ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 6, June 2026

A Unified Visual Topological Modeling Framework for Traction SCADA HMI and Decision Support Systems

[1] Sneha Prajapati, [2] Satwik Anmol

[1] [2] Central Research Laboratory, Bharat Electronics Limited, India
*Corresponding Author Email: [1] snehaprajapati@bel.co.in, [2] satwikanmol@bel.co.in

Abstract*— The engineering approach to SCADA railway traction systems is predominantly manual, prone to errors, and difficult to scale. Engineers must independently design graphical Human-Machine Interface (HMI) layouts and electrical topology definitions for Decision Support Systems (DSS), usually employing large vendor-specific XML files. This division results in discrepancies between visual representations and decision-making processes, heightened commissioning requirements, and diminished capacity for configuration reuse or alteration. This document introduces a comprehensive visual-topological modeling approach that views visual circuit design as the sole authoritative source for both Human-Machine Interface (HMI) and Decision Support System (DSS) configurations. The suggested method allows engineers to design traction substations through a graphical interface, which then automatically creates a topology-aware model and generates XML configurations for HMI displays and DSS connectivity. Furthermore, the framework incorporates a JSON-based intermediate representation, enabling round-trip engineering. This allows the generated configurations to be reopened, edited, and regenerated without loss of structural or semantic information. A hierarchical relational perspective facilitates the manipulation of component and connection elements without spatial constraints. The proposed blueprint eliminates unnecessary manual configuration effort, enforces consistency by design, and significantly enhances the scalability of railway traction SCADA configuration engineering. The effectiveness and practical applicability of the approach are demonstrated through a real-world metro traction SCADA case study.

Index Terms*- Traction SCADA, Railway Electrification, Decision Support System (DSS), Supervisory Control and Data Acquisition (SCADA), Model-Driven Engineering, Visual Circuit Modeling, Graph-Based Topology, HMI Configuration Automation, XML Configuration Generation, JSON Round-Trip Engineering, Interlocking Logic, Traction Substation Modeling, SCADA Configuration Management.

I. INTRODUCTION

Supervisory Control and Data Acquisition (SCADA) systems play a central role in the monitoring, control, and operational supervision of large-scale industrial infrastructures. In a typical SCADA deployment, field devices and Remote Terminal Units (RTUs) acquire process data, while supervisory and analytical components perform system-level monitoring, control, and decision-making. Operator interaction is facilitated through a Human–Machine Interface (HMI), which provides situational awareness and control functionality. Within this layered architecture, the Decision Support System (DSS) serves as the analytical backbone, interpreting system state information and assisting operators in making timely and safe operational decisions [1], [2].

In railway traction power systems, the importance of DSS functionality is amplified due to the distributed and safety-critical nature of the electrical network. Traction SCADA systems oversee geographically dispersed power supply infrastructure that feeds moving trains under strict operational and safety constraints. These systems typically span multiple substations-including Traction Substations (TSS), Receiving Substations (RSS), and Auxiliary Substations (ASS)-each comprising numerous interrelated components such as transformers, circuit breakers, isolators, earthing switches, and interconnecting conductors. Operational actions related to fault isolation, sectionalization, and power restoration must there-fore be executed with high accuracy and minimal delay, as incorrect decisions can directly impact train operations, service availability, and passenger safety.

Despite the critical role of DSS in traction SCADA systems, configuration engineering practices remain largely manual and highly project-specific [3], [4]. In prevailing workflows, engineers independently develop two closely related yet separately managed configuration artifacts. One artifact defines the HMI configuration, describing graphical layouts, equipment symbols, spatial placement, and measurement displays, typically encoded using vendor-specific XML formats [3], [4], [14]. The second artifact specifies the DSS configuration, capturing the electrical and operational topology of the traction network, including connectivity relationships, flow directionality, sectionalization boundaries, traversal sources, and interlocking constraints required for decision-support and protection co-ordination functions.

Although HMI and DSS configurations represent the same physical system, they are commonly engineered using

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 6, June 2026

different tools, schemas, and design assumptions. As a con-sequence, consistency between operator visualization and decision-support logic is not enforced by design and must instead be maintained through manual coordination. In large metro deployments involving dozens of stations per line, configuration artifacts can comprise several thousand lines of XML, with substantial redundancy and strong dependence on fixed screen-level geometry. Even minor changes to layout or topology often necessitate extensive manual updates across multiple files, increasing engineering effort and the risk of configuration inconsistencies.

Existing standards, such as the Common Information Model (CIM), focus primarily on semantic interoperability and data design or operator-oriented configuration workflows. Similarly, Graph-based models developed for power system analysis are typically applied in isolation and are not tightly integrated into practical SCADA configuration engineering processes [6], [7]. As a result, current approaches do not provide a unified engineering representation that simultaneously addresses visualization, topology reasoning, and long-term maintainability.

To overcome these limitations, this paper presents a unified visual–topological modeling framework in which visual circuit design serves as the authoritative source for both HMI and DSS configuration. By combining visual modeling, topology-aware abstraction, and round-trip configuration engineering within a single environment, the proposed frame-work enforces consistency between operator visualization and decision-support logic while reducing manual effort, configuration redundancy, and error risk in safety-critical traction SCADA deployments.

II. RELATED WORK

The configuration of SCADA systems has traditionally been approached as an engineering task tightly coupled to vendor-specific tools and proprietary schemas. CIM-based standards defined under IEC 61970 and IEC 61968 provide a common semantic basis for power-system data exchange; however, they do not address graphical HMI layout geometry or configuration engineering workflows [3], [4].

Graph-based representations of electrical networks are widely employed for power-system analysis, fault detection, and network traversal [6], [7], [9]. While these models emphasize analytical accuracy and computational efficiency, they are generally developed independently of SCADA HMI configuration processes and are not directly integrated into engineering workflows.

Railway-specific SCADA research primarily concentrates on traction power reliability, energy optimization, and fault diagnostics [1], [2], [8]. In this body of work, configuration engineering is typically treated as a manual prerequisite rather than a subject of systematic automation or methodological innovation.

Model-Driven Engineering (MDE) has been successfully applied in industrial automation to reduce manual effort and improve consistency across complex systems [10], [11], [13]. However, its application to railway traction SCADA- particularly with support for round-trip engineering and unified modeling of HMI and DSS configurations- remains limited.

The proposed framework builds on these foundations by explicitly addressing the gap between visual configuration, topological correctness, and long-term maintainability within a unified engineering environment.

Table I summarizes the key differences between conventional SCADA configuration approaches and the proposed unified framework.

III. DESIGN OBJECTIVES AND REQUIREMENTS

The design of the proposed framework is motivated by the need to eliminate manual, repetitive, and error-prone configuration workflows in railway traction SCADA systems, while maintaining strict consistency between system visualization and decision-support logic [10], [14]. Achieving this objective requires an engineering approach that supports intuitive interaction, scalable system representation, and automated generation of configuration artifacts without compromising flexibility or extensibility. Accordingly, the framework is guided by a set of functional and non-functional requirements, which are outlined in the following subsections.

A. Functional Requirements

From a functional standpoint, the proposed framework provides a visual modeling environment that enables engineers to construct traction electrical circuits in a manner aligned with established engineering practice. Rather than interacting with low-level, vendor-specific configuration files, system structure is specified through graphical composition. The framework internally manages identifier assignment, connectivity interpretation, and configuration synthesis, thereby abstracting implementation details from the engineering workflow. It also supports the complete configuration lifecycle, including modification, regeneration, and reuse of existing system designs.

The framework facilitates visual construction of traction electrical networks using drag-and-drop components and structured interconnections, while supporting multiple traction sub-station types such as Receiving Substations (RSS), Traction Substations (TSS), and Auxiliary Substations (ASS). From a single authoritative system model, all required configuration artifacts are generated automatically, including HMI XML for operator visualization, DSS topology XML for decision-support processing, and a JSON-based intermediate representation preserving both visual layout and topological structure. For decision-support applications, the framework provides explicit topology modeling capabilities. Electrical

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 6, June 2026

connectivity defined during visual modeling is transformed into a graph-based representation that enables inference of flow direction, identification of source and sink elements, and modeling of dependency relationships required for traversal, fault isolation, and service restoration analysis. These topology semantics are derived directly from the unified model and are consistently reflected in the generated DSS configuration.

Operational safety requirements are addressed through integrated modeling of interlocking logic. Both hard and soft interlocking constraints are represented as first-class elements within the system model, allowing safety-critical dependencies to be specified alongside electrical topology and incorporated automatically during DSS configuration generation.

To support efficient engineering of large and complex traction networks, configurations can be reopened, edited, and regenerated without loss of structural or semantic information through lossless round-trip engineering. In addition to spatial circuit modeling, hierarchical and relational views of system components and connections are provided to support structure-oriented editing when graphical interaction becomes inefficient.

Table I. Comparison Between Conventional SCADA Configuration Practices and the Proposed Unified Framework

Aspect	Conventional SCADA Configuration	Proposed Unified Framework
Engineering paradigm	Manual, tool-dependent engineering workflows closely tied to vendor-specific environments [1], [14], [16]	Model-driven engineering approach based on tool-independent abstraction and a unified system model [10], [11], [13]
Primary configuration representation	Vendor-specific XML artifacts authored and maintained directly by engineers [1], [5]	Unified abstract system model supported by an editable JSON-based intermediate representation [10], [18]
HMI configuration process	Graphical layouts manually specified using fixed screen coordinates and layout-dependent XML [12], [15]	HMI configurations generated automatically from visual circuit models, ensuring repeatable layout behavior [11], [13]
DSS configuration process	Topology and decision-support logic authored independently of HMI configuration [6], [7], [9]	DSS configuration derived automatically from the same authoritative model used for HMI generation [10], [11]
HMI–DSS alignment	Achieved through manual coordination and review activities [1], [14]	Guaranteed by construction through unified visual–topological modeling [10], [18]
Support for configuration round-trip	Limited; generated XML artifacts are difficult to reopen, modify, or reuse [14], [16]	Fully supported through lossless JSON-based round-trip engineering [11], [13]
Topology representation	Implicit and distributed across multiple configuration artifacts [6], [9]	Explicit graph-based topology model derived from the abstract traction model [6], [7]
Handling of system changes	Requires coordinated manual updates across multiple files and tools [1], [14]	Single-point modification at the model level with automated regeneration of all outputs [10], [18]
Scalability	Poor scalability for large, multi-station traction SCADA deployments [2], [8]	Designed to support large-scale metro and multi-line traction SCADA systems [10], [13]
Configuration error exposure	High risk due to manual duplication, schema complexity, and coordinate-level editing [14], [16]	Substantially reduced through automation, model validation, and single-source configuration [10], [11]
Maintainability	Degrades over time as configuration complexity increases [1], [18]	Sustained through structured models that support long-term evolution [10], [13]

B. Non-Functional Requirements

In addition to functional capabilities, the proposed frame-work must satisfy several non-functional requirements that are critical for large-scale traction SCADA deployments. Given the safety-critical nature of railway systems, configuration generation must be deterministic and reproducible, such that identical system models always yield identical configuration outputs [10], [11]. This determinism is essential to support reliable validation, commissioning, troubleshooting, and long-term maintenance across the system lifecycle.

The framework must ensure strict consistency between HMI visualization and DSS logic by deriving both

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 6, June 2026

configurations from a single authoritative system model. By eliminating independently engineered and manually synchronized arti-facts, the risk of inconsistencies between operator displays and decision-support behavior is significantly reduced. Deterministic generation further enables systematic verification of configuration changes, allowing revisions to be introduced confidently without compromising operational correctness.

Scalability is another key non-functional requirement, as traction SCADA systems often span large metro networks with multiple stations, substations, and traction levels. The framework must therefore support efficient modeling and configuration generation for large-scale deployments, while remaining adaptable to evolving project requirements, equipment variations, and vendor-specific schema extensions.

Finally, the framework emphasizes maintainability and error reduction by minimizing direct human interaction with low-level configuration files. Through model-driven abstraction and automated configuration generation, manual duplication and schema-level editing are avoided, thereby reducing the likelihood of configuration errors and rework over the project lifecycle.

IV. UNIFIED SYSTEM ARCHITECTURE

The proposed framework adopts a model-driven, layered architecture that explicitly separates user interaction, system abstraction, topology processing, and configuration generation. This separation ensures that visual design decisions, electrical semantics, and output representations remain decoupled, while still being derived from a single authoritative system model [10], [13].

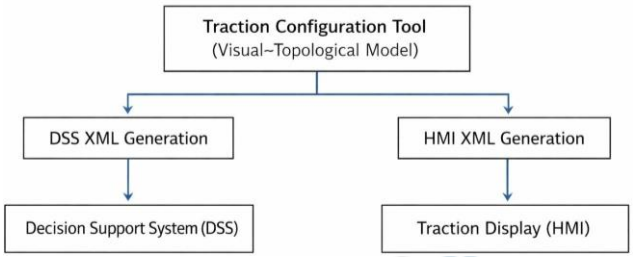

Figure 1. Parallel generation of HMI XML and DSS XML from the unified visual–topological traction configuration model, enabling consistent operator visualization and topology-driven decision-support processing from a single authoritative system representation

Such an architectural organisation enhances maintainability, facilitates extensibility, and supports deterministic configuration generation for large-scale traction SCADA deployments.

The framework is structured into five tightly integrated layers, each addressing a distinct responsibility within the overall engineering workflow.

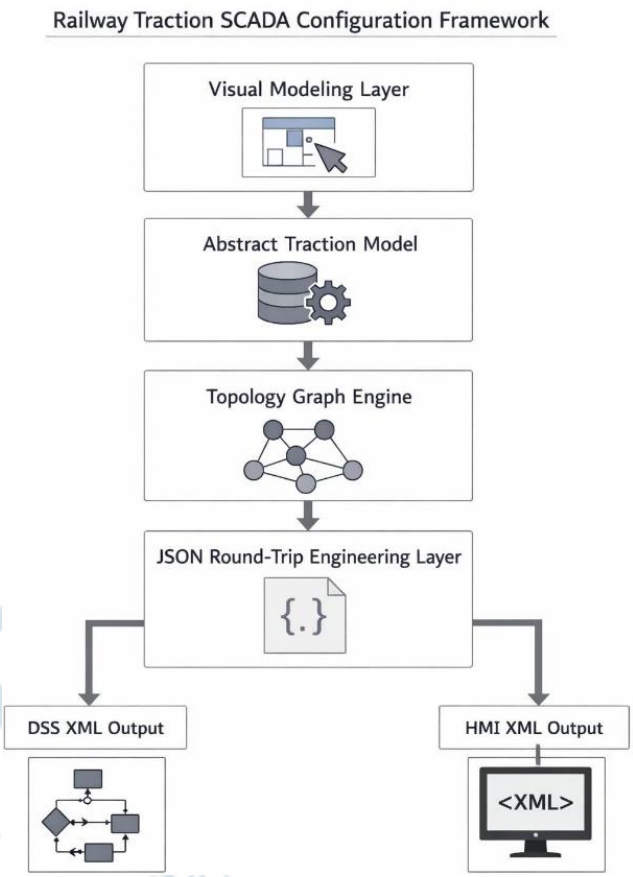

Figure 2. Unified visual–topological configuration framework for railway traction SCADA

A. Visual Modeling Layer

The Visual Modeling Layer provides the primary interface through which engineers design traction electrical circuits. The framework offers a drag-and-drop visual editor that allows users to place electrical components and define interconnections in a manner consistent with conventional single-line traction diagrams [17]. Electrical components are represented as parameterized symbols, while interconnections are modeled as structured line segments composed of ordered points, enabling precise representation of routing and branching structures [12], [15].

Within this layer, engineers can visually position traction equipment, draw and modify electrical connections, and define branching relationships through direct graphical interaction. The editor enforces basic electrical and connectivity constraints to prevent invalid or inconsistent configurations, such as improper terminations or unsupported connections. At the same time, the visual modeling environment remains fully decoupled from low-level XML schemas and vendor-specific configuration formats. This abstraction allows engineers to focus on electrical intent and system design rather than configuration syntax, while ensuring that all necessary information is captured for subsequent abstraction and automated configuration generation.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 6, June 2026

B. Abstract Traction Model

All elements created within the visual modeling layer are mapped to an abstract traction model that captures the structural and logical composition of the system independently of its graphical representation. This abstract model formalizes the relationships between stations, substations, traction levels, and electrical equipment, while preserving all identifiers, attributes, and sequencing information required for downstream configuration generation. By explicitly decoupling logical system structure from visual layout, the abstract traction model serves as the single authoritative source of truth for all subsequent processing stages within the framework.

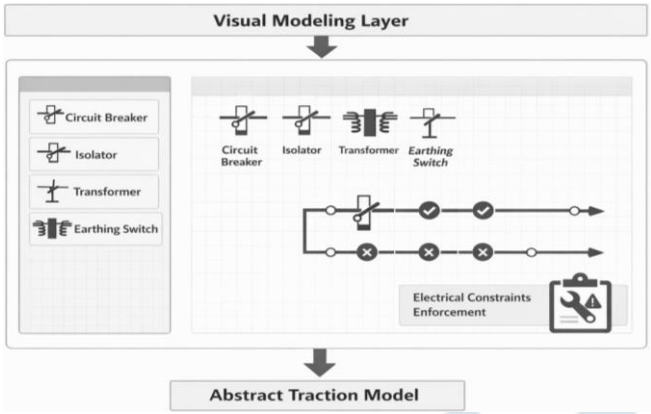

Figure 3. Visual modeling environment for traction SCADA configuration showing component symbols, connectivity, and constraint enforcement

The abstract traction model represents the complete hierarchical organization of the traction system, including stations, substations, and traction levels, as well as individual equipment objects characterized by unique identifiers, equipment types, and configuration attributes. In addition, it captures logical connectivity and sequencing relationships between components, enabling consistent interpretation of electrical structure across visualization, topology analysis, and configuration generation workflows.

Beyond structural and connectivity information, the abstract traction model also encodes operational constraints in the form of interlocking rules. These rules express dependencies and mutual exclusions between electrical equipment and are modeled independently of visual layout considerations. Inter-locking definitions are persisted as part of the unified system model and are subsequently processed by the topology graph engine. Through the JSON round-trip engineering layer, these constraints are serialized and incorporated into the generated DSS configuration, ensuring that operational safety logic remains consistent with the underlying system topology.

C. Topology Graph Engine

The Topology Graph Engine converts the unified abstract traction model into a graph-oriented representation tailored for decision-support analysis. In this representation, traction equipment-including transformers, circuit breakers, and isolators-as well as logical line or bus segments are represented as vertices, while electrical interconnections are captured as edges. This graph abstraction allows structural reasoning over the traction network independently of its graphical layout, while retaining the electrical semantics required for operational evaluation.

To support correct interpretation of electrical behavior, the framework enables engineers to specify high-level power-flow intent at the modeling stage by identifying source and sink elements and defining allowable flow characteristics. These specifications express engineering intent at the model level and serve as constraints for subsequent topology processing, removing the need for manual definition of traversal behavior.

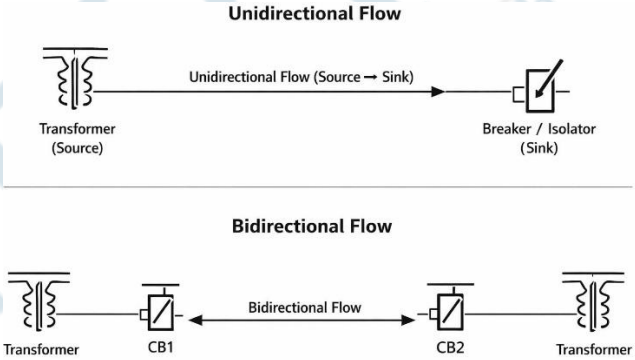

Figure 4. Illustrative topology interpretations derived from visual circuit models under different power-flow configurations

Using the configured intent in combination with predefined inference rules and model attributes, the topology graph engine automatically derives traversal properties such as effective connection direction, identification of traversal entry and exit points, and determination of sectionalization boundaries. For instance, network configurations supplied by a single transformer result in a well-defined downstream traversal pattern, whereas networks containing multiple interconnected transformers allow alternative traversal paths to be inferred depending on operational state and switching conditions.

By clearly separating engineer-defined intent from automatically inferred topology behavior, the resulting graph representation reflects real-world electrical operation without requiring explicit low-level logic specification. The inferred topology and traversal properties are subsequently serialized into DSS topology XML, forming the foundation for decision-support functions including fault traversal, isolation analysis, and power-flow reasoning based on a consistent and authoritative system representation [6], [9].

D. XML Generation Layer

The XML Generation Layer is responsible for producing vendor-specific configuration files for both visualization and

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 6, June 2026

decision-support subsystems. Using the unified abstract traction model as its sole input, the framework generates HMI XML configurations based on geometric and visual attributes, while DSS XML configurations are derived from the inferred topological structure and electrical connectivity produced by the topology graph engine. By sourcing both outputs from the same authoritative model, the framework inherently enforces consistency between operator visualization and decision-support logic, eliminating discrepancies that commonly arise in manually engineered configurations.

XML generation within this layer is fully deterministic, ensuring that identical system models always result in identical configuration outputs across successive generation cycles. This property is essential for reliable validation, commissioning, and long-term maintenance.

The layer systematically translates visual layout information into HMI layout XML and serializes inferred topology and traversal semantics into DSS configuration XML, thereby providing a consistent and repeatable bridge between high-level system modeling and deployable SCADA configuration artifacts.

E. JSON Round-Trip Engineering Layer

In parallel with XML generation, the framework produces a JSON-based representation that captures the complete visual, hierarchical, and topological structure of the traction system. This JSON representation serves as an editable intermediate artifact, enabling engineers to reopen, modify, and regenerate configurations without the need to recreate the visual model from scratch. When a saved configuration is reloaded, the JSON data reconstructs the original virtual circuit exactly as it was designed, preserving spatial layout, component hierarchy, and connectivity semantics [10], [11], [13].

The JSON round-trip engineering layer preserves all geo-metric, hierarchical, and connectivity information required to maintain a faithful representation of the engineered system. It allows direct editing of component attributes and relationships at the model level, while ensuring that such modifications can be propagated consistently to regenerated XML outputs. This bidirectional transformation capability addresses a key limitation of traditional XML-centric SCADA workflows, in which configuration generation is typically one-way and post-generation edits are difficult to manage. By supporting lossless reconstruction and regeneration, the framework aligns closely with model-driven engineering principles and significantly improves maintainability across the configuration lifecycle.

V. VISUAL CIRCUIT MODELING AND VIRTUAL RECONSTRUCTION

The visual circuit modeling environment is designed to shift the engineer's focus from low-level configuration syntax to high-level electrical intent. Instead of manually authoring complex XML definitions, engineers interact with a graphical representation of the traction system that closely resembles conventional single-line diagrams used in railway electrification. System structure, connectivity, and operational semantics are defined intuitively through visual composition, while the framework transparently manages all underlying data representations.

When a configuration is saved, the visual model is serialized into a structured JSON-based intermediate representation rather than being stored as static graphics or XML output. This JSON representation preserves the complete system definition, including geometric layout, electrical topology, and hierarchical organization of stations, substations, equipment, and connections. Upon reopening, the framework reconstructs the original virtual circuit exactly as designed, restoring spatial layout, component relationships, and connectivity semantics without manual re-modeling [11], [13].

By enabling lossless reconstruction and regeneration, the framework supports true lifecycle-level editability of traction SCADA configurations. Engineers can iteratively refine layouts, modify equipment attributes, or update connectivity at any stage, with changes automatically propagated to downstream configuration artifacts. This round-trip capability significantly improves maintainability and revision handling compared to traditional one-way XML-centric workflows.

VI. AUTOMATED HMI XML GENERATION

The automated HMI XML generation process derives directly from the unified system model maintained by the framework, ensuring that graphical visualization remains fully consistent with the engineered traction circuit. Instead of requiring engineers to manually author coordinate-dependent XML files, the framework automatically translates visual and structural information captured during circuit modeling into schema-compliant HMI configuration data, thereby reducing engineering effort and minimizing human error in traction SCADA projects.

HMI XML generation is driven by geometric and visual attributes stored within the model, including component placement, line routing, spatial relationships, and display metadata [12], [15]. Station boundaries and layout regions are inferred from visual extents, while electrical equipment is rendered using symbols with precise coordinates derived from the visual editor. Interconnections are represented as ordered line segments, accurately reflecting routing and branching structures, and associated labels, annotations, and measurement elements are automatically linked to their respective equipment.

Because manual coordinate entry is eliminated, any con-figuration changes-whether introduced through visual editing or modification of the JSON-based intermediate representation-are automatically reflected when the HMI

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 6, June 2026

XML is regenerated. This guarantees synchronization between the engineered model and deployed visualization across design iterations, improving layout consistency, accelerating revision cycles, and enhancing the reliability of operator displays during commissioning and operation.

VII. DSS TOPOLOGY XML GENERATION

A. Topology Modeling for DSS

For Decision Support System (DSS) operation, the frame-work generates topology-oriented XML directly from the unified and authoritative system model. The topology graph engine transforms the abstract traction model into a structured representation that captures the electrical connectivity required for decision-support functions such as fault isolation, sectionalization, and restoration analysis. This representation remains independent of visual layout while staying fully consistent with the engineered circuit.

The generated DSS topology XML describes the traction network in terms of elementary sections and electrical connections between equipment and logical line segments. Elementary sections represent electrically continuous portions of the network and form the fundamental units for traversal and decision logic.

Connections define how these sections and associated equipment are linked, while directionality and traversal semantics are inferred automatically based on equipment type, connectivity rules, and traction context. Nodes within the topology are explicitly classified as source or non-source nodes to ensure correct initiation and propagation of traversal during DSS execution.

Because the DSS topology XML is derived from the same unified model used for HMI configuration, consistency between visualization and decision logic is enforced by construction. Any circuit modification-whether introduced through visual editing or JSON-based model updates-is automatically reflected in regenerated DSS topology XML, eliminating inconsistencies commonly associated with manually authored configurations [6], [7], [9].

B. Interlocking Modeling and Generation

Beyond electrical connectivity, safe traction system operation requires enforcement of operational constraints through interlocking logic. In the proposed framework, interlocking is modeled as a first-class configuration element within the unified system model, alongside electrical topology. This ensures that safety constraints remain inherently consistent with the underlying electrical structure rather than being applied as external or post-processing rules.

The framework supports commonly used interlocking schemes in traction and auxiliary power systems, including 2/3 interlocking, diamond interlocking, and sequential interlocking. In 2/3 interlocking, only two out of three circuit breakers may be closed simultaneously to prevent excessive parallel power paths. Diamond interlocking restricts unintended load transfer between adjacent or parallel lines, while sequential interlocking enforces ordered operational dependencies, typically allowing circuit breaker operation only when associated isolators are in predefined states.

Interlocking rules are defined by selecting relevant equipment nodes directly from the visual or hierarchical relational views and are stored within the unified model. These rules are serialized into the JSON-based intermediate representation and translated into dedicated interlocking XML elements during DSS configuration generation, coexisting with topology and connectivity definitions. By deriving interlocking and electrical topology from the same authoritative model, the framework ensures synchronization between operational constraints and decision logic, while allowing additional interlocking schemes to be incorporated without modifying the overall generation pipeline [6], [9].

VIII. HIERARCHICAL AND RELATIONAL EDITING

In addition to spatial circuit design, the proposed framework provides a hierarchical and relational editing capability that enables engineers to interact with the traction system independently of its visual layout. This hierarchical view organizes stations, substations, equipment, and electrical connections as logically related entities, abstracting system structure from screen geometry.

Table II. Comparison of Conventional XML-Based Configuration and the Proposed Framework

Aspect	Conventional Manual Approach	Proposed Visual–Topological Framework
Configuration artifacts	Separately authored HMI and DSS XML files	HMI and DSS XML generated automatically from a unified model
HMI layout development	Explicit manual definition of screen coordinates, sizes, and alignment	Graphical circuit design with layout parameters derived automatically
DSS topology modeling	Manual specification of connectivity and traversal relationships	Topology inferred automatically from the unified graph representation
Engineering effort	High effort due to repetitive, multi-file XML editing	Reduced effort through single-model modification and regeneration

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 6, June 2026

Aspect	Conventional Manual Approach	Proposed Visual–Topological Framework
Revision and update cycle	Typically requires hours due to coordinated file updates and validation	Typically completed in minutes via model update and regeneration
Debugging and commissioning	Frequent inconsistencies between HMI and DSS requiring iterative correction	Substantially reduced due to deterministic generation from a single source
Consistency enforcement	Relies on manual verification and engineering discipline	Guaranteed by construction through unified modeling

As a result, it supports configuration tasks that are inefficient or impractical to perform through purely graphical interaction, particularly in large and densely populated traction networks.

Through this hierarchical relational view, users can modify equipment attributes, update relationships, and adjust connectivity semantics without affecting the visual placement of components. Such changes are applied directly to the abstract traction model and are automatically propagated to the JSON-based intermediate representation and all derived XML outputs, including HMI and DSS configurations. This ensures consistent synchronization between hierarchical edits, visual representations, and generated configuration artifacts.

By combining spatial circuit modeling with hierarchical relational editing, the framework enables a dual interaction paradigm that improves usability and engineering efficiency. Engineers can select the most appropriate interaction mode based on the task, using visual editing for layout-centric design and hierarchical editing for structural or attribute-centric modifications. This flexibility is especially beneficial for large and complex traction SCADA configurations, where reliance on a single interaction mode can become cumbersome and error-prone [10], [14].

IX. CASE STUDY: METRO TRACTION SCADA DEPLOYMENT

The proposed framework was validated through its application to a production-scale metro traction SCADA deployment comprising multiple Receiving Substations (RSS) and Traction Substations (TSS) distributed along an operational metro corridor. Engineering activities for this deployment included the development of traction HMI displays, DSS topology configurations, and operational constraint definitions across several stations, reflecting the complexity typically encountered in urban metro traction networks.

Relative to conventional configuration workflows based on direct XML authoring [16], the proposed framework demonstrated a marked reduction in engineering effort and configuration turnaround time while producing functionally equivalent HMI and DSS XML artifacts. In traditional workflows, engineers are required to manually define graphical layout parameters-such as coordinates, dimensions, and alignments-and to explicitly encode DSS connectivity and traversal logic across multiple XML files. By contrast, the proposed approach enables engineers to design the system using visual circuit diagrams, from which both HMI and DSS configurations are generated automatically.

The advantages of the framework were most pronounced during configuration updates. Under manual workflows, even minor modifications typically necessitated coordinated edits across several XML artifacts followed by repeated validation, leading to turnaround times on the order of hours. Using the proposed framework, comparable changes were applied at the model level and propagated through automatic regeneration, reducing revision turnaround time to minutes and enabling rapid, controlled iteration without low-level rework.

Commissioning and debugging effort was also reduced. Because HMI and DSS configurations were generated deterministically from a single authoritative system model, in-consistencies between operator visualization and decision-support logic were effectively avoided. This resulted in fewer integration defects and reduced iterative debugging during system integration, thereby simplifying validation activities and improving deployment reliability.

Table II summarizes the qualitative and relative quantitative differences between manual XML-based configuration and the proposed framework. Overall, the case study demonstrates that the framework scales effectively to multi-station metro traction environments and provides a practical, time-efficient, and robust solution for large-scale traction SCADA configuration engineering [1], [2], [8].

X. CONCLUSION AND FUTURE WORK

This paper presents a unified visual–topological modeling framework for railway traction SCADA systems, addressing a long-standing challenge in configuration engineering. By treating visual circuit design as the single authoritative source and deriving both Human–Machine Interface (HMI) and Decision Support System (DSS) configurations from a shared abstract model, the proposed approach eliminates redundant specification efforts, reduces manual intervention, and enforces consistency between visualization and operational logic by construction. The introduction of a JSON-based intermediate representation further enables round-trip engineering, allowing configurations to be reopened, modified, and regenerated across the system lifecycle

without loss of structural or se-mantic information. In addition, the inclusion of hierarchical and relational editing capabilities complements spatial modeling and enhances usability for large and complex traction deployments.

Although the framework demonstrates clear practical advantages, several aspects remain open for further enhancement. XML configuration schemas for HMI and DSS are inherently vendor-specific, necessitating adaptation of the XML generation layer when targeting different SCADA platforms. This limitation motivates future work toward greater abstraction and standardization, including potential alignment with Common Information Model (CIM)–based representations. Moreover, advanced DSS behaviors-such as project-specific protection philosophies, complex interlocking variants, and customized traversal strategies -may require targeted extensions to the topology modeling and rule-inference mechanisms, which will be investigated in subsequent implementations [18].

The modeling process also assumes a baseline level of traction domain expertise to ensure that electrical intent and operational constraints are captured correctly. Future work will therefore focus on strengthening validation support through rule-based and simulation-driven checks, assisting engineers during model creation and reducing reliance on expert intervention. Additional planned extensions include automated RTU and I/O mapping, tighter integration of simulation-based DSS validation, and support for version-controlled configuration management to enable collaborative engineering and long-term maintainability [10], [13].

Overall, the proposed framework provides a scalable and ex-tensible foundation for modern traction SCADA configuration engineering. By unifying visual modeling, topology reasoning, interlocking logic, and round-trip configuration management within a single model-driven workflow, it offers a robust alternative to traditional XML-centric practices and establishes a clear pathway for future research and industrial adoption.

REFERENCES

[1] R. White and M. Baxter, "SCADA Systems for Railway Electrification," *IEE Proceedings – Electric Power Applications*, vol. 150, no. 1, pp. 1–8, Jan. 2003.

[2] A. Steimel, *Electric Traction – Motive Power and Energy Supply*. Munich, Germany: Oldenbourg Verlag, 2008.

[3] IEC 61970-301, "Energy Management System Application Program Interface (EMS-API) – Common Information Model (CIM) Base," International Electrotechnical Commission, Geneva, Switzerland, 2017.

[4] IEC 61968-1, "Application Integration at Electric Utilities – System Interfaces for Distribution Management," International Electrotechnical Commission, Geneva, Switzerland, 2019.

[5] IEC 61850-1, "Communication Networks and Systems for Power Utility Automation," International Electrotechnical Commission, Geneva, Switzerland, 2013.

[6] A. Bose, "Smart Transmission Grid Applications and Their Supporting Infrastructure," *IEEE Transactions on Smart Grid*, vol. 1, no. 1, pp. 11–19, Jun. 2010.

[7] J. Zhu, *Optimization of Power System Operation*, 2nd ed. Hoboken, NJ, USA: Wiley–IEEE Press, 2015.

[8] P. Weston, C. Roberts, G. Yeo, and E. Stewart, "Energy and Power Management for Railway Traction Systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 7, pp. 1988–1998, Jul. 2016.

[9] P. Kundur, *Power System Stability and Control*. New York, NY, USA: McGraw–Hill, 1994.

[10] D. C. Schmidt, "Model-Driven Engineering," *IEEE Computer*, vol. 39, no. 2, pp. 25–31, Feb. 2006.

[11] B. Selic, "The Pragmatics of Model-Driven Development," *IEEE Soft-ware*, vol. 20, no. 5, pp. 19–25, Sep./Oct. 2003.

[12] M. R. Endsley, "Toward a Theory of Situation Awareness in Dynamic Systems," *Human Factors*, vol. 37, no. 1, pp. 32–64, 1995.

[13] T. Stahl, M. Vo¨lter, and K. Czarnecki, *Model-Driven Software Development: Technology, Engineering, Management*. Chichester, U.K.: Wiley, 2006.

[14] A. Fay and S. Mauch, "Engineering of Automation Systems," in *Proceedings of the IFAC World Congress*, Milan, Italy, 2011, pp. 133–138.

[15] ISO 11064-1, "Ergonomic Design of Control Centres – Part 1: Principles for the Design of Control Centres," International Organization for Standardization, Geneva, Switzerland, 2000.

[16] S. Mackay, E. Wright, D. Reynders, and J. Park, *Practical Industrial Data Communications: Best Practice in Industrial Data Communications. Oxford*, U.K.: Newnes, 2004.

[17] W. Bolton, *Programmable Logic Controllers and Industrial Automation. Oxford*, U.K.: Elsevier, 2015.

[18] Forward and T. C. Lethbridge, "Problems and Opportunities for Model-Centric versus Code-Centric Development," in *Proc. Int. Conf. Software Engineering (ICSE)*, Leipzig, Germany, 2008, pp. 27–36