Journal of
Information Systems Engineering
and Business Intelligence

Vol.12, No.1, February 2026
Available online at: http://e-journal.unair.ac.id/index.php/JISEBI

Repolink: A Repository Driven Technique for Reconstructing Missing Links in Business Process Model

Kristina [1], Ary Mazharuddin Shiddiqi [2]*, Daniel Siahaan [3], Adrian Forca [4]

[1)2)3)] *Sepuluh Nopember Institute of Technology, Surabaya, Indonesia*
[1)] 7025221016@student.its.ac.id, [2)] ary.shiddiqi@if.its.ac.id, [3)] daniel@if.its.ac.id

[1)] *Widya Dharma Pontianak University, Pontianak, Indonesia*
[1)] 7025221016@student.its.ac.id

[4)] *College of Science and Technology-Guimaras State University, Jordan, Philippnes*
[4)] *adrian.forca@gsc.edu.ph*

Abstract

Background: The development of modern organization emphasizes the importance of accurate and comprehensive business process models (BPMs). BPMs serves to provide clear work standards for business actors. Business Process Model and Notation (BPMN) is widely used to model and analyse business processes. However, BPM models in practice often contain missing or inconsistent control-flow links, which reduce model correctness and limit effective analysis. Existing BPM retrieval approaches mainly focus on similarity measurement and provide limited support for explicit missing-link reconstruction.
Objective: This study aims to propose a repository-driven approach to detect and reconstruct missing control-flow links in BPMN models while preserving computational efficiency and explainability.
Methods: This study employs a quantitative experimental methodology on the use of an application called *Repolink*., a graph-based technique that transforms BPMN models into directed graphs and computes structural similarity values using Graph Edit Distance combined with semantic weighting. A query BPMN model is compared against a repository of reference BPMN models to identify structural inconsistencies. Missing links are detected using adjacency comparison supported by forward and reverse mappings.
Results: The results show that *Repolink* can detect and reconstruct missing control-flow links in various BPMN structures, including branching and loop-related patterns. It is also able to significantly generate efficient retrieval with an overall time complexity of $O(B * A^2)$, where A is the number of nodes and B is the number of repository models. Compared to existing methods, *Repolink* provides higher explainability by explicitly reporting missing edges.
Conclusion: *Repolink* effectively supports missing-link reconstruction in BPMN models through a repository-driven and explainable approaches. While the method focuses on structural analysis rather than full behavioural semantics, it offers a practical solution for BPMN conformance checking and model debugging.

Keywords: Information Retrieval, Diagram Similarity, Structural Semantic, Graph Edit Distance, Greedy Algorithm

Article history: Received 1 November 2025, first decision 30 December 2025, accepted 23 January 2026, available online 28 October 20XX

I. INTRODUCTION

The use of natural language processing has enabled the automation and digitalization of business processes in modern organizations [1], [2] . Automatically generated business process model is commonly modelled using Business Process Model and Notation (BPMN), which provides a standardized and expressive framework for representing organizational workflows [3]. BPMN diagrams visually describe sequences of activities, decision points, and interactions designed to achieve specific organizational goals. Through clear representation of operational flows, resources, and milestones, BPMN supports process transparency, efficiency improvement, quality assurance, and collaboration across organizational units. Consequently, accurate identification and modelling of business processes play a critical role in reducing operational errors and improving organizational performance.

In practice, organizational activities are often composed of multiple business sub-processes documented in both formal and informal forms. Formal documentation, such as standard operating procedures (SOPs), is typically structured and well defined, serving as a reliable reference for operational execution. However, many emerging or small

* Corresponding author

ISSN 2443-2555 (online) 2598-6333 (print)
doi: http://dx.doi.org/10.20473/jisebi.12.1.98-114

organizations lack comprehensive SOPs and instead rely on informal and unstructured documents. To address this limitation, the Informal Document to Semantic Business Vocabulary and Business Rules (ID2SBVR) method was introduced to translate informal textual documents into BPMN models [4]. While ID2SBVR facilitates BPMN modelling in the absence of formal documentation, it remains vulnerable to incomplete or ambiguous information, which can lead to missing process steps or control-flow links, particularly at decision points or gateways. As a result, BPM models generated from informal documents may become incomplete and unsuitable for further analysis or system implementation.

To mitigate this issue, incomplete BPM models must be systematically completed. One promising approach is to leverage information retrieval techniques using large-scale business process repositories [5], [6]. Such repositories enable organizations to retrieve, compare, and reuse existing process models as references for process completion, redesign, and optimization [7], [8], [9]. Repository-based BPM analysis is also crucial in organizational mergers, where multiple process models must be evaluated for compatibility and integration. Consequently, research on BPM repositories has attracted increasing attention due to its potential to improve modelling efficiency and process quality.

Existing studies on BPM repositories primarily focus on business process model similarity, aiming to identify recurring workflow patterns across organizations with similar functional objectives [10], [11], [12].These approaches explore structural, behavioural, and semantic similarities to support process reuse, redesign, and error reduction. In addition, similarity analysis can assist in detecting missing components and enable automated completion of BPMN models. Several studies have employed graph-based techniques to measure similarity by exploiting lexical information from nodes and edges, as demonstrated in UML Class Graph (UCG) analysis. Prior work has shown that inter-structural and intra-structural graph representations effectively capture relationships and attributes within diagrams [13], [14]. Graph Edit Distance (GED) and greedy algorithms have been widely adopted to compute similarity by minimizing transformation costs between graph elements, thereby improving computational efficiency by pruning mismatched candidates early.

Graph matching has been extensively applied in computer vision and pattern recognition for diagram comparison [15], [16], [17] , and BPMN models, which contain rich metadata, are well suited for graph-based analysis. A graph consists of vertices and edges, enabling explicit modelling of structural relationships. Previous studies have demonstrated that inter-structural and intra-structural graph approaches are effective for assessing structural similarity in UML and BPM (Business Process Model) diagrams [18], [19], [20]. These methods convert diagram metadata into graph nodes and edges, allowing systematic comparison and facilitating the detection of missing information. However, structural similarity alone is often insufficient to fully capture process semantics.

To improve accuracy, semantic similarity can be incorporated using lexical information embedded in BPMN metadata, which is stored in Extensible Markup Language Process Definition Language (XPDL) format. BPMN metadata includes pools, lanes, activities, gateways, data objects, transitions, associations, and message flows. Semantic similarity can be computed using natural language processing techniques such as cosine similarity and pairwise comparison for weighted evaluation [21]. This research method utilizes graph matching techniques and Natural Language Processing (NLP) to identify and reconstruct missing business processes and flows of BPM. The repository is used to handle missing links in a BPM. Retrieval of information in a repository requires considering aspects such as search speed, user relevance, and process complexity, necessitating the use of intelligent query techniques [22], [23], [24], [25]. Although prior research has demonstrated promising results in BPM similarity retrieval, most existing methods focus on direct comparisons between complete BPM models and do not adequately address scenarios involving incomplete models.

This study addresses this gap by proposing *Repolink*, a repository-driven approach that integrates graph matching and natural language processing to reconstruct missing links in BPMN models. *Repolink* transforms BPM models into inter-structural and intra-structural graph representations and gradually compares each node in an incomplete BPM model against nodes in multiple repository models. By combining structural similarity, semantic similarity, and missing-link detection, *Repolink* identifies the most appropriate candidates for reconstructing missing control-flow links. The resulting completed BPM model can serve as a reliable specification for system development and auditing purposes. This approach is particularly beneficial for organizations that lack formal documentation, enabling them to derive complete and explainable BPM models from informal sources.

II. METHODS

A. Repolink Information Retrieval System

The architecture of the proposed method is illustrated in Fig.1, which outlines a BPMN missing links diagram generated from informal document translation. Based on its metadata type, the relationships between metadata are sorted, and the metadata is transformed into inter structural and intra structural graph. Repolink works by detecting

missing links from the graph and comparing them with the graph in the BPMN diagram repository. Replacement nodes are selected from the BPM graph based on the highest values of structural and semantic similarity. *RepoLink* method employs a graph matching approach, utilizing structural and semantic similarity measurements and NLP to reconstruct missing nodes and links in BPM.

The five primary functions in the *Repolink* method are parsing BPM metadata from XPDL, sorting metadata, converting it to inter-structural and intra-structural graphs, detecting missing links, constructing missing links, calculating the GED cost per node, and comparing the overall BPM to identify *MissingNodes* and *MatchingNodes*.

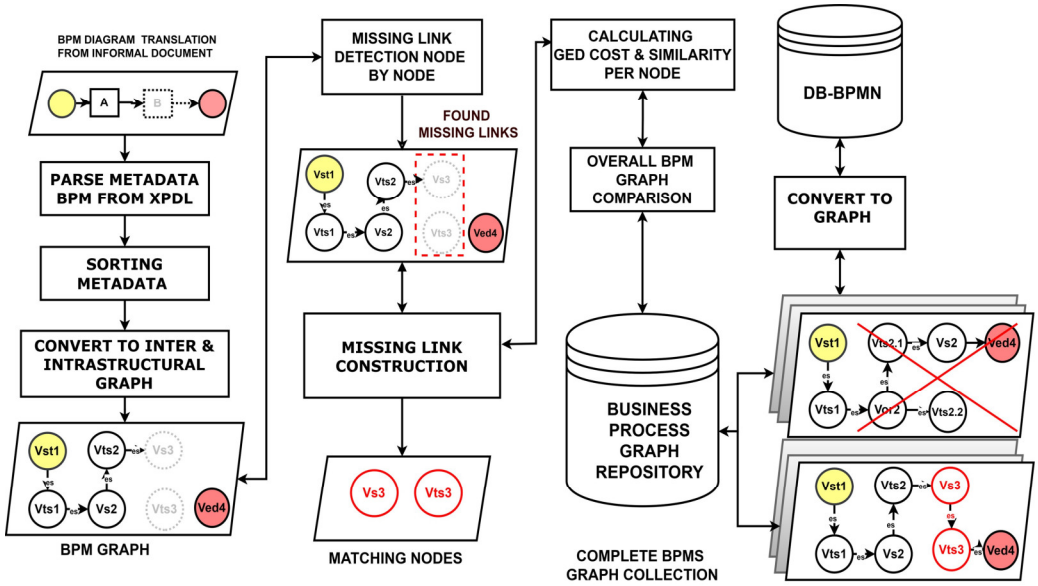

Fig. 1 *Repolink* information retrieval method architecture. *Repolink* is a repository driven technique to find BPM missing links using graph and NLP approaches in BPM repositories.

B. Data Collection

This research was conducted from September 2022 to March 2025. The BPM datasets used in this study were constructed based on interviews with business actors directly involved in operational processes within the selected units, including librarians, hotel operational staff, and laundry service personnel. The interviews were conducted over multiple sessions during the data collection phase of this study. A semi-structured interview guideline was used to capture process activities, decision points, control-flow relationships, and exceptions. There was two main of interview questions such as: (1) How many departments or sections are there in this unit? (2) Can you explain the process carried out in this unit? The information obtained from these interviews served as the primary input for developing BPMN models, which were subsequently formalized and validated by the researchers. In total, 187 BPMN models were developed based on the consolidated interview results and were used as the evaluation dataset for the proposed method.

The datasets used in this study are divided into two groups: testing dataset and repository dataset. All BPMs stored in the repository are complete and have undergone structural verification. The BPMs were populated with the identified metadata, including pool, lane, activity (Act), gateway (Gate), data object (DO), transition (Trans), association (Assoc), and message flow (MsgF), as shown in Table 1.

For the repository datasets, we used 302 BPMs with 9.326 metadata from 42 organizational domains for repositories. The repository dataset of 48 BPMs of library was taken from previous studies [4], 67 BPMs from another study [26], and 187 BPMs which were developed by the researchers. The repository used in this study contains BPM files formatted in XPDL. The resulting graph repository is the foundation for identifying and retrieving missing processes or flows in incomplete BPMs. These files were converted into graph representations consisting of inter-structural and intra-structural elements. Each diagram was completed by retrieving a similar node from the BPM repository using the proposed *RepoLink* method.

The testing dataset used a primary dataset constructed by the researchers from three different types of organizations such as library, hotel and laundry. We used 32 BPM with 725 metadata from 3 organizational domains as shown in

Table 2. The library, hotel, and laundry units were selected as evaluation cases because they represent distinct business domains with different process characteristics. The library unit primarily involves sequential and rule-based processes, the hotel unit contains more complex workflows with branching and parallel activities, and the laundry unit includes repetitive and loop-based operations. This diversity allows *Repolink* to be evaluated across varying BPMN structures and levels of complexity, demonstrating its applicability to different real-world business scenarios.

TABLE 1
REPOSITORY DATASET

Organization type	Number of organization	BPM	Pool	Lane	Act	Gate	Do	Trans	Assoc	MsgF	Total
Library	2	48	52	93	504	78	10	566	2	0	1.305
Primary school	3	15	55	61	223	0	18	217	23	0	597
Credit Union	6	57	70	125	556	101	75	659	87	2	1.675
Hospital/ Pharmacy	9	53	117	121	616	80	163	683	110	3	1.893
Retail	6	50	100	144	610	48	54	659	58	0	1.673
Bakery	1	10	25	23	131	0	15	132	12	0	338
Laundry	1	5	10	12	54	0	0	18	0	0	94
Online Transportation	1	5	10	13	59	0	1	66	1	0	150
Restaurant	8	11	23	39	157	0	0	158	1	0	378
Various shop	9	48	113	145	476	0	31	429	29	0	1.223
Total	42	302	575	776	3.386	307	367	3.587	323	5	9.326

TABLE 2
TEST CASES

Units	BPM	Pool	Lane	Act	Gate	Do	Trans	Assoc	MsgF	Total
Library	14	16	24	127	19	2	152	2	0	356
Hotel	10	10	21	81	8	0	79	0	0	209
Laundry	8	8	14	56	3	0	71	0	0	160
Total	32	34	59	264	30	2	302	2	0	725

C. Data Pre-processing

Preprocessing diagrams help gather lexical and structural information. BPM diagrams created using BPMN in XPDL format were analyzed and found to contain all the structured information of a diagram. The BPM metadata is categorized into property information and relationship information [21]. The property information includes elements such as pools, lanes, activities, gateways, and data objects, while the relationship information encompasses the connections among these elements. In this case, the parsing of properties and relationships of BPM diagrams in XPDL was carried out using the *xml. etree.ElementTree as ET* library of Python.

After all metadata has been parsed, it was sorted using Kahn's algorithm. Sorting nodes was performed so that each activity can be executed in the correct order according to the BPMN workflows. Metadata sorting was carried out by first transforming the metadata into an accurate graph representation, enabling compatibility with graph-based search methods. The graph is traversed using the BFS algorithm, and the generated nodes were then topologically sorted using Kahn's algorithm. This step is purported to mitigate inconsistencies in the 'from' and 'to' relationships associated with transition nodes. Furthermore, variations in source and destination nodes were incorporated as determining factors in the computation of GED costs at the node level.

Once all the metadata was in the correct order after sorting, it was converted into inter-structural and intra-structural graphs excepts pool and lanes. Inter-structural and intra-structural graphs measure the structural similarity between the two BPMs. The inter-structural graph was divided into state nodes and transition nodes. State nodes are BPM diagram metadata, including pools, lanes, activities, and data objects. Transition nodes are BPM diagram metadata, such as sequence, association, and message flows. All metadata types, except for pools and lanes, which were converted into graph nodes. Pools and lanes were not used for structural similarity measurements, but they were used for semantic similarity measurements. This transformation involved labeling each state and transition in the BPM diagram, following metadata labeling rules described in previous studies [20].

D. Detecting Missing links of a BPM graph

A missing link occurred when an interviewee failed to mention one or more business processes or flows that are part of a BPM. To function effectively, a BPM must represent a complete process, including its input and output components. In this study, missing processes or flows were identified by analyzing the connectivity between nodes in the BPM inter-structural graph and the node type from the intra-structural graph. If any node lacked connectivity, the

BPM graph was then classified as having a missing link. Each node was analyzed based on its in-degree and out-degree, as defined in (1) as follows.

$$degree(v) \quad = InDegree(v) + OutDegree(v) \qquad (1)$$

Each node has a minimum of one *InDegree* and one *OutDegree*, except for the start and end nodes. Suppose the degree is less than two, then missing links are found based on (1). The BFS algorithm is used to locate matching nodes. This algorithm traverses the graph to find missing transitions and aligns them with potential matches, as formalized in (2).

$$MissingNodes \; = \{v \in G | \, Degree \, (v)\} \qquad (2)$$

The degree is the number of edges connected to a vertex (v). *MissingNodes* is the vertices in *Graph G* with less than or equal to one degree. *InDegree* is the number of incoming edges, and *OutDegree* is the number of outgoing edges. Once a missing link was identified, the next step was to retrieve a similar node from the BPM repository using the *RepoLink* method. Algorithm 1 calculated the number of degrees, in and out, of a node to detect complete or incomplete nodes. If it is incomplete, it means that BPM has missing links.

Algorithm 1
Missing links detection

```
INPUT: Graph G
FOR each node in G:
    count how many times node appears in edges
missing nodes = all nodes with count < 2 AND node not Start/End
RETURN missing nodes
```

E. Constructing Missing links

The retrieval of missing links in BPMs was based on measuring structural and semantic similarities using inter-structural and intra-structural graph representations. Inter-structural graphs help evaluate node connectivity (e.g., flows between activities or gateways). In contrast, intra-structural graphs focus on the internal properties of individual nodes, such as activity types or associated data. The search process for matching nodes in an incomplete BPM graph follows these core rules: each BPM graph must have a *Start* and an *End* event. Every node must have at least one input (*InDegree*) and one output (*OutDegree*), except the *Start* event and *End* event. In this study, search is performed sequentially from the *Start* node to the *End* node. BPMs were transformed into inter-structural and intra-structural graphs. Inter-structural graphs were alternated between state and transition nodes. Considering that the number of nodes in the incomplete BPM graph (*G1*) must be less than or equal to the number of modes complete BPM graph (*G2*). A cost matrix is constructed incrementally, with dimensions (m + n), where m and n represent the number of nodes in *G1* and *G2* respectively.

Semantic similarity used to complement this approach by analyzing the meaning behind node labels, enabling the identification of functionally equivalent elements that may have different names or descriptions. Similarity values were then computed to determine the best-matching nodes. These were derived using a combination of cosine similarity for semantic content and GED for structural comparison, as shown in (3) as follows.

$$SimVal = \; (0.5 * StrucVal) + \; (0.5 * SemVal) \qquad (3)$$

SimVal is the total similarity score, *StrucVal* is the structural similarity value, and *SemVal* is the semantic similarity value. Each node's structural and semantic similarity value determined was 0.5, based on the chance value of finding a replacement node candidate from the BPM repository. The comparison involved calculating each node's structural and semantic similarity values from the start node to the end node. Regarding this calculation, if a graph does not meet the required similarity thresholds (*SimVal* more than 50%), it will be eliminated from the search. The node-matching process was carried out using structural and semantic similarity assessments and is optimized for computational speed to locate nodes or edges similar to the missing elements. The method of missing link reconstruction in an inter-structural graph was carried out by comparing each node of the missing link graph with each node of all complete graphs in the repository as shown in Algorithm 2 as follows.

To support the detection of efficient graph traversal and missing edge, *Repolink* builds two additional data structures called forward-map and reverse-map. The forward-map stores the outgoing control flow relation for each BPMN node, mapping each node to its directly reachable successors. Meanwhile, the reverse-map stores the incoming control flow relation, which maps each node to its predecessors. These mappings are constructed during XML parsing by iterating over the sequence flow and updating the adjacency relations accordingly. During similarity computation, the forward-map facilitates the identification of forward structural paths, while the reverse-map enables backtracking and validation of incoming dependencies. Thus, both structures enable *Repolink* to efficiently detect missing or inconsistent links by comparing the expected and actual adjacency relations between BPMN models.

Algorithm 2
Missing links construction

```
INPUT: missing_nodes, forward_map, reverse_map
FOR each node in missing_nodes:
  IF reverse_map[node] exists:
    add link (reverse_map[node][0] → node)
  IF forward_map[node] exists:
    add link (node → forward_map[node][0])
RETURN reconstructed_links
```

F. Calculating GED cost and similarity per node

The GED is a computational method for measuring the structural similarity between two graphs by calculating the number and cost of the minimum editing operations required to convert one graph into another. Meanwhile, the greedy algorithm determines the optimal value of the costs resulting from GED editing operations. As for word occurrence analysis and pairwise comparison for weighted evaluation, semantic similarity between nodes was calculated using techniques such as Cosine Similarity.

Algorithm 3
GED cost calculation and similarity per node

```
INPUT: nodeA, nodeB
IF nodeA or nodeB missing → cost = 1
ELSE IF types differ → cost = 2
ELSE IF labels differ → cost = 1
ELSE cost = 0

IF cost = 0 → sim = 1.0
ELSE IF cost = 1 → sim = 0.5
ELSE sim = 0.0

RETURN cost, sim
```

Algorithm 3 shows that the GED algorithm determines a value of 1 for each operation (insertions, deletions, substitutions) performed when converting one graph to another. The formation of GED costs for reconstructing missing links has been adjusted by gradually calculating the cost per node. The results of the GED cost calculation per node were used to determine the two compared nodes' structural similarity (inter-structural and intra-structural). Graph similarity was measured using the GED Greedy algorithm, which calculated the minimum set of operations (insertions, deletions, substitutions) required to transform one graph into another. For example, *nodeA* and *nodeB* represent the graphs being compared, each consisting of a set of vertices. Each operation of vertices transformation in *nodeA* was adjusted to equal the vertices in *nodeB*. And each operation costs one which is recorded in the cost matrix.

G. Comparing overall BPMs in the repository

Finding replacement nodes in the repository graph, BPM was performed by comparing all the missing link graph nodes with all the graph BPMs in the repository. In this case, if a node found did not meet the threshold value, then all nodes on the graph after that node were eliminated. While, the cost matrix was generated gradually to find the matching nodes. The dimensions of the cost matrix depend on the number of nodes in the incomplete BPM graph and the candidate graph in the repository, which indicates the difference in transformation costs. The GED algorithm

calculates total transformation costs, while the greedy strategy minimizes the required operations [20]]. If similarity values of nodes fail to meet the thresholds value, the whole graph is eliminated at once to avoid unnecessary computation. Algorithm 4 compares the missing BPM link graph with all other complete graphs in the repository by calculating the total similarity value between the graphs and recommends the node from the graph with the highest similarity value.

Algorithm 4
Comparison of overall BPM

```
INPUT: target_graph, other_graphs
missing_in_target = detect_missing_nodes(target_graph)

FOR each graph G in other_graphs:
   total_sim = 0
   FOR each pair of aligned nodes:
      IF target_node missing:
         try match with any next node in G
      compute GED similarity
      accumulate similarity
   compute average similarity
select graph with highest similarity
RETURN best_graph, similarity_score
```

A. Evaluation Methods

The evaluation of the results and analyses was conducted using two approaches: Gwet's AC2 and mathematical efficiency analysis. Gwet's AC2 was employed to measure the level of agreement among experts regarding the utilization and outcomes of the Repolink method [27]. Each expert was asked to assess the degree of structural and semantic similarity between BPMN diagrams using a numerical scale ranging from 0% to 100%.

To further evaluate the accuracy of expert assessments relative to the similarity values produced by the Repolink method, the Mean Relative Error (MRE) was calculated [28]. The MRE quantifies the deviation between expert-provided similarity scores and the similarity scores generated by Repolink, as defined in (4) and (5).

$$\mathrm{RE_i} = |(\mathrm{A_i} - \mathrm{P_i}) / \mathrm{A_i}| \quad (4)$$

$$\mathrm{MRE} = \frac{1}{\mathrm{n}}\sum_{\mathrm{i-1}}^{\mathrm{n}} \left|\frac{\mathrm{A_i} - \mathrm{P_i}}{\mathrm{A_i}}\right| \quad (5)$$

The relative error (RE) represents the error of the *i*-th data point, where $A_i$denotes the actual value and P_i denotes the predicted value. Expert assessment scores were grouped into four evaluation categories based on their similarity ratings: Strongly Disagree (<25%), Disagree (25%–<50%), Agree (50%–<75%), and Strongly Agree (>=75%). The expert assessment data are treated as an ordinal scale and evaluated using weighted agreement. The accuracy of the similarity measurement produced by Repolink was evaluated using the Mean Relative Error (MRE). If the MRE value is less than 10% (regression metrics), the similarity measurement method employed by Repolink is considered to perform well [29]. When this condition is satisfied, the expert assessment results are further evaluated for inter-rater agreement using Gwet's AC2, as defined in (6).

$$\mathrm{AC2} = \frac{\mathrm{A_0} - \mathrm{A_e}}{1 - \mathrm{A_e}} \quad (6)$$

In Gwet's AC2 formulation, A_0represents the observed agreement, while $A_e$denotes the chance agreement. The interpretation of AC2 values follows established agreement categories, namely less than chance agreement (<0), slight agreement (0.01–0.20), fair agreement (0.21–0.40), moderate agreement (0.41–0.60), substantial agreement (0.61–0.80), and almost perfect agreement (0.81–1.00)[30].

Big-O notation was employed to evaluate the time complexity of the proposed method (mathematical efficiency analysis. Big-O notation provides a mathematical framework for describing the upper bound of an algorithm's running time or space requirements as a function of input size [31]. This notation characterizes worst-case behavior and enables

researchers to understand how an algorithm scales as data volume increases. In the context of BPM repositories that may contain hundreds or even thousands of process models, scalability analysis is essential for assessing the practical applicability of the proposed approach.

III. RESULTS

A. The Result of Applying the Repolink Method to a Missing Link Case

To simulate realistic operational disruptions, the test dataset included various missing elements, such as a single missing node, multiple missing nodes sequentially or non-sequentially, missing gateways, and missing data objects. In this study, we investigated a case involving the loss of a single transition in the BPM diagram to demonstrate how the *Repolink* method works. This case corresponds to the first scenario (Case ID 01) listed in Table 3, namely Equipment Recap and Periodic Issue. Fig. 2 shows a BPMN diagram, "Equipment Recapp" that has missing links at the activities "put a stamp on each tool" and "input data to system".

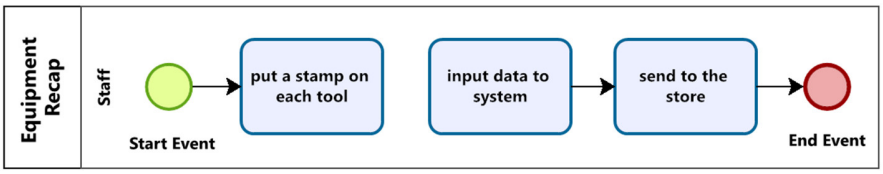

Fig 2. Missing Links BPMN diagram translated from informal document

The diagram above explains how the *Repolink* method works in Equipment Recap Laundry BPM. This BPM represents a business process in a laundry service, including stamping equipment, entering equipment data into a database, and dispatching the equipment to a storage area. Before identifying the missing nodes, the BPMN file was converted into an XPDL file. The metadata extracted from the XPDL file corresponding to the Equipment Recap Laundry BPM Diagram serves as pools: *Equipment Recap; l*ane: s*taff; a*ctivities: s*tart event, put stamp on each tool, input data to system, send to the store, end event.* There are three transitions extracted, namely: ***"from"*** *Start event* ***"to"*** *Put a stamp on each tool;* ***"from"*** *Input data to System* ***"to"*** *Send to the store; and* ***"from"*** *Send to the store* ***"to"*** *End event.* As shown in Fig. 2, the transition ***"from"*** *Put a stamp on each tool* ***"to"*** *Input data to System* is missing.

Before identifying the missing links, the BPMN file was converted into an XPDL file. Likewise, the metadata BPM was extracted from XPDL file. Each metadata was converted into two types of graphs using label, namely the inter-structural and intra-structural graph [21], which are shown in Fig. 3a and Fig. 3b.

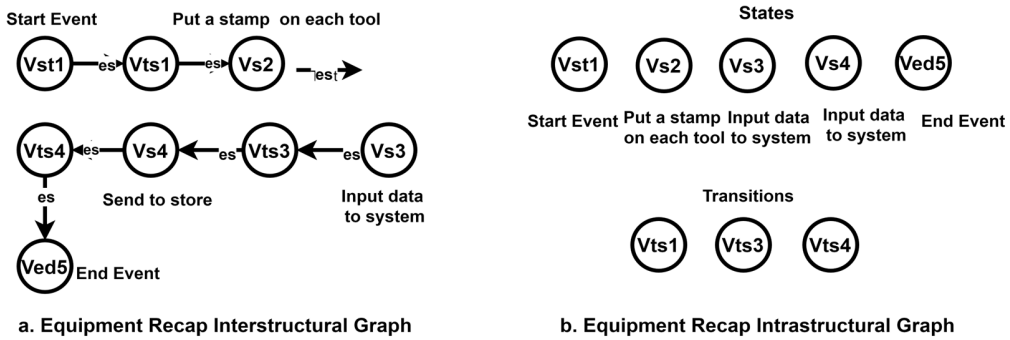

Fig. 3 Interstructural and intrastructural graph of Equipment Recap Laundry BPM metadata translation results.

Fig. 3a shows the inter-structural graph when node Vst_1 represents the *Start Event*. Vst_1 and is connected to Vs_2 (through Vts_1 (*transition node).* The missing node can be identified by examining graph connectivity and the number of Degrees using (2). The inter-structural graph is composed of state and transition arrangements. State and transition attributes were identified from intra-structural graphs as shown in Fig. 3b. Meanwhile, the intra-structural graph was used to examine the internal attributes of the nodes supporting the identification of missing nodes during repository matching. Missing nodes can be identified if a node state has a degree of less than two based on (1), except for the *Start* and E*nd* nodes. Fig. 4 shows that *Vs2* only has one *InDegree* (Vts_1, a transition node) and zero *OutDegree (no*

transition nodes found; Vs_3 is a state node). The relationships between nodes in the graph consists of state nodes alternating with transition nodes can be seen Fig. 3a (interstructural graph). Vs_2 and Vs_3 are both state nodes. The nodes type can be seen in Fig.3b (intrastructural graph). If a state node is found after the previous node is also a state node, it indicates that a missing link exists in the graph. The missing link is cleary a transition node between Vs_2 and Vs_3, although more states nodes may be missing if multiple nodes are missing.

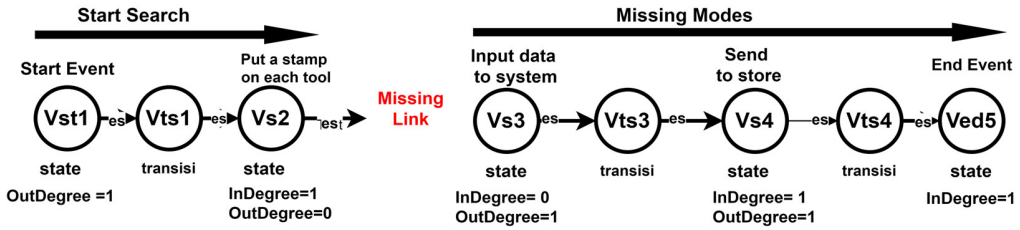

Fig. 4 The process of detecting missing edges in a BPM diagram. Detecting missing links on an inter-structural graph will be done by calculating the number of degrees from the node. The structure of the inter-structural graph is governed by the alternation of state and transition nodes.

Once the missing links in the BPM were successfully detected, the missing links' search process was carried out by comparing the missing link in the BPM with the BPMs in the repository. The inter-structural and intra-structural graphs of the incomplete BPM are compared with graphs in the complete BPM repository. The comparison process was carried out by gradually calculating the similar values between nodes, shown in Fig. 5.

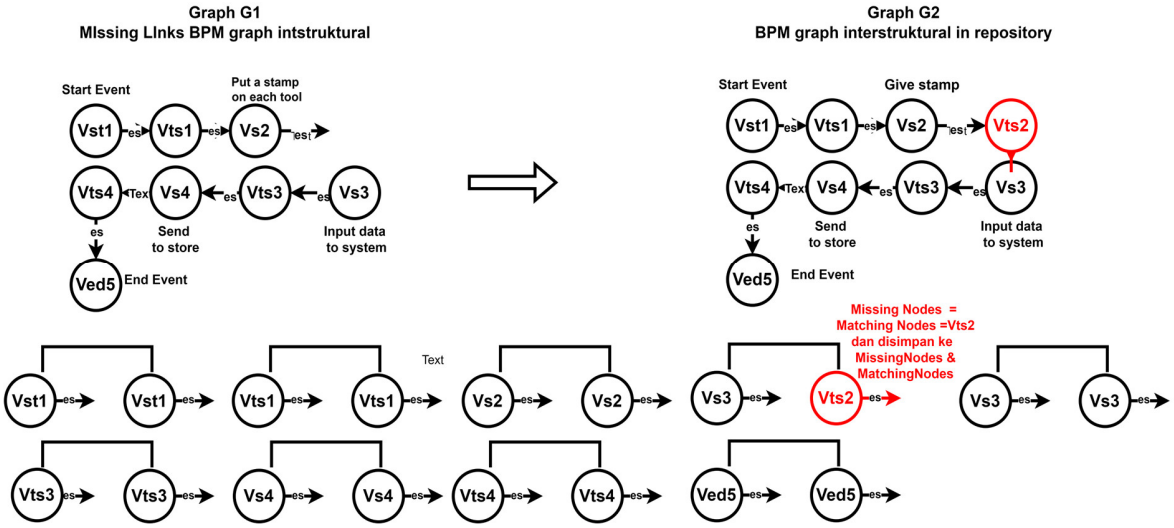

Fig. 5 The process to retrieve missing one node in the repository. Graph G1 is an inter-structural graph of Equipment Recap Laundry BPM. Graph G2 is an inter-structural graph of the BPM present in the repository. The search for *MatchingNodes* on the repository graph is done in stages by calculating each node's structural and semantic similarity compared between the graphs.

Table 4 summarizes the gradual calculation of GED cost and similarity values node-by-node between graphs. The reference values for the matrix cost are 0, 1 and greater than 1. If the cost of two vertices is 0, then they are identical, and the search can be continued. Similarly, if the cost of two vertices is 1, they have a similarity value of more than 50%, so the search can be also continued. However, if the calculated cost of two vertices is greater than 1, the search can still be continued provided it is still in the missing link condition, and the graph is not eliminated. On the other hand, if it is not in the missing link condition, the search will be stopped, and its graph will be eliminated.

The minimum total similarity value that must be met for the search to continue was 50%. The Vst_1 is the Start node, so the Vst_1 in G_1 was directly compared to the similarity to Vst_1' in G_2. The cost of the two nodes was 0.00. A value of 0.00 means that both nodes were the same type (state) and name (Start). The structural similarity value of the two nodes was 1.00, and the semantic similarity value was one. Based on (3), the total similarity value of the two vertices was one. The minimum total similarity value that must be met for the search to continue was 50%. The similarity

value of the two nodes was 100%, and it met the minimum value criteria. So, it means that the search can continue to the next node, the Vts_1. Vs_2 is a node detected to have a missing link because the Vs_2 is a state node with only one *InDegree* and zero *OutDegree*. Based on (3), there were missing links after Vs_2. Graph G_1 was already in missing nodes mode after Vs_2. The structural and semantic similarity values of the two graphs was 0.00. Based on equation (3), the total similarity value was 0.00, which is less than 50%. However, the graph already has a missing link node so that it can be concluded that the recommended node (MatchingNode) has been found. Vts2' (transition node) in Graph A1 is recommended as a replacement. Both nodes have a total similarity value of less than 50% but were missing links. The Vts_2' in G_1 was recommended as a replacement node (*MatchingNode*). The Vts_2 is the transition node.

TABLE 4
SIMILARITY VALUE BETWEEN NODES

Graf (G_1, G_2)	Cost Value	StrucVal	SemVal	To-tal	Missing-Nodes
Vst_1, Vst_1'	0.00	1.00	1.00	1.00	-
Vts_1, Vts_1'	1.00	0.50	1.00	0.75	-
Vs_2, Vs_2'	1.00	0.50	0.64	0.57	-
Vs_3, Vts_2'	2.00	0.00	0.00	0.00	***Vts₂***
Vs_3, Vs_3'	1.00	0.50	0.62	0.56	-
Vts_3, Vts_3'	1.00	0.50	1.00	0.75	-
Vs_4, Vs_4'	1.00	0.50	0,78	0.64	-
Vts_4, Vts_4'	1.00	0.50	1.00	0.75	-
Ved_5, Ved_5	0.00	1.00	1.00	1.00	-

The search results indicate that the Periodic Issues Library BPM graph from the repository is most similar to the Equipment Recap Laundry BPM graph. The repository's BPM with the highest structural and semantic similarity value to BPM Equipment Recap Laundry is the BPM Periodic Issue Library, shown in Fig. 6.

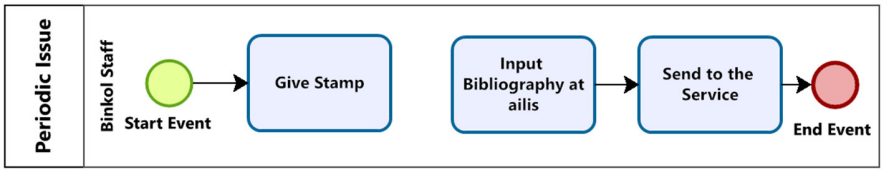

Fig. 6 The Periodic Issues Library BPM chart is a complete BPM found in the repository

Structural similarity is obtained by calculating the GED cost of inter-structural and intra-structural similarity using Greedy. The calculation of structural similarity has been described in previous research [15]. Semantic similarity is obtained by calculating the similarity of lexical information in each metadata and metadata's weight using pairwise comparison [16]. Both graphs have a structural similarity value of 62% (*StrucVal*) and a semantic similarity value of 88 % (*SemVal*). The total similarity value was 75% (*SimVal*) based on (4). The search results found that the missing node from G_1 was a transition node (*Vts2)*. This missing node was stored in the *MissingNode*. The application of *Repolink* method was repeated for each graph in repositories with more than 50% similarity values. When the BPM was incomplete, each node was then compared with all graphs in the repository. In this case, the *MatchingNodes* search will stop if a node found does not meet the specified minimum similarity value (threshold). The search then will continue only for graphs whose nodes meet the threshold.

B. The Result of Detecting and Retrieving Various Missing Nodes of BPM

The search steps of the Repolink method were applied to 32 BPM models listed in Table 2. If a BPM in Table 2 is detected as having missing links (*MissingNodes*), then the retrieval process for similar BPMs will be carried out. The retrieval process starting to comparing the missing links BPM with the complete BPM in the repository (Table 1). Table 3 shows the BPM that has the highest structural and semantic similarity value (*SimVal*) will be recommended as the *MatchingNodes*. The results show that 20 BPM models exhibit various missing-link conditions, each representing a different scenario, while the remaining 12 BPM models contain no missing links.

The missing-link cases are classified into five categories: (1) a single missing process and/or flow; (2) two or more consecutively missing processes and/or flows; (3) two or more non-consecutively missing processes and/or flows; (4)

missing processes and/or flows related to gateways; and (5) missing data objects. Table 3 presents the 20 BPM cases with diverse missing-link conditions, including BPM Matching found in the repository, recommended types for MissingNodes and MatchingNodes, Structural Similarity Value (StrucVal), Semantic Similarity Value (SemVal), and Total Similarity Value (SimVal). Despite the presence of missing links, structural elements such as gateways and data objects can still be identified during the retrieval process. In addition, recommended types for MissingNodes and MatchingNodes may change if the repository is updated and the higher similarity value (structural and semantic) of BPM complete are found.

TABLE 3
TWENTY BPM DIAGRAM WITH MISSING LINKS

Case ID	Missing links BPM	BPM Matching	Recommended Type Missing-Nodes & MatchingNodes	StrucVal (%)	SemVal (%)	SimVal (%)
01	Equipment Recap	Periodic Issue	1 Transition	62.00	88.00	75.00
02	New Member Registration	New Member registration	Sequential 2 Transition and 1 Activity	75.00	83.00	79.00
03	Critics & Suggestion	Critics and Suggestion Management	Unsequential 2 Transition	56.00	77.00	67.00
04	Sales Transaction	Payment Process Retail	Data Object	80.00	94.00	87.00
05	Regular Check-In	Regular Borrowing Library	Gateway	56.00	90.00	73.00
06	Breakfast Listing	Document Review of Library Promotion	Sequential 4 Activity and 5 Transition	62.00	86.00	74.00
07	Performance Audit Result	Field Visit	Unsequential Data object, 1 transition	62.00	86.00	74.00
08	Loan Paymen	Loan Repayment	Unsequential Data object, 1 Transition	58.00	93.00	76.00
09	Visitor Attendace Rate	Library101	Sequential 1 Activity and 2 Transition	70.00	92.00	81.00
10	Review Loan Staff	Review Application Processing	Sequential 2 Transition	57.00	91.00	74.00
11	Staf Pre-Credit	Pre-Qualification Process	Unsequential Gateway and 7 Transition	70.00	94.00	82.00
12	Maintenance Room	Maintenance Public Area	Sequential Data object, 1 activity and 2 transition	66.00	90.00	78.00
13	Resupply	Online Order	Sequential Gateway, 1 Activity and 4 Transition	60.00	87.00	73.00
14	Processing Section	Procuring_goods_and_services	Unsequential Gateway, 3 Activity and 11 Transition	57.00	84.00	70.00
15	Reference Acces	Exhibition_organized_	Sequential Gateway, 8 Activity and 14 Transition	62.00	88.00	75.00
16	Thesis Completion	Returning Book	1 Transition	60.00	85.00	72.00
17	Sorting Book	Weeding Process	Unsequential 1 Activity and 3 Transition	67.00	89.00	78.00
18	Return the Book	Training Process	Unsequential 1 Gateway and 4 Transition	54.00	81.00	68.00
19	Payment After Check Out	Hospital Payment of Cashier	Sequential Gateway and 3 transition	57.00	84.00	71.00
20	Library System	Procurement Process	Sequential 2 Transition	62.00	88.00	75.00

C. Result Evaluation

The expert consensus evaluation involved five participants holding doctoral or master's degrees, each with a minimum of five years of professional experience in business process modelling and computer science. To validate the similarity results produced by Repolink, an expert-based evaluation was conducted using original BPM models, incomplete BPM models, and reconstructed BPM models generated by Repolink. The evaluation focused on three aspects: (1) the correctness of structural similarity ranking between BPM models, (2) the correctness of semantic similarity ranking between BPM models, and (3) the correctness of the combined structural and semantic similarity ranking. Experts independently assessed whether the reconstructed elements were structurally and semantically appropriate with respect to the reference BPM models. The final evaluation results were obtained by aggregating expert judgments to assess consistency and practical validity.

The accuracy of structural and semantic similarity measurement in Repolink was further evaluated using Mean Relative Error (MRE), while the level of expert agreement was measured using Gwet's Agreement Coefficient 2 (AC2). Each expert was asked to assess the degree of structural and semantic similarity (*StrucVal*, *SemVal*, *SimVal*) between BPM models with missing links and corresponding repository models. The expert agreement analysis was conducted using five cases from Table 3, namely Case ID 01, 02, 03, 04, and 05, which represent the five types of missing-link scenarios described earlier. The similarity value produced by Repolink is the actual value (A_i). The semantic and structural similarity value of the expert is the prediction value (P_i). The two values (A_i and P_i) will be calculated for their prediction accuracy value (MRE).

The result of MRE using (4) show RE score of structural and similarity value of Case ID 01, 02, 03, 04 and 05 are less than < 10%. An MRE value of less than 10% means the expert's score is reliable. The expert's score is then calculated by Gwet's AC2 (weight agreement) using (6). The results of AC2 show agreement scores of 0.93 for structural similarity, 0.91 for semantic similarity, and 0.93 for combined structural and semantic similarity. All values fall within the "almost perfect agreement" category [30], confirming the reliability and practical effectiveness of the Repolink similarity measurement method for identifying and retrieving missing metadata in BPM diagrams.

The Big-O notation was used to evaluate the computational efficiency of the RepoLink method. In this study, we used and adjusted time complexity to analyze several stages of the Repolink to determine their respective time complexities. First, when parsing metadata of BPM using the ElementTree XML is $O\ (n^2)$, where n is the total number of elements in the XML tree. Thus, based on the main functions of *the Repolink* method, we calculated the total time complexity of XPDL parsing was $O\ (A\ +\ T\ +\ D)$, where A represents the number of activities, T represents the number of transitions, D represents amount of data object. Second, we counted the total time complexity of sorting metadata which was $O\ (\ A\ +\ T)$. Third, the total time complexity of detecting and constructing missing links was $O\ (M)$, where M represents the amounts of missing nodes. Detecting missing transitions involved iterating over the absent transitions. In the worst case, each activity node might be connected to one or more transitions that needed to be traversed during the search process. Fourth, once identified, retrieving the correct transition source from the XPDL and reinserting it into the graph structure incurred a constant time complexity at *O (1)*. The GED cost was calculated for each node by examining differences in the 'from' and 'to' relationships between nodes. The operations to determine matching or differing nodes and the normalization of edit distances were executed constantly. Fifth, when comparing an incomplete BPMN to a repository of complete BPMNs, each BPMN in the repository was iterated with a complexity in $O\ (B)$, where B represents the number of BPM compared. Sixth, a further iteration through its nodes was required for each BPMN, with a complexity $O\ (A\ *\ B)$. The overall time complexity of the *Repolink* method by combining all the primary components was $O\ (A\ +\ T\ +\ D)\ +\ O\ (A\ +\ T)\ +\ O\ (M)\ +\ O\ (1)\ +\ O\ (B\ *\ A)$. In the worst-case scenario, where the repository contained many BPMNs with considerable activity and structural complexity, the dominant factor in the overall complexity was in the BPMN comparison step.

The efficiency analysis results, mathematically calculated, showed that the *Repolink* method was more efficient because it was based on the direct relationship between nodes from the BPM graph, and the early stopping mechanism was applied to nodes that did not meet the specified threshold. Given that if a graph is compared to all other graphs, and only a similar node is found, then the time complexity must be higher than the *Repolink* method. Thus, using the GED algorithm with direct relationship-based comparison at the node level significantly improves performance. This study's findings indicate that computing-based direct relationship-based comparison at the node is more efficient than complete graph comparison in BPM's missing process search method. Since each node comparison performed constantly in *O (1),* the *Repolink* method achieves efficient graph similarity measurements, even in large-scale repositories. Cases ID 01, 02, 03, 04, and 05 from Table 3 were used to evaluate the performance of the Repolink retrieval algorithm. To assess scalability, experiments were conducted using different sizes of testing datasets, as illustrated in Fig.7. The experimental results of Case ID 01 show that Repolink scales efficiently, requiring approximately 500–800 ms in the best-case scenario and up to 4,000 ms in the worst-case scenario when processing 302 BPMN models. For Case IDs 02, 03, 04, and 05, the results are the same. These results indicate that the information retrieval time remains relatively stable as the dataset size increases, demonstrating the robustness of the proposed method against data growth.

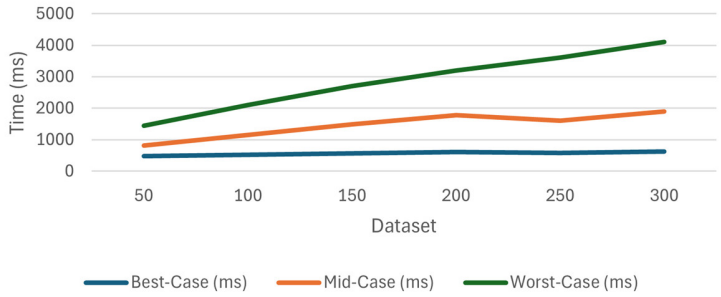

Fig. 7 Time Taken to Finish Based on the Number of Datasets

To avoid oversimplification, the overall complexity of *Repolink* is analyzed by decomposing the algorithm into its main computational stages. The XML parsing and metadata extraction steps operate in linear time O (A + T + D), where A represents the number of nodes, T the number of transitions, and D the number of data objects. Metadata sorting and missing-link detection introduce additional linear traversal costs but do not dominate the runtime. The dominant computational cost arises from the Graph Edit Distance (GED) computation, which requires pairwise node matching and incurs a quadratic complexity of $O(A^2)$ per BPM comparison. When retrieving missing links by comparing a query BPM against B BPMs in the repository, the overall retrieval complexity becomes O (B * A^2). Thus, the performance of Repolink is dominated by GED-based similarity computation rather than preprocessing or XML parsing steps. The reported similarity scores and missing-link recommendations are derived from comparisons between incomplete BPMN models and repository BPMN models using the proposed structural and semantic similarity measures. The efficiency-related results reflect the number of candidates BPMN models processed after applying the similarity threshold and pruning strategy described in Section METHODS.

IV. Discussion

A BPM constructed using BPMN contains complex metadata types. During testing, there were cases where missing links were detected, but no matching nodes were found. The *RepoLink* method has limitations in identifying missing links when the missing metadata involves data types such as activity, gateway, data object, sequence flow, association, and message flow. BPMs with missing links that involve data types beyond those data types mentioned, additional data preprocessing and metadata sorting adjustments are required. The number of BPMs in the repository can be increased to support searching of matching nodes for missing link BPMs. The *RepoLink* leverages the Greedy algorithm to search for replacement nodes, prioritizing search time efficiency; however, this approach may result in reduced accuracy due to its heuristic nature. To increase accuracy, behavioral similarity can be utilized. Compared to the existing information retrieval [5], [6], [22], the *Repolink* search system compares graph nodes individually, resulting in faster performance as shown in Table 5 where there are 12 criteria used in the comparison. Each information retrieval method has a unique purpose. However, *Repolink* is the most efficient method in terms of preprocessing and similarity cost because the search is carried out node by node. Especially, the search process is shorter because it only compares nodes in the graph that meet the threshold value, rather than the entire BPM graph. Besides, the preprocessing cost is lower than that of the other method.

Table 5 highlights the fundamental differences among Repolink, Process Tree-based matching, ORTP Indexing, and the Intelligent GED approach in terms of representation, similarity strategy, scalability, and explainability. Although both Repolink and Intelligent GED rely on Graph Edit Distance, they differ significantly in design goals and operational behavior. Repolink integrates GED with semantic weighting and explicit missing-edge detection, enabling direct comparison of full BPMN models while simultaneously exposing structural discrepancies in an interpretable manner. In contrast, Intelligent GED focuses primarily on structural similarity computation using similarity nodes, edge and behavioral metrics, which provides detailed numerical breakdowns but lacks explicit reporting of missing edges or actionable structural differences.

Process Tree-based methods prioritize formal behavioral semantics by transforming BPMN models into Petri nets and process trees, allowing accurate detection of parallelism, choices, and loops. However, this transformation introduces substantial preprocessing overhead, limiting scalability and real-time applicability. ORTP-based methods shift the computational burden to an expensive offline indexing phase, achieving excellent query-time performance for extremely large repositories but at the cost of flexibility when models are frequently updated.

TABLE 5
EXISTING METHOD COMPARISON

No	Criteria	*Repolink* (GED + Semantic + Missing Edges)	Process Tree[5]	ORTP Index [6]	Intelligent GED [22]
1	Model Representation	BPMN → Graph	Petri Net → Process Tree	Timed Workflow Net	BPMN → Process Graph
2	Main Objective	Structural and semantic similarity + missing edge detection	Behavioural process retrieval	Large-scale workflow retrieval	Structural similarity via GED
3	Similarity Method	Graph Edit Distance (inter & intra) + semantic weighting	Process Tree Edit Distance (structural/behavioural)	Ordering, timing, probability vectors	GED with SN/SE/SB metrics
4	Semantic Support	Yes (labels + types)	No	Limited	Yes (label rules)
5	Behavioural Semantics	Basic (adjacency + missing links)	Strong (parallel, choice, loop)	Very strong (time + probability)	Limited (partial behavioural cues)
6	Explainability	Very high (explicit missing edges)	Low	Medium	High (metric breakdown)
7	Scalability	High (real-time, no indexing)	Low (heavy preprocessing)	Very high after indexing	Medium
8	Preprocessing Cost	Low (O (A + T + D))	Very high	Extremely high (ORTP index)	Medium
9	Similarity Cost	O (B x A)	O (depth × retain²)	O ($m^2 \cdot n^2$)	O ($N^2 + E^2$)
10	Strengths	Fast, semantic-aware, highly explainable	Formal behaviour accuracy	Excellent for huge repositories	Strong structural metrics
11	Limitations	Limited behaviour understanding	Expensive and complex	High indexing cost	Heavy GED mapping
12	Best Use Case	BPMN conformance, debugging	Academic/formal behavioural analysis	Cloud-scale workflow search	Structural variation detection

From a scalability perspective, Repolink achieves high efficiency without relying on indexing, making it suitable for real-time BPMN conformance checking and debugging scenarios. Intelligent GED, while more computationally intensive due to quadratic GED mapping, remains practical for medium-sized repositories and excels at detecting structural variations. Overall, Repolink distinguishes itself by balancing efficiency, semantic awareness, and explainability, whereas Intelligent GED emphasizes structural rigor, Process Tree approaches emphasize formal behavioural correctness, and ORTP prioritizes large-scale retrieval performance.

TABLE 6
QUANTITATIVE COMPARISON

No	Method	*Repolink* (GED + Semantic + Missing Edges)	Process Tree [5]	ORTP Index [6]	Intelligent GED ([22]
1	Processing Complexity	O (A + T) (XML parsing & graph construction)	O(|T|·Rξ) + O(|E|³) + O(|loopRL|⁴)	O ($m \cdot 2^k \cdot n^2$) (offline index construction)	O (A + T)
2	Similarity/Query Complexity	O(A^2) (Graph Edit Distance)	O (depth × retain²)	O(B) (online query)	O ($A^2 + E^2$) (node + edge mapping)
3	Overall Retrieval Complexity	O (B * A^2)	O(B·(|T|·Rξ +|E|3+|loopRL|4+depth·retain2))	O(B·m·2k·n2) +O(B)	O ($B \cdot (A^2 + E^2)$) ≈ O (B * A^2)

Where A = number of nodes; T / E = number of edges; B = number of BPMN models in the repositorym, n, k = ORTP-specific parameters (ordering relations, branching factors); |T| = jumlah transitions pada Petri Net; Rξ = jumlah reachable markings / reachable states; |E| = jumlah edges / control-flow relations; |loopRL| = jumlah loop-related relations.

Table 6 indicates that the proposed method consistently outperforms the Intelligent GED approach in terms of execution time. This performance advantage is primarily due to the use of direct Graph Edit Distance computation combined with lightweight semantic scoring, as well as explicit missing-edge detection, which eliminates the need for complex node-mapping and rule-based matching mechanisms. In contrast, the Process Tree Edit Distance approach incurs higher execution time because it requires an additional model transformation step and relies on tree-based structural and behavioral operations; while these operations provide strong formal semantics, they introduce significant computational overhead. The ORTP-based approach achieves the lowest query time among all methods compared; however, this benefit is obtained only after an expensive offline index construction phase. Consequently, ORTP is well suited for very large process model repositories but is less practical for iterative development tasks or real-time debugging scenarios where frequent model updates are required. Repolink reduces computational steps by progressively eliminating BPM models that do not satisfy the similarity threshold, thereby limiting unnecessary graph

comparisons. This pruning strategy also lowers memory usage, as only relevant candidate models and their corresponding graph structures are retained during the retrieval process.

Despite its effectiveness, this study has several limitations that should be acknowledged. First, Repolink relies primarily on graph-level structural representations to evaluate BPMN models. Complex BPMN constructs such as loops, parallel gateways, and branching are captured through cycles and multiple incoming or outgoing edges. While this representation enables the detection of structural inconsistencies such as missing loop-back edges or unmatched parallel branches. Which does not explicitly model full behavioural semantics, including synchronization constraints or execution ordering. This design choice prioritizes computational efficiency and explainability but limits behavioural expressiveness.

Second, the similarity threshold of 50% used in Repolink is a heuristic choice intended to provide an interpretable decision boundary between low and moderate-to-high similarity BPMN models. Although this threshold aligns with the balanced weighting of structural similarity, semantic similarity, and missing-edge penalties, it has not been statistically optimized through Receiving Operating Characteristic (ROC) analysis or labelled benchmarks. Future work will investigate data-driven threshold selection using sensitivity analysis and ROC/Area Under Curve (AUC) evaluation.

Third, the quantitative comparison with existing methods is based on complexity analysis and controlled simulations rather than full re-implementation of all baseline approaches. While this allows a fair analytical comparison, comprehensive empirical benchmarking would further strengthen the evaluation. In addition, scalability claims are primarily supported by algorithmic design and observed experimental trends; large-scale benchmarking on standardized repositories is left for future work.

Finally, *Repolink* currently focuses on control-flow reconstruction and does not address other BPMN perspectives, such as resource allocation or data-flow semantics. These limitations highlight opportunities for future extensions toward richer behavioural modelling, broader BPMN construct coverage, and learning-based optimization strategies.

V. CONCLUSIONS

The *RepoLink* method was developed to detect and retrieve missing control-flow links in BPMN diagrams, particularly those generated from informal or unstructured documents. The effectiveness of *RepoLink* is demonstrated by its ability to identify missing nodes and edges through combined structural and semantic similarity analysis, as well as by its capability to recommend the most relevant metadata replacements from a repository of reference BPMN models. The experimental evaluation shows that *RepoLink* consistently detects structural inconsistencies and reconstructs missing links in various BPMN scenarios.

The gradual elimination of dissimilar graphs during the retrieval process significantly reduces computational steps and memory usage. This design choice is supported by the complexity analysis, which indicates that the overall retrieval complexity is dominated by $O(B * A^2)$, enabling efficient comparison across large repositories. In addition to performance efficiency, *RepoLin*k provides explainable results by explicitly reporting missing edges and their suggested replacements, making the approach suitable for BPMN conformance checking and model debugging.

Nevertheless, the proposed approach primarily operates at the graph-structural level and does not fully capture detailed behavioural semantics or other BPMN perspectives such as resource and data-flow modelling. Furthermore, scalability conclusions are currently supported by complexity analysis and controlled experiments rather than large-scale standardized benchmarks. These limitations motivate future research directions.

Future work will focus on enhancing *RepoLink* by incorporating learning-based techniques to improve recommendation accuracy in more complex or ambiguous BPMN scenarios. In addition, the repository will be extended with cross-domain BPMN datasets to further evaluate the scalability and generalizability of the proposed method.

Author Contributions: *[Kristina]*: Methodology, Data Curation, Software, Validation, Investigation, Writing - Original Draft, Visualization, *[Ary Mazharuddin Shiddiqi]*: Conceptualization, Writing - Review & Editing, Supervision, Project Administration, Funding acquition, *[Daniel Siahaan]*: Writing - Original Draf, Formal Analyze, Resources, [*Adrian Forca*]: Writing-Review & Editing, Formal Analyze.

All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Department of Informatics, Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia, under the Departmental Research Grant Batch 1, 2026.

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

Data Availability: Dataset link will be available upon request.

Informed Consent: There were no human subjects.

Institutional Review Board Statement: Not applicable.

Animal Subjects: There were no animal subjects.

ORCID:
Kristina: https://orcid.org/0000-0001-6828-3236
Ary Mazharuddin Shiddiqi: https://orcid.org/0000-0002-8762-3141
Daniel Siahaan: https://orcid.org/0000-0001-6560-2975
Adrian Forca: https://orcid.org/0000-0002-4634-1201

REFERENCES

[1] S. Sholiq, R. Sarno, and E. S. Astuti, "Generating BPMN Diagram from Textual Requirements," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10, pp. 10079–10093, 2022, doi: 10.1016/j.jksuci.2022.10.007.

[2] U. Indahyanti, A. Djunaidy, and D. Siahaan, "Auto-Generating Business Process Model from Heterogeneous Documents: A Comprehensive Literature Survey," in *2022 9th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, 2022, pp. 239–243. doi: 10.23919/EECSI56542.2022.9946460.

[3] I. Compagnucci, F. Corradini, F. Fornari, and B. Re, "A Study on the Usage of the BPMN Notation for Designing Process Collaboration, Choreography, and Conversation Models," *Business and Information Systems Engineering*, no. March, 2023, doi: 10.1007/s12599-023-00818-7.

[4] I. Tangkawarow, R. Sarno, and D. Siahaan, "ID2SBVR: A Method for Extracting Business Vocabulary and Rules from an Informal Document," *Big Data and Cognitve Computing*, vol. 6, no. 4, pp. 1–23, 2022, doi: http://dx.doi.org/10.3390/bdcc6040119.

[5] H. Huang *et al.*, "Efficiently Querying Large Process Model Repositories in Smart City Cloud Workflow Systems Based on Quantitative Ordering Relations," *Inf Sci (N Y)*, vol. 495, pp. 100–115, 2019, doi: 10.1016/j.ins.2019.04.058.

[6] R. Zhu *et al.*, "Business Process Retrieval from Large Model Repositories for Industry 4.0," *IEEE Trans Serv Comput*, vol. 17, no. 1, pp. 306–321, 2024, doi: 10.1109/TSC.2023.3348294.

[7] S. Lee, I. Choi, H. Kim, J. Lim, and S. Sung, "Comprehensive Simulation and Redesign System for Business Process and Organizational Structure," *IEEE Access*, vol. 8, pp. 106322–106333, 2020, doi: 10.1109/ACCESS.2020.3000248.

[8] A. Kumar and R. Liu, "Business Workflow Optimization Through Process Model Redesign," *IEEE Trans Eng Manag*, vol. 69, no. 6, pp. 3068–3084, 2022, doi: 10.1109/TEM.2020.3028040.

[9] F. Milani, K. Kubrak, and J. Nava, "Strategic Redesign of Business Processes in the Digital Age: A Framework," *Data Knowl Eng*, vol. 154, no. September, p. 102367, 2024, doi: 10.1016/j.datak.2024.102367.

[10] A. Haj, Y. Balouki, and T. Gadi, "Automated Identification of semantic Similarity between Concepts of Textual Business Rules," *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 1, pp. 147–156, 2020.

[11] M. T. Garcia, M. M. Nunes, M. Fantinato, S. M. Peres, and L. H. Thom, "BPMN-Sim: A Multilevel Structural Similarity Technique for BPMN Process Models," *Inf Syst*, vol. 116, p. 102211, 2023, doi: 10.1016/j.is.2023.102211.

[12] F. Niu, C. Li, J. Ge, L. Wen, Z. Li, and B. Luo, "Measuring Business Process Behavioral Similarity Based on Token Log Profile," *IEEE Trans Serv Comput*, vol. 15, no. 6, pp. 3344–3357, 2022, doi: 10.1109/TSC.2021.3104898.

[13] R. Fauzan, D. Siahaan, S. Rochimah, and E. Triandini, "Automated Class Diagram Assessment using Semantic and Structural Similarities," *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 2, pp. 52–66, 2021, doi: 10.22266/ijies2021.0430.06.

[14] R. Fauzan, D. Siahaan, S. Rochimah, and E. Triandini, "An Automated Statechart Diagram Assessment using Semantic and Structural Similarities," *Int J Adv Sci Eng Inf Technol*, vol. 11, no. 6, pp. 2436–2444, 2021, doi: 10.18517/ijaseit.11.6.13372.

[15] J. Yan, X. C. Yin, W. Lin, C. Deng, H. Zha, and X. Yang, "A Short Survey of Recent Advances in Graph Matching," in *The 2016 ACM International Conference on Multimedia Retrieval*, 2016, pp. 167–174. doi: 10.1145/2911996.2912035.

[16] S. Riyanarto, S. Kelly R, and R. Septiarakhman, "Graph-Based Approach for Modeling and Matching Parallel Business Processes," *International Information Institute*, vol. 21, no. 5, pp. 1603–1614, 2018, doi: 10.1016/j.is.2015.10.012.

[17] Y. Setiawan, K. R. Sungkono, and R. Sarno, "A New Similarity Method Based on Weighted Graph Models for Matching Parallel Business Process Models," *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 5, pp. 267–276, 2020, doi: 10.22266/ijies2020.1031.24.

[18] H. Munawaroh, D. O. Siahaan, R. Fauzan, and E. Triandini, "Structural Similarity Measurement using Graph Edit Distance-Greedy on State chart Diagrams," *2020 2nd International Conference on Cybernetics and Intelligent System, ICORIS 2020*, 2020, doi: 10.1109/ICORIS50180.2020.9320764.

[19] F. Zulfa, D. O. Siahaan, R. Fauzan, and E. Triandini, "Inter-Structure and Intra-Structure Similarity of Use Case Diagram using Greedy Graph Edit Distance," *2020 2nd International Conference on Cybernetics and Intelligent System, ICORIS 2020*, pp. 3–8, 2020, doi: 10.1109/ICORIS50180.2020.9320840.

[20] Kristina, A. M. Shiddiqi, D. Siahaan, and H. Munawaroh, "Structural Similarity Assessment of Business Process Graph Using GED-Greedy," *Proceedings - 13th IEEE International Conference on Control System, Computing and Engineering, ICCSCE 2023*, pp. 45–50, 2023, doi: 10.1109/ICCSCE58721.2023.10237172.

[21] Kristina, A. M. Shiddiqi, and D. Siahaan, "Business Process Model Diagram Similarity Measurement Using Pairwise Comparison," *Conference Proceeding - 23rd International Symposium on Communications and Information Technologies, ISCIT 2024*, pp. 47–52, 2024, doi: 10.1109/ISCIT63075.2024.10793688.

[22] A. Sohail, A. Haseeb, M. Rehman, D. D. Dominic, and M. A. Butt, "An intelligent Graph Edit Distance-Based Approach for Finding Business Process Similarities," *Computers, Materials and Continua*, vol. 69, no. 3, pp. 3603–3618, 2021, doi: 10.32604/cmc.2021.017795.

[23] S. D. Azmi and R. Kusumaningrum, "Relevance Feedback using Genetic Algorithm on Information Retrieval for Indonesian Language Documents," *Journal of Information Systems Engineering and Business Intelligence*, vol. 5, no. 2, p. 171, Oct. 2019, doi: 10.20473/jisebi.5.2.171-182.

[24] M. C. Wijanto, R. Rachmadiany, and O. Karnalim, "Thesis Supervisor Recommendation with Representative Content and Information Retrieval," *Journal of Information Systems Engineering and Business Intelligence*, vol. 6, no. 2, p. 143, Oct. 2020, doi: 10.20473/jisebi.6.2.143-150.

[25] R. S. Perdana and P. P. Adikara, "Multi-task Learning for Named Entity Recognition and Intent Classification in Natural Language Understanding Applications," *Journal of Information Systems Engineering and Business Intelligence*, vol. 11, no. 1, pp. 1–16, Feb. 2025, doi: 10.20473/jisebi.11.1.1-16.

[26] S. Sholiq and M. A. Yaqin, "BPMN Diagram Dataset: Comprehensive Collection of Functional Requirements," *Data Brief*, vol. 57, p. 110882, 2024, doi: 10.1016/j.dib.2024.110882.

[27] K. L. Gwet, *Handbook of Inter-Rater Reliability Fith Edition the Definitive Guide to Measuring the Extent of Agreement Among Raters Volume 1 Analysis of Categorical Ratings*. Gaithersburg: AgreeStat Analytic, 2021.

[28] M. Shrif, S. Barakat, Z. Al-Sadoon, O. Mostafa, and R. Awad, "Optimized Neural Network-based Model to Predict the Shear Strength of Trapezoidal-corrugated Steel Webs," *Heliyon*, vol. 10, no. 15, Aug. 2024, doi: 10.1016/j.heliyon.2024.e35778.

[29] H. Khoshvaght, R. R. Permala, A. Razmjou, and M. Khiadani, "A Critical Review on Selecting Performance Evaluation Metrics for Supervised Machine Learning Models in Wastewater Quality Prediction," Dec. 01, 2025, *Elsevier Ltd*. doi: 10.1016/j.jece.2025.119675.

[30] L. Mertens, K. De Martelaer, A. Sääkslahti, and E. D'hondt, "The inter-rater and intra-rater reliability of the Actual Aquatic Skills Test (Aast) for Assessing Young Children's Motor Competence in the Water," *Int J Environ Res Public Health*, vol. 19, no. 1, Jan. 2022, doi: 10.3390/ijerph19010446.

[31] S. Phalke, Y. Vaidya, and S. Metkar, "Big-O Time Complexity Analysis of Algorithm," *2022 International Conference on Signal and Information Processing, IConSIP 2022*, pp. 1–5, 2022, doi: 10.1109/ICoNSIP49665.2022.10007469.