 Circuit Representation Model

LearnPLC models ladder logic circuits as directed graphs, where nodes correspond to PLC elements and edges represent electrical connections. This representation supports efficient simulation algorithms, aligns naturally with electrical circuit principles, and enables straightforward validation of circuit integrity.

The ***CircuitNode*** abstract class serves as the foundation for all circuit elements, with specialized subclasses providing distinct different functionalities:

- ***ElementNode*:** Input elements (contacts) with user-controllable states.
- ***OutputNode*:** Output elements (coils) that respond to input signals.
- ***CounterNode*:** Supports up/down counting operations with preset values.
- ***TimerNode*:** Implements ON-delay and OFF-delay timing functions.
- ***ConnectorNode*:** Represents junction points for parallel circuit paths.

The ***CircuitEdge*** class models the electrical connections between elements, maintaining signal state and visual properties to ensure efficient traversal while preserving electrical characteristics.

3.3 Real-Time Simulation Engine

The simulation engine provides a sophisticated yet educationally accessible approach to modeling circuit behavior. It employs a modified breadth-first search (BFS) algorithm for signal propagation, ensuring performance consistent with industrial PLC operations while maintaining real-time responsiveness suitable for educational environments.

Core Simulation Architecture

The engine operates in discrete cycles that mirror industrial PLC scan processes. Each simulation cycle follows a structured sequence:

1. **Initialization Phase:** Resets visual indicators and prepares the circuit for execution.
2. **Reset Signal Processing:** Handles reset commands with assigned priority.
3. **Element State Evaluation:** Updates the internal states of timers and counters.

4. **Signal Propagation:** Traverses the circuit graph according to logical rules.
5. **Visual Update Phase:** Updates all visual indicators in real-time.

Advanced Simulation Capabilities

The engine supports a range of advanced behaviors, including:

- Priority handling for reset signals.
- Edge detection for counter inputs.
- Timer threading for background execution.
- Efficient signal propagation with logical evaluation.
- Real-time visual feedback updates for immediate comprehension.

Educational Benefits

This simulation approach enables students to observe authentic PLC behaviour without the need for physical hardware. Immediate visual feedback reinforces cause-and-effect understanding, while error visualization accelerates learning. Furthermore, adaptive complexity management ensures that the interface remains intuitive as students' progress toward mastering advanced timing and control interactions.

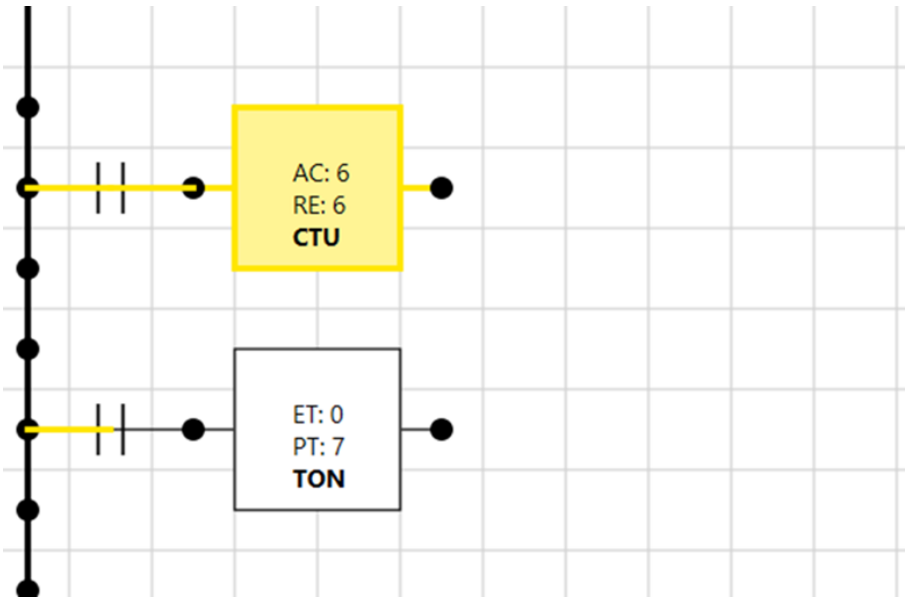

Fig. 2: Example of a simulation in action.

3.4 Advanced Educational Features

LearnPLC integrates several advanced features aimed at enhancing the educational experience:

• **Immediate Visual Feedback:** Circuit connections dynamically change color in real time to indicate signal flow, enabling students to visualize circuit behavior. Active elements are highlighted to provide a clear understanding of system operation.

• **Error Prevention:** The platform automatically validates circuit connections and element configurations to prevent common beginner errors. Invalid connections are rejected, and informative messages guide students toward correct implementation.

• **Progressive Complexity:** The component palette is organized by functional complexity, allowing instructors to introduce concepts progressively. Basic contacts and coils offer introductory-level functionality, while timers and counters support the development of more advanced automation scenarios.

• **State Monitoring:** A comprehensive monitoring panel displays the current state of all circuit elements, including input values, output states, timer progress, and counter accumulation. This real-time feedback helps students correlate programmed logic with observable system behavior.

4. IMPLEMENTATION AND TECHNICAL DETAILS

The LearnPLC educational platform was implemented and evaluated to demonstrate both its technical functionality and pedagogical effectiveness. The system was developed in Java 11 using JavaFX 17, ensuring full cross-platform compatibility across Windows, macOS, and Linux educational environments. Following established software engineering principles, a modular architecture was adopted to support current educational use while enabling future scalability and extensibility.

4.1 Development Environment and Implementation Framework

The technology stack was carefully selected to meet educational objectives while ensuring long-term maintainability and scalability. **Java 11** provides robust, cross-platform compatibility essential for heterogeneous educational computing environments, whereas **JavaFX 17** enables the development of modern, responsive user interfaces with

advanced visual effects and real-time updates. **JSON serialization** is employed to generate human-readable project files, facilitating file sharing and collaboration among students and instructors. **Multithreading** is implemented to handle background processes such as timers and real-time simulation, maintaining interface responsiveness even under complex circuit operations.

The modular design supports the independent development, testing, and maintenance of components while preserving the educational focus. Each user interface component and logical element is encapsulated as a separate module, allowing for straightforward debugging and future feature expansion. Educational design principles guide the implementation, prioritizing user experience and instructional effectiveness over raw computational performance.

Additional features designed specifically for educational environments include:

- **Classroom Mode**: Simplified interfaces optimized for live demonstrations and group activities.
- **Accessibility Support**: Compatibility with screen readers and full keyboard navigation to ensure inclusive learning.
- **Multilingual Framework**: Built-in support for internationalization to accommodate diverse educational contexts.
- **Network Integration**: Potential for linkage with learning management systems and collaborative educational platforms.

4.2 Circuit Representation and Algorithm Implementation

The core technical innovation of **LearnPLC** lies in its representation of ladder logic circuits as **directed graphs**, where nodes correspond to PLC elements and edges represent electrical connections. This design enables efficient simulation algorithms while preserving the integrity of fundamental electrical circuit principles.

The **CircuitDigraph** class functions as the central component, implementing graph traversal algorithms to manage signal propagation. The simulation engine utilizes a **modified breadth-first search (BFS)** algorithm that processes signals in discrete cycles, accurately reproducing the **scan cycle behavior** of industrial PLCs.

The signal propagation process follows a structured sequence:

1. **Initialization:** Each simulation cycle begins with the reset of edge colors and the prioritized handling of reset signals.

2. **Element Processing:** Timers and counters are updated according to the current input conditions.
3. **Signal Propagation:** A modified BFS algorithm propagates signals through the circuit, taking into account the electrical properties of each element type.
4. **Visual Updates:** Connection states and element indicators are refreshed in real-time, providing immediate feedback to the user.

```
public void simulateSignal() {
     // Initialize simulation cycle
     startNewSimulationCycle();
     resetEdgeColors();
    // Process reset signals first
     processResetSignals();

    // Handle timers and counters
     processTimersAndCounters();

    // Propagate signals through the circuit
     Queue<CircuitNode> queue = new LinkedList<>();
     addSourceNodesToQueue(queue);

    while (!queue.isEmpty()) {
         CircuitNode current = queue.poll();
         if (current.processSignal()) {
             propagateToConnectedNodes(current, queue);
         }
     }

    // Update visual feedback
     updateConnectionVisuals();
     updateElementDisplays();
 }
```

The simulation algorithm effectively handles complex scenarios, including priority-based reset signal processing, edge detection for counter inputs, background timer threading, and real-time visual updates, thereby ensuring an accurate and responsive circuit simulation environment ().

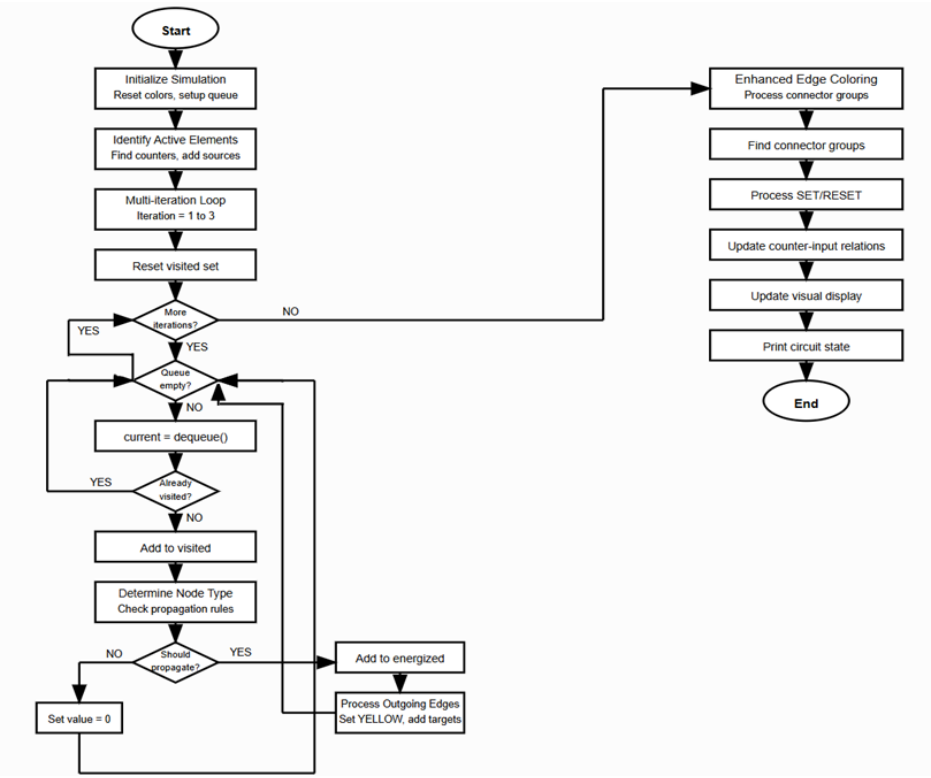

Fig. 3: Control flow of the signal simulation algorithm in LearnPLC, illustrating signal propagation, timer/counter handling, and the real-time update cycle.

4.2.1 Directed Graph Advantages in Circuit Representation

The directed graph representation offers several key advantages that make it particularly effective for educational PLC simulation:

- **Natural Circuit Modeling:** The graph structure maps directly onto ladder logic topology, where nodes represent PLC elements (contacts, coils, timers) and directed edges represent electrical connections with defined signal flow. This approach establishes an intuitive correspondence between the visual circuit layout and the underlying data model, thereby enhancing readability and conceptual understanding.
- **Efficient Simulation Algorithms:** Graph traversal techniques—particularly the modified breadth-first search (BFS) algorithm—enable systematic signal propagation throughout the circuit. The

directed nature of edges ensures that signals follow correct electrical flow patterns, moving from input elements through intermediate logic to output elements. This behavior closely mirrors that of industrial PLC systems.

- **Source Traceability and Debugging:** The directed graph structure supports efficient backward traversal for troubleshooting and educational analysis. When circuit elements are modified or removed, the system can trace connections back to their source nodes, maintaining complete circuit integrity and clearly illustrating signal dependencies. This bidirectional navigation capability is essential for interactive circuit editing and for providing students with meaningful feedback on signal origins and logical flow.

Educational Benefits: The graph abstraction allows students to focus on logical relationships rather than low-level implementation details. By working with familiar ladder logic symbols, learners develop an intuitive understanding of automation principles, while the platform's underlying graph algorithms automatically handle signal timing, state persistence, and interactions between parallel paths.

Validation Capabilities: Graph-based algorithms enable the automatic detection of common circuit configuration errors, such as unconnected elements, invalid connections, and incomplete circuit paths. This real-time validation provides immediate feedback, allowing students to identify and correct mistakes as they occur. Such instant reinforcement promotes proper circuit construction techniques and significantly enhances learning outcomes.

Extensibility: The modular system design allows new PLC element types to be introduced easily by implementing the *CircuitNode* interface. These newly added elements automatically inherit the platform's graph traversal and simulation capabilities without requiring changes to the core algorithms. This approach supports the gradual integration of advanced automation concepts and ensures the platform's long-term scalability and maintainability.

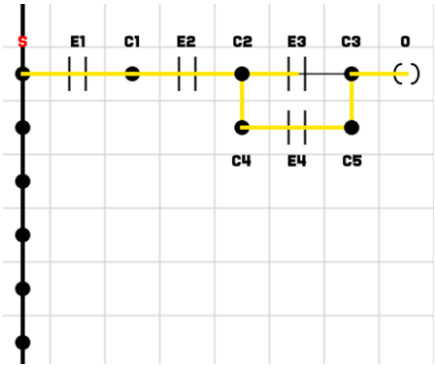

Fig. 4: Circuit example.

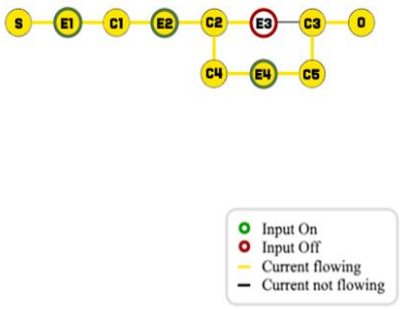

Fig. 5: Digraph implementation of the circuit.

4.3 Educational Feature Implementation and User Interface Design

The LearnPLC user interface is developed in JavaFX and follows established educational usability principles. Its three-panel layout provides a clear and intuitive workspace comprising:

- **Component Palette** for drag-and-drop circuit construction
- **Central Grid Area** for circuit development
- **Real-Time Monitoring Panel** for observing system behavior

Component Palette Design: The component palette enables intuitive circuit construction without requiring programming syntax. PLC elements are organized by functional complexity to support progressive learning. Grid-based positioning ensures orderly layouts, helping students maintain visually structured circuits as their projects increase in complexity.

Visual Feedback Systems: Circuit connections employ dynamic color-coding to illustrate signal flow. Active connections appear in bright yellow, while inactive connections remain black, giving students an immediate and intuitive understanding of signal propagation patterns. This visual feedback reinforces conceptual learning by making cause-and-effect relationships within the circuit explicit and easy to follow.

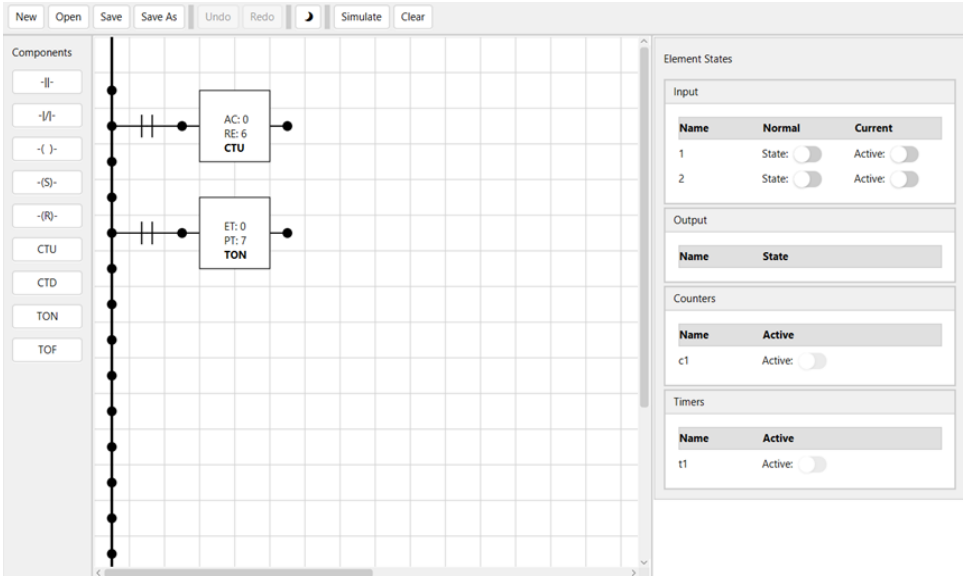

Fig. 6: User interface of the application.

4.4 Element Library and Specialized Component Implementation

LearnPLC supports a comprehensive library of standard PLC elements, each designed to balance authentic industrial behavior with educational accessibility.

Basic Elements

- **Input Contacts (NO/NC):** User-controllable circuit inputs that provide immediate visual feedback and reflect authentic logic behavior.
- **Output Coils:** Respond to input signals through name-based linking mechanisms, with clear visual indicators to reinforce understanding of output states.

Advanced Components

- **Timer Nodes (TON/TOF):** Provide ON-delay and OFF-delay functionality with configurable preset values and real-time elapsed time display.

- **Counter Nodes (CTU/CTD):** Support both count-up and count-down operations with automatic edge detection and real-time count display.
- **Set/Reset Output Nodes:** Offer latching functionality to demonstrate memory concepts in automation systems.
- **Connector Nodes:** Enable the creation of parallel circuit paths and junction points to model complex logic flows.

Educational Benefits

All components adhere to industry standards while remaining intuitive for learners. This design supports a natural progression from basic to advanced automation concepts, maintaining a consistent visual style and integrated validation mechanisms that prevent common configuration errors. As a result, students can focus on mastering core concepts without being distracted by technical complexities.

4.5 File Management and State Persistence

LearnPLC provides robust project management features, enabling students to save, share, and resume their work across multiple sessions through JSON-based serialization.

Project Data Model: The **LadderDiagramProject** class encapsulates project metadata, a complete catalog of circuit elements with configurations, a detailed connection map, and grid layout settings. The use of JSON facilitates project debugging, version control integration, and easy collaboration among students and instructors.

State Management: The **UndoRedoManager** class implements snapshot-based state management, allowing students to experiment freely without the risk of losing progress. This approach encourages exploratory learning and supports the development of practical problem-solving skills.

Error Handling: The system includes comprehensive error handling tailored to educational environments. File format validation, graceful recovery from element recreation errors, and clear, instructive error messages help students identify and correct mistakes without frustration.

Educational Impact: These implementation features demonstrate how purpose-built educational tools can offer substantial advantages over adapted industrial software. By prioritizing accessibility, ease of use, and pedagogical objectives, LearnPLC enables students to focus on learning while still experiencing an authentic PLC programming environment.

5. EDUCATIONAL VALIDATION AND EFFECTIVENESS

The platform's design integrates established educational principles and software engineering best practices to create an effective learning environment. Its implementation demonstrates how purpose-built educational tools can enhance in accessibility, usability, and learning support through thoughtful, research-informed design decisions.

5.1 Learning Progression Support and Cognitive Benefits

LearnPLC supports a natural learning progression through carefully managed complexity, aligning with key principles of educational psychology. Students begin with basic contacts and coils before progressing to timers and counters, following cognitive load theory and scaffolded learning strategies. Each new concept builds on previously mastered skills, creating a solid foundation for advanced topics.

The platform mitigates cognitive overload through multiple mechanisms:

- **Information Chunking:** Automation concepts are divided into manageable, logically connected units.
- **Visual Hierarchy:** Interface design directs attention to relevant information at each stage of learning.
- **Context-Sensitive Help:** Assistance appears only when needed, reducing interface clutter.
- **Reduced Extraneous Load:** Students focus on mastering automation concepts rather than struggling with complex software interfaces.

5.2 Real-Time Learning Reinforcement and Feedback Systems

Immediate Learning Validation:

LearnPLC delivers multiple layers of immediate feedback to enhance learning effectiveness:

- **Instant Visual Confirmation:** Color-coded connections and element states reflect the outcomes of programming decisions.
- **Logical Relationship Visualization:** Real-time updates illustrate cause-and-effect relationships within automation systems.

- **Error Detection and Correction:** Immediate identification of circuit issues supports rapid learning from mistakes.
- **Positive Reinforcement:** Clear visual confirmation of correctly functioning circuits strengthens good programming habits.

Experiential Learning Support

The platform facilitates hands-on exploration that builds practical skills:

- **Safe Experimentation:** Students can test ideas without risk to equipment or data.
- **Rapid Iteration:** Quick modification and testing encourage experimental learning.
- **Hypothesis Testing:** Students validate their understanding by building circuits and observing outcomes.
- **Problem-Solving Development:** Real-time feedback promotes systematic troubleshooting and analysis.

Metacognitive Skill Development

Monitoring and feedback mechanisms cultivate higher-order thinking skills:

- **Self-Assessment:** Students evaluate their understanding through circuit behavior.
- **Strategy Development:** Multiple solution paths encourage robust programming approaches.
- **Transfer Learning:** Skills gained with simple circuits transfer naturally to complex automation challenges.
- **Reflection:** Built-in feedback prompts students to analyze and understand circuits behavior.

5.3 Practical Application and Skill Transfer

Authentic Learning Contexts

LearnPLC immerses students in realistic automation scenarios using genuine PLC programming concepts:

- **Real-World Problem Solving:** Students design, test, and debug ladder logic circuits that reflect industrial challenges.
- **Industrial Relevance:** All functionalities and behaviors mirror those of real PLC systems.

- **Scalable Complexity:** Students progress from basic on/off control to advanced timing and counting applications.
- **Professional Practice:** The development environment parallels professional automation programming workflows.

Skill Development Outcomes

Students develop a comprehensive set of automation competencies:

- **Technical Skills:** Accurate ladder logic construction, element configuration, and circuit debugging.
- **Conceptual Understanding:** Deep comprehension of automation principles, timing relationships, and logic dependencies.
- **Problem-Solving Abilities:** Systematic approaches to troubleshooting and design.
- **Professional Readiness:** Confidence and competence transferable directly to industrial settings.

Project-Based Learning Benefits

The platform's project management system promotes extended and collaborative learning:

- **Iterative Development:** Students build complex automation solutions over multiple sessions, reflecting authentic workflows.
- **Portfolio Building:** Completed projects document learning progression for assessment and career development.
- **Collaborative Learning:** Project sharing enables peer learning and instructor feedback throughout the process.
- **Real-World Application:** Students experience the complete automation project lifecycle—from concept to testing.

6. CONCLUSIONS

This paper introduced LearnPLC, an educational platform for programmable logic controller (PLC) programming. The system employs a directed graph-based model to represent control logic, enabling efficient and accurate simulation of circuit behavior consistent with established electrical principles. The JavaFX-based user interface provides a modular and responsive environment that supports interactive learning and structured exploration of PLC concepts through visual programming.

Key contributions of this work include:

Novel Circuit Representation: A directed graph model that supports efficient simulation algorithms while preserving electrical circuit semantics essential for PLC education.

Education-Focused Design: A purpose-built interface prioritizing learning effectiveness over feature completeness, incorporating immediate visual feedback and built-in error prevention.

Modern Software Architecture: A modular and extensible design that facilitates future enhancements and customization across diverse educational contexts.

Accessibility: Free, cross-platform availability that removes common barriers to PLC education.

By balancing functionality with usability, LearnPLC demonstrates the value of purpose-built educational tools. Its emphasis on fundamental concepts and immediate feedback enables students to build a strong foundation in automation programming before transitioning to industrial platforms.

6.1 Future Research Directions

Several opportunities exist for extending LearnPLC to broaden its educational impact:

- **Enhanced Element Library:** Incorporating function blocks, mathematical operations, and communication modules would expand the range of instructional scenarios while maintaining simplicity.
- **Collaborative Learning:** Multi-user support with real-time collaboration could facilitate peer learning, instructor guidance, and remote teaching.
- **Assessment Integration:** Automated assessment and analytics could provide objective evaluation of student progress and support adaptive instruction.
- **Industrial Integration:** Introducing graduated complexity modes could bridge the gap between educational and industrial platforms, offering smoother transition pathways for learners.
- **Mobile Platforms:** Developing native mobile applications could enhance accessibility and enable ubiquitous learning opportunities.

The implications of this work extend beyond PLC education to the broader challenge of making complex technical concepts accessible to diverse learners. By applying contemporary software engineering practices, LearnPLC demonstrates how educational effectiveness can be

significantly enhanced while reducing barriers to entry. As automation technologies continue to evolve, platforms such as LearnPLC will play an increasingly important role in preparing skilled automation professionals. Its open architecture ensures continued adaptability to emerging educational needs and technological advancement.

Declarations:

Data accessibility:

There are no databases to be made public, but the codes are available upon request to the author.

Declaration of AI use:

There has been no use of AI when writing the actual paper.

Authors' contributions:

Papadhopulli I – Idea generation, study design, methodological development, data evaluation, provision of materials, review and editing, corresponding author; **Çaro M** - Idea generation, study design, experimental analysis, data evaluation, implementation, provision of materials, drafting the original manuscript.

Authors approval:

All authors gave final approval for publication and agreed to be held accountable for the work performed therein.

Conflict of interest declaration:

The author declares there are no conflicts of interest.

Funding:

The author declares no funding was received when writing this paper.

REFERENCES

Alves T, Buratto M, de Souza FM, Virgínia Rodrigues TV. **2014.** OpenPLC: An open source alternative to automation. In *IEEE Global Humanitarian Technology Conference*. DOI: 10.1109/GHTC.2014.6970342.

Cormen TH, Leiserson ChE, Rivest RL, Stein C, *Introduction to Algorithms*, 4th ed. Cambridge, MA: MIT Press, 2022. ISBN: 978-0262046305. DOI: 10.7551/mitpress/13329.001.0001.

Gamma E, Helm R, Johnson R, Vlissides J. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, MA: Addison-Wesley. ISBN: 978-0201633610.

Oracle Corporation. *JavaFX Documentation*. Oracle Help Center, 2023. Available at: https://docs.oracle.com/javafx/ Last accessed 01.09.2025.

Pavković B, Živanović D. 2011. Development of a virtual environment for PLC programming and simulation. In *Proc. 34th International Convention MIPRO*. DOI: 10.1109/MIPRO.2011.5967315.

Petruzella FD. 2016. *Programmable Logic Controllers*, 5th ed. New York: McGraw-Hill Education, 2016. ISBN: 978-0073373843.

PLCLogix LLC. *PLCLogix 5000 Simulation Software*. Official product website, 2022. Available at: https://www.plclogix.com Last accessed 28.08.2025

Siemens AG. *TIA Portal: Integrated Engineering Framework*. Official product website, 2023. Available at: https://www.siemens.com/tia-portal Last accessed 20.07.2025.

Vargas H, Romero D, García F. 2023. Teaching automation with Factory I/O under a competency-based curriculum. *Multimedia Tools and Applications*, **82**, pp. 19221–19246. DOI: 10.1007/s11042-022-14047-9.