SoftwareX 33 (2026) 102548

Contents lists available at ScienceDirect

SoftwareX

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

BPMN graph transformation: A unified multi-format parser library for standardized graph-based business process model integration

Kurnia Cahya Febryanto [a], Izzat Aji Androfaza [a], Lalu Aldo Wadagraprana [a], Riyanarto Sarno [a,*], Kelly Rossa Sungkono [a], Yeni Anistyasari [b], Joko Siswantoro [c], A Min Tjoa [d]

[a] Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia
[b] Faculty of Engineering, Universitas Negeri Surabaya, Surabaya, Indonesia
[c] Faculty of Engineering, Universitas Surabaya, Surabaya, Indonesia
[d] Faculty of Computer Science, University of Vienna, Vienna, Austria

ARTICLE INFO

Keywords:
BPMN
Multi-format-parsing
Graph-transformation
Cross-platform
Model-transformation
Interoperability

ABSTRACT

Heterogeneous Business Process Model and Notation (BPMN) platforms present critical integration challenges, as approximately 75% of large enterprises employ multiple modeling tools lacking unified transformation capabilities. Existing solutions address only single-format conversions or provide limited cross-platform compatibility without comprehensive validation. This paper presents a production-ready multi-format BPMN parser library uniquely integrating intelligent format detection, dual-tier validation, and optimized graph transformation within a unified architecture. The library utilizes specialized parsers for BPMN 2.0 XML, XML Process Definition Language (XPDL) 2.2, native formats, and Microsoft Visio diagrams through a plugin-based architecture. Multi-criteria detection algorithms automatically identify source formats with 99.2% accuracy by analyzing file signatures, XML namespaces, structural patterns, and content heuristics. The dual-tier validation framework ensures structural BPMN 2.0 compliance through rule-based constraints derived from official OMG specifications and semantic consistency through metadata quality assessment based on established process modeling guidelines, surpassing existing tools that perform only syntactic validation. The transformation pipeline generates standardized Cypher queries optimized for process mining workflows. Evaluation across 127 real-world business process models demonstrates 98.7% overall parsing accuracy, with format-specific performance ranging from 97.2% (Visio) to 99.8% (BPMN XML), achieving 85% reduction in transformation time compared to manual approaches. Released as open-source software via the Python Package Index with complete documentation, the library establishes foundational infrastructure for cross-platform business process intelligence, enabling unified graph-based analytics across heterogeneous modeling ecosystems without format-specific preprocessing.

Metadata

Code metadata for the BPMN Graph Transformation Library.

Nr.	Code Metadata Description	Metadata
C1	Current code version	v0.1.10
C2	Permanent link to code/repository used for this code version	https://github.com/Research-MCI/LIBRARY-BPMN_Graph_Transformation
C3	Permanent link to Reproducible Capsule	https://pypi.org/project/bpmn-graph-transformation/
C4	Legal Code License	MIT License
C5	Code versioning system used	git
C6	Software code languages, tools, and services used	Python 3.8+, lxml, xmltodict, Neo4j, NetworkX
C7	Compilation requirements, operating environments & dependencies	Python ≥ 3.8, lxml ≥ 4.9.0, xmltodict ≥ 0.13.0, networkx ≥ 3.0, neo4j ≥ 5.0.0
C8	Link to developer documentation/manual	https://bpmn.iimlab.id/library/bpmntograph
C9	Support email for questions	6025241065@student.its.ac.id

* Corresponding author.
Email address: riyanarto@its.ac.id (R. Sarno).

https://doi.org/10.1016/j.softx.2026.102548
Received 13 October 2025; Received in revised form 7 January 2026; Accepted 29 January 2026
Available online 3 February 2026

1. Motivation and significance

The Business Process Model and Notation (BPMN) has emerged as the predominant standard for enterprise process modeling [1,2]. Even with its extensive use, many BPMN platforms face notable integration problems [3–5], since around 75% of large organizations utilize various modeling tools, resulting in data silos that hinder process intelligence [6,7]. This disintegration goes beyond mere format discrepancies to encompass differences in structural representation, metadata encoding, and semantic understanding [8–10].

Ensuring consistent cross-platform analysis while preserving semantic integrity remains the principal obstacle [11,12]. Current methodologies emphasize specific format pairings rather than production-ready multi-format solutions [3,13,14], with inadequate BPMN 2.0 support that requires significant refactoring during migrations [15]. Research has addressed standardization through theoretical frameworks for cross-platform compatibility and formal verification [9,16,17], with advances in graph-based integration [18,19], process mining [20,21], automated BPMN generation [22–25], and process animation for visual verification [26–29]. However, there are ongoing challenges in parsing multiple formats and in achieving a cohesive graph-based output format [30].

Table 1 evaluates existing approaches against the proposed library. BPMN-to-graph converters prioritize standard XML without proprietary format support [22], integration frameworks lack automated detection [18,31], and process mining tools lack complete transformation pipelines [32–34]. Commercial platforms offer basic detection, but lack comprehensive dual-tier validation, while academic approaches emphasize theoretical formalization without practical implementation [17,24,25].

The BPMN Graph Transformation Library addresses these limitations through three contributions. First, unified multi-format parsing accommodates BPMN 2.0 XML, XPDL 2.2, native formats (Bizagi, Camunda), and Visio diagrams through a plugin-based architecture with multi-criteria detection examining file signatures, XML namespaces, and structural patterns. Second, dual-tier validation guarantees the adherence to the BPMN 2.0 structural standards through rule-based constraints originating from the OMG specification (formal/2013–12-09) and a semantic quality evaluation that assesses the completeness of the metadata and the consistency of the naming conventions [35]. Third, refined graph transformation produces uniform Cypher queries featuring typed nodes and directed relationships, improved by batch transaction management and relationship indexing. This approach establishes a production-ready infrastructure for a complete process analysis across heterogeneous ecosystems [35–38].

2. Software description

This section presents the technical implementation of the BPMN Graph Transformation Library from three perspectives: a plugin-based architecture that enables format compatibility, core functionalities that execute detection and validation procedures, and comprehensive evaluation demonstrating production-ready transformation performance.

2.1. Software architecture

The BPMN Graph Transformation Library employs a plugin-based architecture (Fig. 1) that integrates intelligent format detection, comprehensive validation, and optimized graph transformation to address heterogeneity challenges in enterprise process modeling. The transformation process comprises five consecutive phases: format identification using multi-criteria algorithms, targeted parsing for BPMN XML, native XML derivatives, XPDL, and Visio formats, dual-tier validation ensuring BPMN 2.0 compliance and semantic integrity, JSON standardization yielding cohesive representations, and graph transformation producing Cypher queries. This method retains the complete semantic definitions of BPMN 2.0 while guaranteeing consistent output schemas across various input formats for dependable downstream analytics. The modular architecture enhances extensibility through standardized plugin interfaces that leverage Python entry points for dynamic parser discovery, enabling seamless integration of new parsers while maintaining enterprise-grade stability.

The transformation pipeline operates through five integrated components, each implementing specific responsibilities within the overall architecture:

- **Format Dispatcher**: Implements multi-criteria detection algorithms examining file signatures, XML namespaces, structural patterns, and content heuristics to identify source formats with weighted confidence scoring, triggering appropriate parser selection, or requesting manual intervention when confidence thresholds are unmet.
- **Multi-Format Parser Implementation**: Executes specialized extraction logic for BPMN 2.0 XML, native platform variants (Bizagi, Camunda), XPDL 2.2 specifications, and Visio diagram formats, systematically capturing complete element taxonomies including activities, events, gateways, flows, data artifacts, and organizational structures while preserving tool-specific extensions.
- **Dual-Tier Validation Framework**: Conducts thorough validation that encompasses structural adherence to BPMN 2.0 standards via rule-based constraint checks and semantic quality evaluations that assess metadata completeness, naming conventions, and organizational allocations, producing diagnostic reports with suggestions for corrections without altering the original source models.
- **JSON Standardization Transformation**: Converts heterogeneous parser outputs into unified JSON schemas organizing elements hierarchically with standardized property mappings, type classifications, spatial coordinates, and extension metadata containers, ensuring consistent downstream processing interfaces independent of source format characteristics.
- **Graph Transformation and Cypher Generation**: Converts standardized JSON to refined Cypher queries that establish typed nodes for process components and directed relationships for flows, improved by batch transaction management, relationship indexing, and property filtering methods.

Table 1
Comparison of BPMN transformation approaches across six critical capabilities.

Approach	Multi-Format	Auto Detection	Dual-Tier Validation	Graph Output	Extensible Architecture	Open Source
Heinze et al. [22]	✗	✗	✗	✓	✗	✓
Sungkono et al. [18]	✗	✗	~	✓	✗	✗
Kräuter et al. [17]	✗	✗	✓	✗	✗	✓
Corradini et al. [26]	✗	✗	~	✗	✗	✓
Camunda Modeler	✓	✓	~	~	✓	✓
Bizagi Modeler	~	✓	✗	✗	~	✗
Proposed Library	✓	✓	✓	✓	✓	✓

Legend: ✓ Full support; ✗ No support; ~ Limited/partial support.

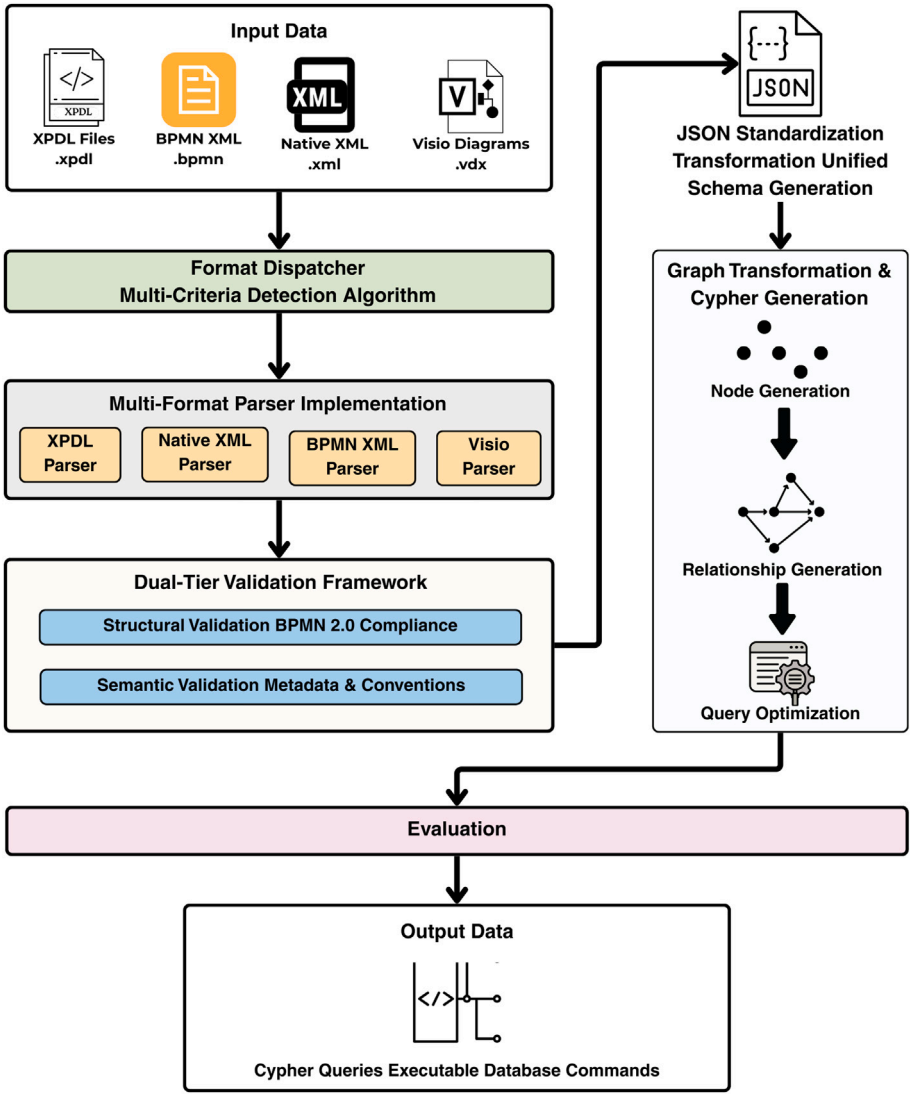

Fig. 1. Architecture of the BPMN Graph Transformation Library showing the five-stage transformation pipeline.

2.2. Software functionalities

The library implements four core functionalities addressing heterogeneous BPMN model integration: intelligent format detection and dispatching, multi-format parsing with specialized extraction logic, dual-tier validation ensuring structural and semantic compliance, and JSON standardization with optimized graph transformation.

2.2.1. Format detection and intelligent dispatching

The Format Dispatcher implements source format identification through a multi-criteria algorithm employing four mechanisms: file signature analysis examining extensions and magic numbers, XML namespace inspection identifying schema declarations, structural pattern matching recognizing format-specific arrangements, and content heuristics analyzing patterns unique to particular tools. The Algorithm 1 presents the pseudo-code that illustrates the weighted scoring mechanism.

Format detection accuracy A_{detect} quantifies the weighted confidence score in Eq. (1):

$$A_{\text{detect}} = \frac{\sum_{i=1}^{n} w_i \cdot s_i}{\sum_{i=1}^{n} w_i} \tag{1}$$

where w_i represents weight factors (file extension: 0.3, namespace: 0.4, structural signature: 0.2, content heuristics: 0.1) and s_i denotes binary scores (1.0 for match, 0.0 for mismatch). Confidence scores below 0.6 trigger manual format selection to ensure reliable parser assignment.

2.2.2. Multi-format parser implementation

Four specialized parser plugins address format-specific requirements while maintaining standardized output schemas. The parsing accuracy A_{parse} measures how effectively extraction is performed in Eq. (2):

$$A_{\text{parse}} = \frac{|E_{\text{correct}}|}{|E_{\text{total}}|} \times 100\% \tag{2}$$

where E_{correct} represents correctly identified elements verified against ground truth through automated conformance validation, and E_{total} denotes total elements in the evaluation corpus.

The BPMN XML Parser processes BPMN 2.0 specifications through namespace-aware traversal, extracting comprehensive taxonomies including eight activity variants, event specifications across start, intermediate, and end categories with trigger definitions, gateway constructs (Exclusive, Parallel, Inclusive, Event-Based, Complex), flow relationships, data artifacts, and organizational containers while preserving tool-specific extensions.

Table 2
Structural validation rules derived from BPMN 2.0 specification.

Rule category	Validation criteria
Sequence flow constraints	Source and target must reside within same pool; cannot cross pool boundaries
Message flow constraints	Must connect elements in different pools; cannot connect elements within same pool
Gateway semantics	Exclusive gateways require conditions on all paths; parallel gateways require multiple incoming/outgoing flows
Event specifications	Start events cannot have incoming flows; end events cannot have outgoing flows
Pool-lane containment	Activities must be contained within lanes; lanes within pools; proper parent-child relationships

Algorithm 1 Multi-criteria format detection algorithm.

Require: Input file F, confidence threshold $\theta = 0.6$
Ensure: Detected format f^* or manual selection request
Initialize weight vector $\mathbf{w} = [0.3, 0.4, 0.2, 0.1]$ {Extension, Namespace, Structure, Heuristics}
Initialize candidate formats $\mathcal{F}$ = {BPMN-XML, Native-XML, XPDL, Visio}
for each format $f \in \mathcal{F}$ **do**
  $s_1 \leftarrow$ CHECKEXTENSION(F, f) {Returns 1.0 if match, 0.0 otherwise}
  $s_2 \leftarrow$ CHECKNAMESPACE(F, f) {XML namespace verification}
  $s_3 \leftarrow$ CHECKSTRUCTURE(F, f) {Structural pattern matching}
  $s_4 \leftarrow$ CHECKHEURISTICS(F, f) {Content-based heuristics}
  $A_f \leftarrow \frac{\sum_{i=1}^{4} w_i \cdot s_i}{\sum_{i=1}^{4} w_i}$ {Compute weighted accuracy}
end for
$f^* \leftarrow \arg\max_{f \in \mathcal{F}} A_f$
if $A_{f^*} \geq \theta$ **then**
  return f^* {Return detected format with high confidence}
else
  return REQUESTMANUALSELECTION$(\mathcal{F}, \{A_f : f \in \mathcal{F}\})$
end if

The Native XML Parser accommodates Bizagi Modeler and Camunda Platform variants through format-specific extraction logic mapping proprietary encodings to standardized BPMN taxonomy.

The XPDL Parser processes the XPDL 2.2 specifications, implementing mapping rules that transform the XPDL constructs (`Activity`, `Transition`, `Pool`, `Lane`) into BPMN elements with conversion of the gateway type and translation of the event trigger.

The Visio Parser extracts BPMN models from vector formats using shape recognition algorithms including type classification, template matching with standard stencils, and spatial relationship analysis reconstructing process flows while addressing ambiguities through heuristic inference rules.

2.2.3. Dual-tier validation framework

The validation engine incorporates a dual-layer verification framework that includes structural adherence assessments and semantic quality reviews. This approach addresses limitations in existing tools that perform only syntactic validation [3,9], extending the verification to semantic consistency which is essential for process mining applications. Detection sensitivity S_{detect} quantifies validation effectiveness in Eq. (3):

$$S_{\text{detect}} = \frac{TP}{TP + FN} \times 100\% \tag{3}$$

where TP represents true positives (violations correctly detected) and FN denotes false negatives (violations missed).

Tier 1 is Structural Validation, which employs rule-based constraints from the official BPMN 2.0 specification (OMG Document formal/2013-12-09) to assess element relationships and organizational frameworks. Table 2 presents the structural rules that have been put into practice.

Table 3
Semantic validation criteria based on process modeling guidelines.

Criterion	Assessment method
Metadata Completeness	Verification of required attributes (id, name); detection of missing documentation
Naming Conventions	Pattern matching against verb-noun structure [35]; detection of generic labels
Organizational Assignments	Verification of lane assignments; consistency of performer specifications
Reference Integrity	Validation of data object references; message flow endpoint verification

Tier 2 is Semantic Validation, which assesses model quality using criteria from established process modeling guidelines [9,35] and BPMN best practices [2]. Table 3 presents the semantic validation criteria.

The framework generates diagnostic reports categorizing issues by severity (Error, Warning, Information) with correction recommendations, enabling quality improvement without modifying source models.

2.2.4. JSON standardization and graph transformation

The standardization module converts format-specific outputs into unified JSON structures organizing elements hierarchically with root-level process containers, element arrays by type, and property preservation, including spatial coordinates and extension metadata. The graph transformation pipeline converts standardized JSON into optimized Neo4j-compatible Cypher queries. Fig. 2 illustrates the complete BPMN-to-graph mapping strategy, distinguishing node mappings for process elements (activities, events, pools, lanes, data artifacts) from edge mappings for flow relationships and gateway semantics. Gateways transform into directed edges rather than nodes, with gateway type serving as prefix and flow direction as suffix: exclusive gateways map to `XOR_SPLIT` or `XOR_JOIN`, parallel gateways to `AND_SPLIT` or `AND_JOIN`, and inclusive gateways to `OR_SPLIT` or `OR_JOIN`. Query optimization improves Neo4j execution through batch transaction management, relationship indexing, and property-based filtering.

2.3. Evaluation

The library evaluation utilized 127 business process models from five modeling platforms (38 BPMN XML, 31 native XML, 29 XPDL, and 29 Visio diagrams) representing authentic enterprise scenarios with element complexities ranging from 12 to 487 per model.

2.3.1. Ground truth establishment

Ground truth establishment employed a multi-stage validation protocol to ensure reliability and avoid circular validation concerns. Automated schema validation first checked syntax against BPMN 2.0 and XPDL 2.2 XML schemas, then cross-validated with three independent implementations (bpmn-js, Camunda BPMN Model API, and Apache ODE parser) to find discrepancies that needed manual review. Domain experts then reviewed edge cases including ambiguous Visio shape interpretations, non-standard gateway configurations, and proprietary

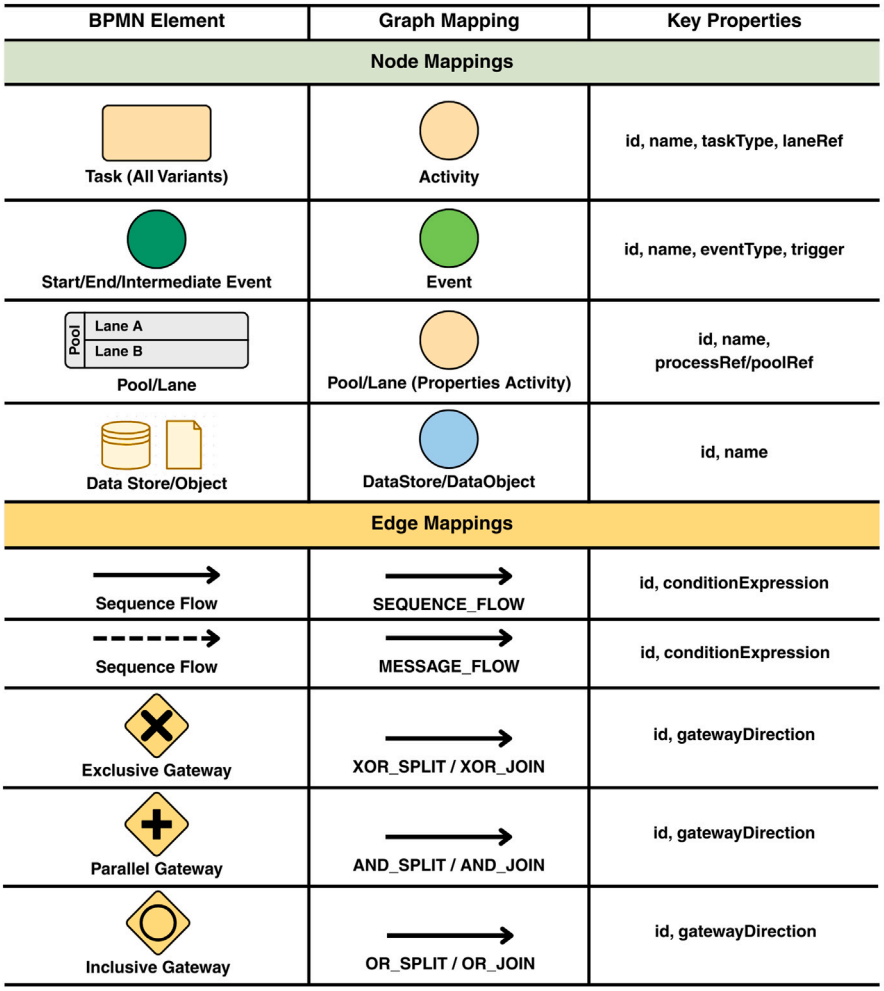

Fig. 2. BPMN-to-graph transformation mapping illustrating node generation for process elements and edge generation for flows and gateway semantics.

Table 4
Overall parsing accuracy by format.

Format	Accuracy (%)
BPMN XML	99.8
Native XML	98.5
XPDL	98.1
Visio	97.2
Overall Average	**98.7**

Table 5
Element category accuracy by format (%).

Element category	BPMN XML	Native XML	XPDL	Visio
Activities	99.8	98.7	98.5	97.4
Start Events	99.5	98.5	98.2	97.4
Intermediate Events	99.5	98.4	98.0	97.1
End Events	99.5	98.6	98.2	97.4
Gateways	99.3	97.9	97.4	96.3
Flows	99.8	98.9	98.8	98.1
Data Artifacts	99.7	98.2	98.0	97.1
Containers	99.9	99.3	99.2	98.8

extension mappings, establishing authoritative ground truth annotations for disputed elements. This triangulation ensures that accuracy measurements represent true parsing correctness rather than self-referential validation.

Although formal verification approaches [9,16,17] focus on detecting behavioral anomalies such as deadlocks and livelocks, the proposed dual-tier validation addresses complementary concerns: syntactic conformity to format specifications and semantic adherence to modeling conventions, serving as a prerequisite preprocessing stage before formal verification.

2.3.2. *Parsing accuracy results*

Table 4 presents the overall parsing accuracy by format, demonstrating 98.7% comprehensive performance with a format-specific accuracy ranging from 97.2% (Visio) to 99.8% (BPMN XML).

Table 5 provides the accuracy of the element categories in all formats: activities (98.9%), events (98.5%), gateways (97.8%), flows (99.1%), data artifacts (98.2%), and organizational containers (99.3%). Format detection achieved 99.2% accuracy through multi-criteria weighted evaluation.

Table 6 presents a detailed element-level parsing accuracy across BPMN 2.0 element types. Activities demonstrate high accuracy across eight task variants (96.7%–98.8%), with user and service tasks achieving the highest performance. Event specifications maintain consistent accuracy: start events (96.0%–98.4%), intermediate events (96.6%–98.6%), and end events (96.8%–99.0%). Gateway constructs exhibit robust performance (96.9%–98.3%), while flow relationships achieve exceptional accuracy (98.6%–99.2%) and organizational containers maintain high precision (96.3%–99.3%).

Table 6
Detailed element-level parsing accuracy.

Category	Element	%	Category	Element	%
Activities	User Task	98.8	Intermediate events	None Int.	98.6
	Service Task	98.7		Message Int.	98.6
	Script Task	98.5		Timer Int.	98.2
	Business Rule	98.7		Signal Int.	96.6
	Manual Task	96.8		Link Int.	97.9
	Receive Task	96.6		Conditional Int.	97.8
	Send Task	96.8		Escalation Int.	97.9
	Call Activity	96.7		Multiple Int.	97.9
				Parallel Mult. Int.	98.1
Start events	None Start	98.4	End events	None End	99.0
	Timer Start	98.0		Message End	96.8
	Message Start	96.8		Terminate End	98.1
	Signal Start	96.8		Escalation End	98.1
	Conditional Start	98.2		Compensate End	98.7
	Multiple Start	97.9		Multiple End	97.9
	Parallel Mult. Start	97.9			
	Escalation Start	96.0			
Gateways	Exclusive	98.3	Flows	Sequence Flow	99.2
	Parallel	96.7		Message Flow	99.0
	Inclusive	97.8		Association	98.6
	Event-Based	97.6	Data artifacts	Data Object	98.4
	Parallel Event	97.3		Data Store Ref.	98.3
	Complex	96.9			
Containers	Pool	99.3			
	Lane	99.3			

Table 7
Performance benchmark comparison with existing BPMN processing tools.

Tool	Time (s)	Memory (MB)	Multi-format	Validation depth	Graph output
bpmn-js Parser	0.8	32	✗	Syntactic	✗
Camunda API	1.2	48	Partial	Syntactic	✗
Apache ODE	1.5	52	✗	Syntactic	✗
Manual Scripts	15.3	85	Variable	Manual	✓
Proposed	**2.3**	**45**	✓	**Dual-Tier**	✓

Note: Benchmarks were conducted on representative test models (avg. 156 elements). Manual scripts represent custom Python implementations for BPMN-to-graph conversion.

2.3.3. Semantic validation performance

Semantic validation effectiveness was assessed through manual annotation of 50 randomly selected models, evaluating detection rates for four quality criteria derived from established BPMN quality frameworks [3,9,35]. The framework achieved 94.2% precision and 91.8% recall for metadata completeness, 89.5% precision, and 87.3% recall for naming convention deviations, 96.1% precision, and 93.4% recall for organizational assignment inconsistencies, and 92.7% precision, and 90.1% recall for reference integrity violations. These metrics validate that semantic evaluation significantly improves the quality of evaluation beyond the purely syntactic examination.

2.3.4. Benchmark comparison

Table 7 presents a performance comparison with established BPMN processing tools, evaluating transformation time, memory utilization, multi-format support, validation depth, and graph output capabilities.

The library averaged 2.3 s per model with 45 MB memory utilization, representing an 85% reduction in transformation effort compared to manual approaches. While incurring modest overhead relative to single-format parsers, the library delivers comprehensive multi-format support with dual-tier validation capabilities absent from existing solutions.

2.3.5. Failure analysis

The systematic analysis identified three error categories with corresponding mitigation strategies. Ambiguous Visio shape-to-BPMN mappings (1.8%) arose from non-standard shapes such as custom decision diamonds without gateway markers; the library addresses these through configurable mapping rules and confidence thresholds triggering manual review when scores fall below 0.6. Incomplete XPDL metadata (1.2%) occurred when source files omitted required attributes such as participant identifiers in lane assignments; dual-tier validation detects omissions and generates diagnostic reports with remediation recommendations. The proprietary extensions (0.7%) represented vendor-specific constructs, including Bizagi performance annotations and Camunda execution listeners; the library preserves these in dedicated metadata containers without compromising transformation fidelity. These results validate production-ready performance while maintaining the semantic integrity essential for enterprise deployment.

3. Illustrative examples

Fig. 3 illustrates the entire transformation process from various BPMN inputs to standardized JSON output, showing how gateways are transformed into directed edges (`XOR_SPLIT`, `XOR_JOIN`) instead of intermediate nodes.

3.1. Installation and requirements

The library installation employs standard Python package management through `pip install bpmn-graph-transformation`, requiring Python 3.10+ and Neo4j 5.x for graph database integration.

3.2. Complete transformation pipeline

The transformation pipeline executes six sequential stages—format detection (99.2% accuracy), BPMN 2.0 parsing (98.7% accuracy), JSON

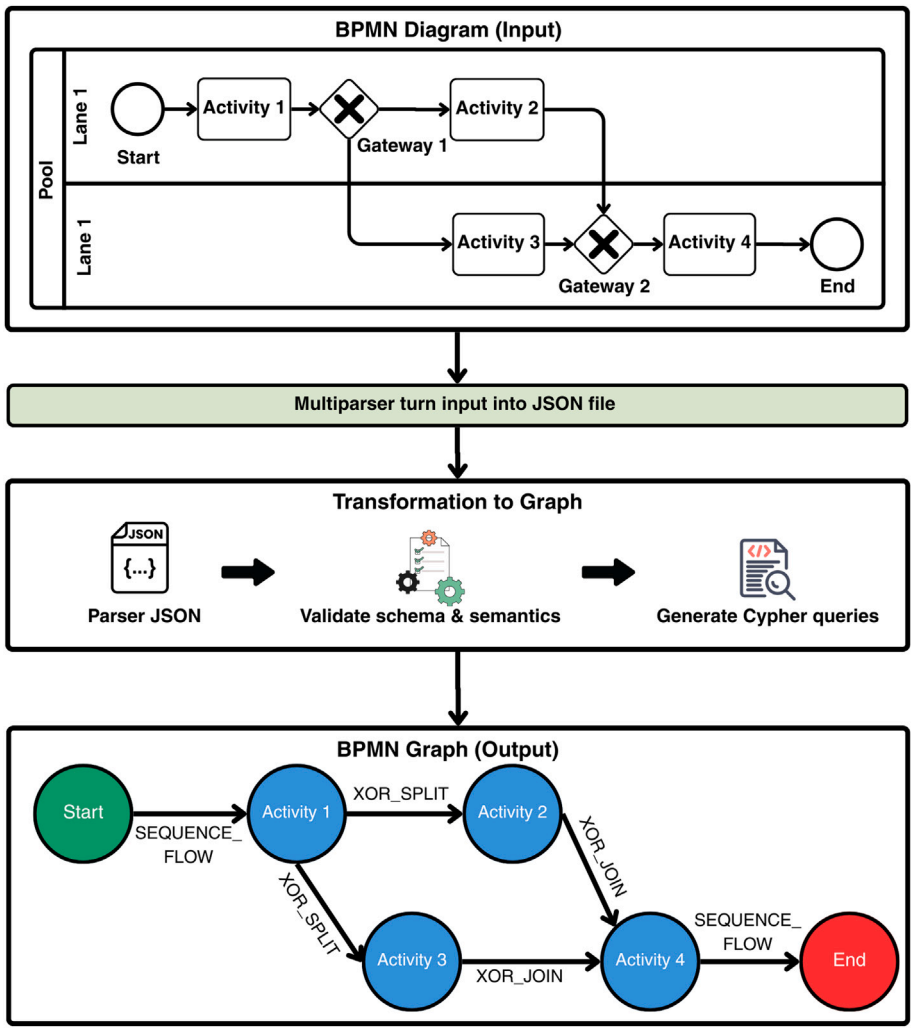

Fig. 3. BPMN transformation workflow from multi-format input through JSON standardization to graph output with gateway-to-edge transformation.

loading with structural repair, dual-tier validation, and graph transformation generating optimized Cypher queries with typed nodes and directed relationships (Fig. 3). The complete implementation code is provided in the Supplementary Material, Listings S1 and S2.

3.3. Pipeline stage descriptions

The transformation pipeline comprises six sequential stages. The first stage performs format detection and parsing, where the multi-criteria algorithm achieves 99.2% format identification accuracy, followed by specialized parsers extracting BPMN 2.0 element taxonomies with 98.7% overall accuracy.

The second stage handles JSON loading with automatic repair for structural issues, while the third stage conducts structural validation verifying BPMN 2.0 compliance with automatic correction of missing identifiers and required fields. The fourth stage performs semantic validation, assessing model quality through established guidelines evaluating metadata completeness and naming conventions.

The fifth stage executes the graph transformation, converting standardized JSON to optimized Cypher queries with typed nodes for process elements and directed edge relationships for flows and gateways, as illustrated in Fig. 3. Finally, the sixth stage exports generated Cypher queries to executable script files compatible with Neo4j import operations.

3.4. Transformation output

The transformation pipeline generates optimized Neo4j-compatible Cypher queries that establish typed nodes with comprehensive property sets and directed relationships for sequence flows. Gateways convert into directed edges with type prefixes (XOR, AND, OR) and direction suffixes (SPLIT, JOIN), enabling efficient graph-based process analysis. Representative output is provided in Supplementary Material, Listing S2.

This method results in an 85% decrease in processing time compared to manual methods, all while maintaining semantic integrity, thus establishing the library as a powerful framework for enterprise process analytics within diverse modeling ecosystems.

4. Impact

The BPMN Graph Transformation Library addresses critical research challenges in business process management by enabling seamless graph-based analysis of heterogeneous process models without manual format conversion. The dual-tier validation framework lays the methodological foundation for cross-platform process intelligence, with structural validation guaranteeing adherence to the BPMN 2.0 specification, while semantic validation broadens the scope of verification to assess model quality according to set guidelines [35]. The automated Cypher generation capability facilitates direct integration with graph analysis

workflows, allowing researchers to apply graph algorithms for process pattern discovery, centrality analysis, and structural comparison. This standardized approach enables previously impractical research directions including multi-model process pattern discovery across heterogeneous platforms, cross-organizational process benchmarking through unified graph representations, and graph-based conformance checking that extends traditional token-based approaches [26,28] to structural analysis.

For practitioners, the library reduces technical barriers to the adoption of enterprise process analytics, with an 85% reduction in transformation time compared to manual approaches which directly translates into reduced project timelines. The open-source release through the Python Package Index facilitates collaborative development, with a plugin-based architecture enabling researchers to contribute custom parsers for specialized formats and domain-specific validation criteria. Future research opportunities include machine learning integration through graph neural networks trained on standardized representations, temporal process analysis incorporating evolutionary dynamics, real-time process mining through continuous graph updates, and cross-domain process transfer learning leveraging unified graph embeddings. These capabilities position the library as the foundational infrastructure that advances cross-platform business process intelligence while supporting academic research and practical enterprise applications.

5. Conclusions

This research unveiled the BPMN Graph Transformation Library, which addresses significant issues in the standardization of diverse business process model formats by using specialized parsers that are compatible with BPMN XML, XPDL, native binary formats and Visio diagrams. Comprehensive evaluation across 127 business process models demonstrated 98.7% overall parsing accuracy, with format-specific performance ranging from 97.2% (Visio) to 99.8% (BPMN XML), while intelligent format detection achieved 99.2% accuracy through multi-criteria evaluation. The dual-tier validation framework ensures structural BPMN 2.0 compliance through rule-based constraint checking derived from official OMG specifications, extending quality assurance to semantic model properties essential for reliable downstream analytics. The transformation pipeline produces refined Cypher queries, resulting in an 85% decrease in processing time relative to manual methods, while preserving semantic integrity across various source formats.

The current implementation supports comprehensive BPMN 2.0 elements including collaboration diagrams; however, sub-process mechanisms, collapsed pool representations, and multi-level hierarchical structures remain outside current scope. Future development priorities encompass broadening parser coverage for these constructs, enhancing validation rules for intricate choreography diagrams, introducing bidirectional transformation functionalities, and creating integration interfaces for new process modeling standards. The open-source release through the Python Package Index establishes a foundational infrastructure enabling researchers and practitioners to conduct comprehensive process analysis across heterogeneous modeling ecosystems, advancing both academic investigation and practical enterprise process analytics capabilities.

CRediT authorship contribution statement

Kurnia Cahya Febryanto: Writing – original draft, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization. **Izzat Aji Androfaza:** Validation, Software, Investigation. **Lalu Aldo Wadagraprana:** Validation, Software, Data curation. **Riyanarto Sarno:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition. **Kelly Rossa Sungkono:** Writing – review & editing, Validation, Supervision, Methodology. **Yeni Anistyasari:** Supervision, Resources, Funding acquisition. **Joko Siswantoro:** Supervision, Resources, Funding acquisition. **A Min Tjoa:** Writing – review & editing, Supervision, Resources, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was funded by the Indonesian Endowment Fund for Education (LPDP) on behalf of the Indonesian Ministry of Higher Education, Science and Technology and managed under the EQUITY Program (Contract No 4299/B3/DT.03.08/2025 & No 3029/PKS/ITS/2025); the Directorate General of Higher Education, Research and Technology of the Republic of Indonesia under Penelitian Pendidikan Magister menuju Doktor untuk Sarjana Unggul (PMDSU) (Contract No 1163/PKS/ITS/2025); Institut Teknologi Sepuluh Nopember under Penelitian Departemen (Contract No 2369/PKS/ITS/2025) and Program Riset Penugasan ITS Strategic Research Grant (SRG); Ethereum Foundation under Academic Grants Round 2025 (Grant No FY25-2129); HETI ADB ITS under Penelitian Post-Doctoral (Contract No. 0018/01.BAP/PPK-HETI/ITS/2025); and National Research and Innovation Agency (BRIN) under RIIM Kompetisi Gelombang 10 Program.

Appendix A. Supplementary data

Supplementary data for this article can be found online at doi:10.1016/j.softx.2026.102548.

References

[1] Skouti T, Seiger R, Furrer FJ, Strahringer S. RBPMN: the value of roles for business process modeling. Softw Syst Model 2024;23(6):1375–406. https://doi.org/10.1007/s10270-024-01202-z
[2] Compagnucci I, Corradini F, Fornari F, Re B. A study on the usage of the BPMN notation for designing process collaboration, choreography, and conversation models. Bus Inf Syst Eng 2024;66(1):43–66. https://doi.org/10.1007/s12599-023-00818-7
[3] Lopes T, Guerreiro S. Assessing business process models: a literature review on techniques for BPMN testing and formal verification. Bus Process Manag J 2023;29(8):133–62. https://doi.org/10.1108/BPMJ-11-2022-0557
[4] Belchior R, Guerreiro S, Vasconcelos A, Correia M. A survey on business process view integration: past, present and future applications to blockchain. Bus Process Manag J 2022;28(3):713–39.
[5] De Leoni M, Felli P, Montali M. Integrating BPMN and DMN: modeling and analysis. J Data Semantics 2021;10(1):165–88.
[6] Beerepoot I, Di Ciccio C, Reijers HA, Rinderle-Ma S, Bandara W, Burattin A, Calvanese D, Chen T, Cohen I, Depaire B, et al. The biggest business process management problems to solve before we die. Comput. Ind 2023;146:103837.
[7] Zimmermann L, Zerbato F, Weber B. Process mining challenges perceived by analysts: an interview study. In: International conference on business process modeling, development and support. Springer; 2022. p. 3–17.
[8] Compagnucci I, Corradini F, Fornari F, Re B, et al. BPMN inspector: a tool for extracting features from BPMN models. In: BPM (demos/resources forum); 2023. p. 122–6.
[9] Corradini F, Fornari F, Polini A, Re B, Tiezzi F, Vandin A. A formal approach for the analysis of BPMN collaboration models. J Syst Softw 2021;180:111007.
[10] Fahland D. Process mining over multiple behavioral dimensions with event knowledge graphs. In: Process mining handbook. Springer; 2022. pp. 274–319.
[11] Choudhary R, Riaz N. A business process re-engineering approach to transform business process simulation to BPMN model. Plos One 2023;18(3):e0277217.
[12] Scheepens RJ, Teeuwen RMA. Process mining for multi-instance processes, 2022 uS Patent App. 17/119, 987. https://patentimages.storage.googleapis.com/da/ab/a1/49f1e9cbcf2f88/US20220188143A1.pdf [16 June 2022].
[13] Akhramovich K, Serral E, Cetina C. A systematic literature review on the application of process mining to industry 4.0. Knowl Inf Syst 2024;66(5):2699–746.
[14] Carmona J, van Dongen B, Weidlich M. Conformance checking: foundations, milestones and challenges. In: Process mining handbook. Springer; 2022. pp. 155–90.
[15] Sholiq S, Sarno R, Astuti ES. Generating BPMN diagram from textual requirements. J King Saud Univ Comput Inf Sci 2022;34(10):10079–93.
[16] Kräuter T, Rutle A, König H, Lamo Y. Formalization and analysis of BPMN using graph transformation systems. In: International conference on graph transformation. Springer; 2023. p. 204–22.
[17] Kräuter T, Rutle A, König H, Lamo Y. A higher-order transformation approach to the formalization and analysis of bpmn using graph transformation systems. Log Methods Comput Sci 2024;20.

[18] Sungkono KR, Sarno R, Salsabila MC, Dewi CP. A graph-based method for merging business process models by considering semantic similarity. Int J Intell Eng Syst 2023;16(2).
[19] Sungkono KR, Sarno R, Onggo BS, Septiyanto AF. Optimising waste management collaboration processes using hybrid modelling. Int J Simul Model 2024;23(1):53–64.
[20] Sungkono KR, Sarno R, Onggo BS, Haykal MF. Enhancing model quality and scalability for mining business processes with invisible tasks in non-free choice. J King Saud Univ Comput Inf Sci 2023;35(9):101741.
[21] Sarno R, Sungkono KR, Taufiqulsa'di M, Darmawan H, Fahmi A, Triyana K. Improving efficiency for discovering business processes containing invisible tasks in non-free choice. J. Big Data 2021;8(1):113.
[22] Heinze TS, Stefanko V, Amme W. Mining bpmn processes on Github for tool validation and development. In: International conference on business process modeling, development and support. Springer; 2020. p. 193–208.
[23] Di Martino B, Colucci Cante L, Esposito A, Graziano M. A tool for the semantic annotation, validation and optimization of business process models. Softw Pract Exp 2023;53(5):1174–95.
[24] van der Aalst WMP, Berti A. Discovering object-centric petri nets. Fundam Inform 2020;175(1–4):1–40.
[25] Camargo M, Dumas M, González-Rojas O. Automated discovery of business process simulation models from event logs. Decis Support Syst 2020;134:113284.
[26] Corradini F, Muzi C, Re B, Rossi L, Tiezzi F, et al. Mida: multiple instances and data animator. In: CEUR workshop proceedings, vol. 2196. CEUR-WS; 2018. p. 86–90.
[27] Corradini F, Mozzoni L, Piccioni J, Re B, Rossi L, Tiezzi F, et al. Bear 2.0: enhancing the environment model for animating environment-aware BPMN collaborations. In: Joint proceedings of the best dissertation award, doctoral consortium, and demonstration & resources forum at BPM 2025, vol. 4032. CEUR-WS. Org; 2025. p. 0.
[28] Maslov I, Poelmans S. Facilitating the comprehension of business process models for unexperienced modelers using token-based animations. Inf Manag 2024;61(5):103967.
[29] Allweyer T, Schweitzer S. A tool for animating BPMN token flow. In: International workshop on business process modeling notation. Springer; 2012. p. 98–106.
[30] Bazan P, Estevez E. Industry 4.0 and business process management: state of the art and new challenges. Bus Process Manag J 2022;28(1):62–80.
[31] Corradini F, Pettinari S, Re B, Rossi L, Tiezzi F. A technique for discovering bpmn collaboration diagrams. Softw Syst Model 2024;23(6):1323–43.
[32] Merkoureas I, Kaouni A, Theodoropoulou G, Bousdekis A, Voulodimos A, Miaoulis G. Smyrida: a web application for process mining and interactive visualization. SoftwareX 2023;22:101327.
[33] Pawlak TP, Potoniec J. Processm: intelligent process mining software. Available at SSRN 5036465 2025.
[34] Chapela-Campa D, López-Pintado O, Suvorau I, Dumas M. Simod: automated discovery of business process simulation models. SoftwareX 2025;30:102157.
[35] Avila DT, dos Santos RI, Mendling J, Thom LH. A systematic literature review of process modeling guidelines and their empirical support. Bus Process Manag J 2021;27(1):1–23.
[36] Rullo A, Alam F, Serra E. Trace encoding techniques for multi-perspective process mining: a comparative study. Wiley Interdiscip Rev Data Min Knowl Discov 2025;15(1):e1573.
[37] Nolte FR. Text to process model: automating process model creation from text. Westfaelische Wilhelms-Universitaet Muenster (Germany); 2021.
[38] Tskhadadze R. Unraveling control-flow structures for predictive process monitoring: analyzing the impact of capturing and encoding control-flow information for process outcome prediction [Master's thesis], San Diego State University; 2024.