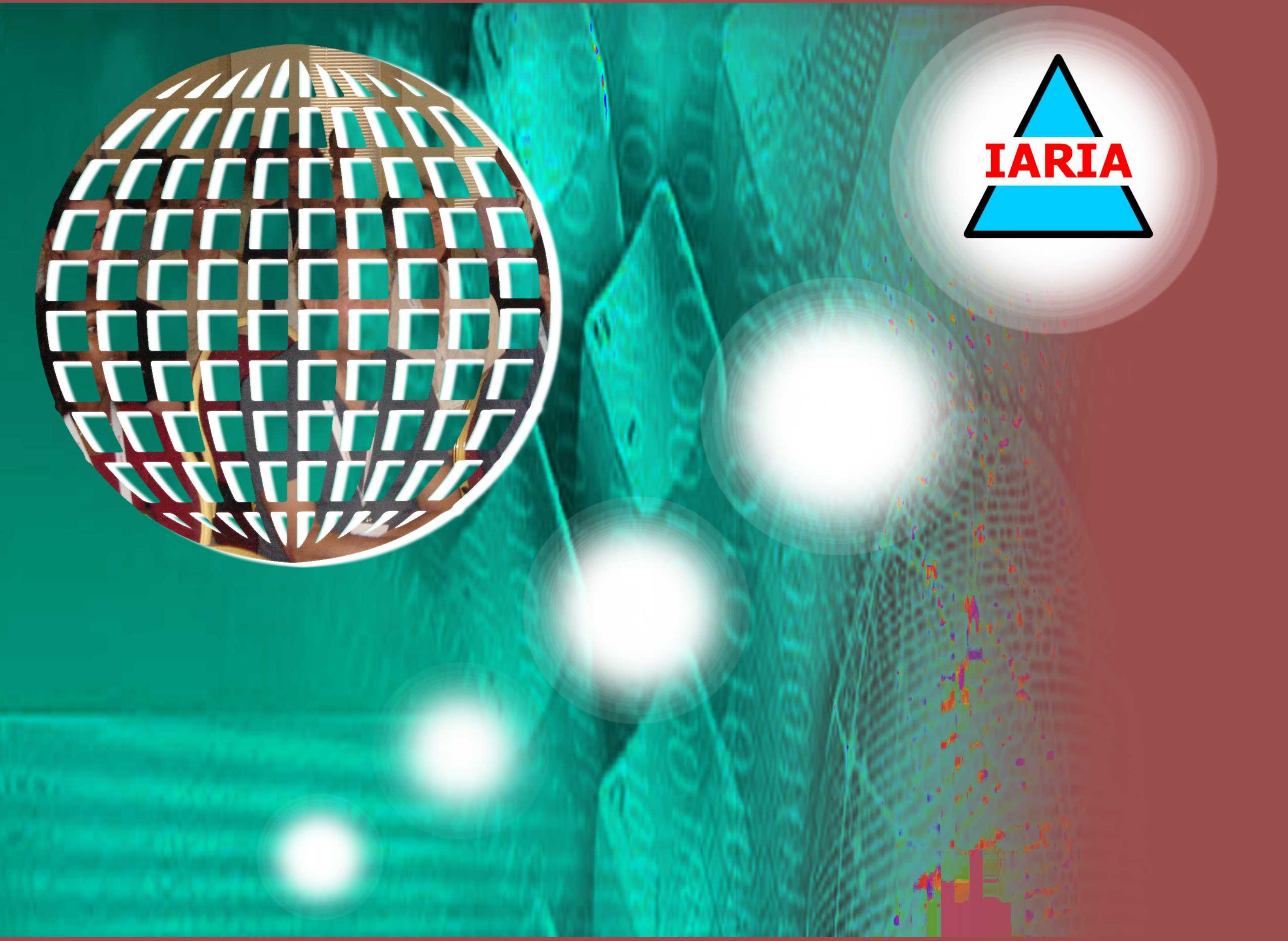
# International Journal on Advances in Software



2025 vol. 18 nr. 3&4

The *International Journal on Advances in Software* is published by IARIA.
ISSN: 1942-2628
journals site: http://www.iariajournals.org
contact: petre@iaria.org

**International Journal on Advances in Software**
**Volume 18, Number 3&4, 2025**

**International Journal on Advances in Software**
**Volume 18, Numbers 3&4, 2025**

**CONTENTS**

# Design Pattern Detection in Source Code: A Neural Graph Database Approach as an Ensemble Base Model

Roy Oberhauser[0000-0002-7606-8226] and Peter Schneider
Computer Science Dept.
Aalen University
Aalen, Germany
e-mail: roy.oberhauser@hs-aalen.de, peter.schneider@studmail.htw-aalen.de

*Abstract* - **Software design patterns offer reusable structural solutions that support developers and maintainers in addressing common design problems. Their abstractions can support program code documentation and comprehension, yet manual pattern documentation via code or code-related artifacts (documents, models) can be unreliable, incomplete, and labor-intensive. Various automated Design Pattern Detection (DPD) techniques have been proposed, yet adoption remains limited and further investigation of viable solutions is needed. Towards more effective automated DPD, this paper contributes our Neural Graph Database approach DPD-NGDB, which also functions as a base model in our Ensembles Methods approach DPD-EM. The realization demonstrates the feasibility of our approaches, while the evaluation compares and benchmarks the DPD performance against a Gang-of-Four (GoF) software design pattern dataset, demonstrating its potential.**

***Keywords – software design pattern detection; machine learning; neural graph databases; graph neural networks; ensemble methods; software design patterns; software engineering.***

## I. INTRODUCTION

This paper extends our previous work [1], in that it investigates further potential DPD methods, leveraging Ensembles Methods (EMs), Neural Graph DataBases (NGDBs), and Graph Neural Networks (GNNs) approaches for DPD. The breadth and depth of the evaluation is extended to and benchmarked against the entire Gang of Four (GoF) design patterns. Our prior hybrid DPD approach has been changed to an EM approach.

Program source code worldwide continues to rapidly expand, yet code comprehension remains a limiting productivity factor. Program comprehension may consume up to 70% of the software engineering effort [2]. Activities involving program comprehension include investigating functionality, internal structures, dependencies, run-time interactions, execution patterns, and program utilization; adding or modifying functionality; assessing the design quality; and domain understanding of the system [3]. Code that is not correctly understood by programmers impacts quality and efficiency.

Software Design Patterns (DPs) have been documented and popularized, including the Gang of Four (GoF) [4] and POSA [5]. The application of abstracted and documented solutions to recurring software design problems has been a boon to improving software design quality, efficiency, aiding comprehension, refactoring, reuse, reverse-engineering, and maintenance tasks. These well-known macrostructures or associated pattern terminology in code can serve as beacons to abstracted macrostructures, and as such may help identify aspects such as the author's intention or the purpose of a code segment, which, in turn, supports program comprehension.

Automated DPD in code overs various benefits, including: quicker comprehension of DP-related structural aspects of unfamiliar software; automatically documenting DPs; supplementing and validating the design documentation; reducing dependence on error-prone DP documentation; and detection of inadequately implemented DPs. Yet the challenges for automated DPD include: 1) tool support for heterogeneous programming languages, as DPs are independent of programming language; 2) internationalization and labeling, since developers may name and comment in their natural language or any way they like; 3) varying pattern abstraction levels, such as design vs. architectural patterns; 4) similarities and intent differentiation, since some similar pattern structures are primarily differentiated via their intention; 6) DP localization to indicate where in code a DP was detected; and 7) detecting variants, since each pattern implementation is unique.

Traditional DPD approaches rely on static code analysis, with tools extracting structural features such as class hierarchies, method invocations, and object instantiations to identify patterns [6]. Curated design pattern datasets, such as the Pattern-like Micro-Architecture Repository (P-MARt) [7][8], provide a collection of micro-architectures from known open source projects which applied the canonical design patterns and can serve as benchmarks for evaluating DPD techniques. While various DPD approaches have been explored [9][10], no approach has thus far achieved significant practical traction, and thus additional investigation into further possibly viable approaches and improvements is warranted.

GNNs [11] are a class of Deep Learning (DL) models that are specifically designed to work with graph-structured data. They learn representations that capture the features of individual nodes and the relationships between them. Unlike traditional neural networks (NNs) that assume independent data points, GNNs leverage the topology of graphs to propagate information across nodes via edges, making them particularly suitable for domains where relationships are key.

NGDBs [12][13] extend classical graph database systems by integrating GNNs directly into the graph storage and the query engine. An NGDB is designed to store, manage and query graphs using both traditional graph operations and neural inference to enrich incomplete or uncertain data, perform link prediction, and extract embeddings on-the-fly. This hybrid paradigm unifies transactional graph queries (e.g., Cypher [14]) with GNN-based tasks (e.g., node classification, link prediction), enabling real-time inference powered by the rich information encoded in a Labeled Property Graph (LPG) data model.

EMs [15][16] incorporate a finite set of alternative Machine Learning (ML) algorithms and models to enhance predictive performance, especially where a single model may not perform ideally. As design patterns can exhibit significant variance and non-linear relationships, we believe that no single technique (ensemble) will likely perform well in all circumstances. Thus, a mix of models (ensembles) may improve results when faced with significant variance, diversity, and non-linear relationships in the datasets, as is often the case with DPD.

Our previous work includes: our ML-based DPD approach DPDML that utilizes semantic and static analysis metrics [17]; our hybrid DPD approach HyDPD [18], which combines our ML-based model with an expert-based graph analysis model; and HyDPD-B [1], which applies a Bayesian network probabilistic reasoning to integrate various DPD subsystems, including HyDPD-ML utilizing graph embeddings, with our expert rule system with DP rule language and micropattern detection.

This paper contributes our NGDB-based solution approach (DPD-NGDB), which is embedded as a base model in our EM approach (DPD-EM). We describe our realization, which demonstrates the feasibility of the DPD approaches. Our evaluation uses a dataset consisting of the Gang-of-Four (GoF) design patterns benchmarked against the P-MARt repository.

This paper is structured as follows: the next section discusses related work. Section III describes our solution. In Section IV, our realization is presented, which is followed by our evaluation in Section V. Finally, a conclusion is provided.

## II. Related Work

Surveys including categorizations of DPD approaches include Al-Obeida et al. [9] and Yarahmadi and Hasheminejad [10]. Graph-based DPD approaches include: Yu et al. [19] transform code to UML class diagrams, analyze the XMI for sub-patterns in class-relationship directed graphs; Mayvan and Rasoolzadegan [20] use a UML semantic graph; Bernardi et al. [21] apply a DSL-driven graph matching approach; DesPaD [22] extract an abstract syntax tree from code, create a single large graph model of a project, and then apply an isomorphic sub-graph search method. Further isomorphic subgraph approaches include Pande et al. [23] and Pradhan et al. [24], both of which require UML class diagrams.

Learning-based approaches map the DPD problem to a learning problem, and can involve classification, decision trees, feature maps or vectors, Artificial Neural Networks (ANNs), etc. Examples include Alhusain et al. [25], Zanoni et al. [26], Galli et al. [27], Ferenc et al. [28], Uchiyama et al. [29], and Dwivedi et al. [30]. Thaller et al. [31] describe a micro-structure-based structural analysis approach based on feature maps. Chihada et al. [32] convert code to class diagrams, which are then transformed to graphs, and have experts create feature vectors for each role based on object-oriented metrics and then apply ML.

Additional approaches include: reasoning-based approaches such as Wang et al. [33] based on matrices; rule-based approaches like Sempatrec [34] and the ontology-based FiG [35]; metric-based approaches such as MAPeD [36], Uchiyama et al. [29], and Dwivedi et al. [37]; Fontana et al. [38] analyze microstructures based on an abstract syntax tree; semantic-analysis style includes Issaoui et al. [39]; while DP-Miner [40] uses a matrix-based approach based on UML for structural, behavioral, and semantic analysis. Singh et al. [41] combines static rules with graph analysis. GEML [42] initializes a population of random structures, applying genetic algorithms to mutate and generate new patterns from the initial population. Kouli and Rasoolzadegan [43] utilize micro-patterns with binary logic.

Graph-based code representations have emerged to capture syntactic and semantic dependencies more effectively. Liu et al. [44] propose a Code Property Graph (CPG)-based GNNs for code similarity detection, achieving high performance by learning multi-hop dependencies. Ampatzoglou et al. [45] use neural sub-graph matching with GNNs to detect design patterns in large codebases, demonstrating robustness to structural variations. Li et al. [46] integrated multi-feature fusion with GNNs, combining semantic embeddings and structural metrics for enhanced detection in real-world projects.

NGDBs represent an innovative extension, integrating GNNs with graph databases such as Memgraph, offering real-time pattern analysis [12][13]. NGDBs enable dynamic feature computation and scalable querying, which could support tasks like pattern recognition in evolving codebases [47].

Ensemble Methods have proven effective in improving robustness and accuracy for ML tasks [15][16]. For DPD, ensembles like Random Forests and stacked generalization combine multiple classifiers to leverage complementary strengths [48]. However, integrating GNNs, NGDBs, and traditional ML models in ensembles remains underexplored, particularly for addressing class imbalance and generalization, and we see an opportunity and flexibility for addressing DPD issues via its utilization.

## III. Analysis and Requirements

DPD approaches can arguably be categorized into three primary approaches: 1) learning-based, where DPs are (semi-)automatically learned (e.g., via supervised learning) from provided data and requiring minimal expert intervention; 2) knowledge-based, whereby an expert defines DPs by describing elements and their associations; and 3) similarity-based, whereby DPs are grouped based on similar metrics or characteristics.

### *A. Analysis*

DPD in object-oriented code, such as Java, involves identifying structural, creational, and behavioral patterns from the Gang of Four (GoF) catalog. However, several challenges complicate this task:

*C1: Variability in Implementations*: DPD must take variability into account, since patterns can be implemented in non-standard ways, with variations in naming, structure, or partial realizations. Traditional rule-based tools like P-MARt or PINOT (Pattern Identification using Optimization and Transformation) [49] often fail to detect these variants due to rigid matching criteria.

*C2: Dataset Limitations*: Existing "labeled" DPD datasets often consist of clean, textbook examples but lack diversity, including real-world project integrations. Imbalances between pattern and non-pattern instances further complicate ML approaches.

*C3: Feature Representation*: Static analysis alone misses dynamic behaviors, while graph-based representations (e.g., Abstract Syntax Tree (AST) or Call Graph (CG)) require sophisticated handling to capture inter-class relationships without losing semantic information.

*C4: Scalability*: DPD must handle large codebases efficiently.

*C5: Generalization*: DPD must generalize to unseen projects, avoiding overfitting to specific implementations. Ensemble methods and GNNs show promise in addressing these by combining complementary strengths:

- ML classifiers for feature-based detection,
- GNNs for structural resilience, and
- NGDBs for real-time querying.

## B. Requirements

To address these challenges, we identify the following DPD requirements, categorized by functional and non-functional:

### 1) Functional Requirements (FR):

1. *DPD Coverage*: The system shall detect all 23 GoF design patterns (creational, structural, behavioral) in Java source code or bytecode.
2. *Feature Extraction*: Extract multi-modal features including numerical (e.g., method counts, complexity), graph-based (e.g., ASTs, CGs), and derived structural metrics (e.g., inheritance degrees).
3. *Model Training and Inference*: Implement training pipelines for individual models (e.g., SVM, GNN, NGDB) and an ensemble combiner (e.g., using soft voting).
4. *Dataset Management*: Extend the P-MARt dataset with real-world samples, class-level labeling, and imbalance handling via oversampling.
5. *Evaluation Framework*: Provide an evaluation framework that offers cross-validation (K-Fold, Leave-One-Project-Out (LOPO)) and performance metrics (F1-Score, confidence).
6. *Real-time Processing*: Support batch processing of multiple repositories and real-time pattern detection for integration into development workflows.

### 2) Non-Functional Requirements (NFRs):

1. *Performance:* The system must process medium-sized Java projects (up to 10,000 classes) within reasonable time constraints (under 30 minutes for complete analysis).
2. *Scalability:* Support horizontal scaling for batch processing of multiple repositories simultaneously.
3. *Accuracy:* Achieve F1-scores above 0.80 for common design patterns and maintain robustness against code variations.
4. *Extensibility:* Provide a modular architecture allowing addition of new pattern types, feature extractors, and an ensemble integration of further base models.
5. *Reproducibility:* Ensure deterministic results through proper random seed management and version control of models and datasets.

## IV. Solution

Our DPD solution approach incorporates the features and conceptual architecture described in the following.

## A. Features

*Full GoF Dataset:* The DPD scope encompasses the 23 GoF design patterns, covering creational, structural, and behavioral design patterns.

*Class-Level Feature Aggregation*: Since many design patterns are object-oriented, the feature extraction pipeline operates at the class level to preserve contextual relationships between methods and their containing classes.

*Multi-Modal Feature Engineering:* The feature extraction incorporates:

- Numerical features from AST analysis,
- Graph-based features from CG analysis, and
- Structural features like inheritance relationships.

*Multi-model Support:* The open approach can incorporate multiple varying base model types, including GNNs, NGDBs, SVMs, and can be combined with Ensemble Methods.

## B. Modular Architecture

A modular, pipeline-based architecture separates concerns between feature extraction, model training, and inference. The architecture is designed to support the identified functional and non-functional requirements while maintaining flexibility for future extensions.

It consists of four primary modules, each responsible for a specific stage in the DPD process, as shown in Figure 1.

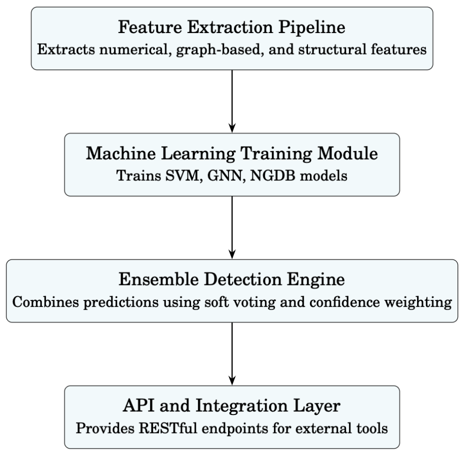

Feature Extraction Pipeline
Extracts numerical, graph-based, and structural features
Machine Learning Training Module
Trains SVM, GNN, NGDB models
Ensemble Detection Engine
Combines predictions using soft voting and confidence weighting
API and Integration Layer
Provides RESTful endpoints for external tools

Figure 1. DPD-EM Module Architecture

1. *Feature Extraction*: Processes source code (or bytecode/binary code if desired) to extract numerical, graph-based, and structural features at both method and class levels.
2. *ML Training*: Implements training pipelines for individual models (e.g., GNN, NGDB, SVM) including training evaluation and model storage.
3. *DPD and Ensembles*: The DPD predictions from a single (or multiple) trained models can be flexibly utilized. In the multi-model case, any ensemble technique can be applied (voting, expert rules, decision trees, etc.) to leverage the complementary strengths of individual models as desired and thereby enhance overall DPD performance. While our DPD-EM ensemble method offers flexible multi-model support, when only a single ensembles classification results are used, we then refer to its specific model (e.g., DPD-NGDB), even though our modular architecture remains ensemble-enabled.
4. *System Integration and API*: Provides Web API endpoints for training, inference, and result management with standardized interfaces for external tool integration.

#### *1) Data Flow Stages*

The system processes data through a series of stages, each transforming the input into a more refined output. The stages are as follows and illustrated in Figure 2.

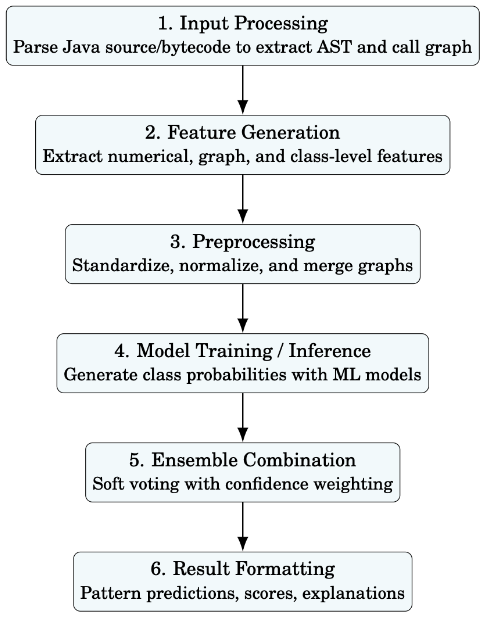

1. Input Processing
Parse Java source/bytecode to extract AST and call graph
2. Feature Generation
Extract numerical, graph, and class-level features
3. Preprocessing
Standardize, normalize, and merge graphs
4. Model Training / Inference
Generate class probabilities with ML models
5. Ensemble Combination
Soft voting with confidence weighting
6. Result Formatting
Pattern predictions, scores, explanations

Figure 2. DPD-EM Data Flow Stages

1. *Input Processing:* Raw source code (or bytecode/binary) is parsed to extract AST and CG information.
2. *Feature Generation:* Multi-modal feature extraction generates numerical metrics, graph structures, and derived features at the class-level (which subsumes method-level).
3. *Preprocessing:* Feature standardization, normalization, and graph merging prepare data for ML models.
4. *Model Training/Inference*: Individual models are trained or used for inference, generating probability distributions over pattern classes.
5. *Ensemble Combination:* Any ensemble technique such as soft-voting with confidence weighting) to combine individual model predictions and enhance overall predictions.
6. *Result Formatting:* Standardized output includes pattern predictions, confidence scores, and detailed explanations.

## V. REALIZATION

This section describes the realization of our DPD solution approach, providing details about our modules and pipeline.

### *A. Module Overview*

The solution architecture was realized in Python, with each module addressing distinct functionality within the DPD pipeline. Any filenames listed are for reference purposes for subsequent descriptions and not intended to be comprehensive. The four modules consist of:

Feature Extraction Module (M1): Implements numerical feature extraction, including from ASTs, CGs, or static analysis metrics:

- NumericalFeaturePreprocessing.py: Handles feature standardization and preprocessing.
- GraphFeatureProcessor.py: Processes method-level CGs into class-level representations.

ML Training Module (M2): Implements training pipelines for individual models (e.g., GNN, NGDB, SVM) including evaluation and model persistence:

- TrainGNN.py: Implements the training pipeline for GNNs.
- TrainPatternPipeline.py: Implements the training pipeline for NGDB approaches.
- TrainSVM.py: Implements the training pipeline for SVMs.

DPD and Ensembles Module (M3): Implements the ensemble technique that combines predictions from multiple trained models. It currently applies soft voting and confidence-based weighting, but any other EM technique can be applied:

- Detection.py: Provides unified interfaces for applying trained models for DPD.

System Integration and API Module (M4): Provides RESTful Web API endpoints for training, inference, and result management. Offers standardized interfaces for external tool integration:

- MLAPI.py: Offers system integration via a FastAPI backend.

### *B. Feature Extraction Module (M1)*

Accurate DPD depends on the quality and structure of the underlying features extracted from software artifacts. Our multi-stage pipeline module transforms raw code into numerical and graph-based representations suitable for supervised learning and graph-based inference. Our AST and CG extraction utilizes JavaParser for Java source code and SootUp for bytecode, but multi-language support is feasible.

*1) Numerical Feature Extraction:* The numerical feature extraction process (NumericalFeaturePreprocessing.py) operates on AST, CG data, and static metrics to generate quantitative representations of code characteristics. Both bytecode and source code extraction modes are supported.

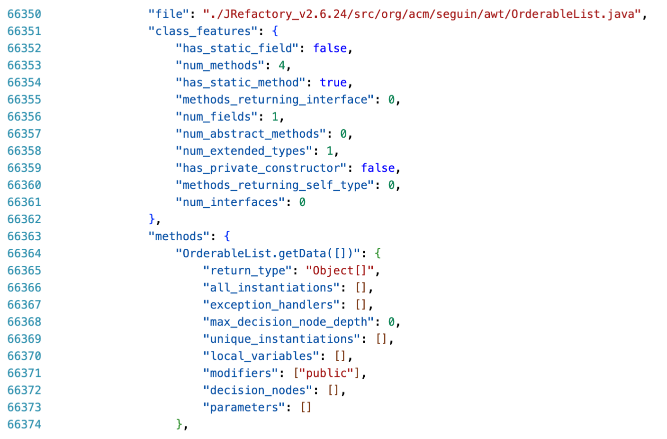
Numerical features are extracted from code to support analysis and classification. The method-level features extracted include:

- The number of method parameters and object instantiations,
- The presence of modifier flags such as public, private, static, final, abstract, and synchronized,
- The complexity of return types, categorized as: void (0), primitive type (1), custom class (2), generic type (3),
- Cyclomatic complexity,
- Maximum nesting depth, and
- The number of local variables and exception handlers.

To account for dynamic behavior and interprocedural relationships, metrics derived from the static CG are incorporated, including:

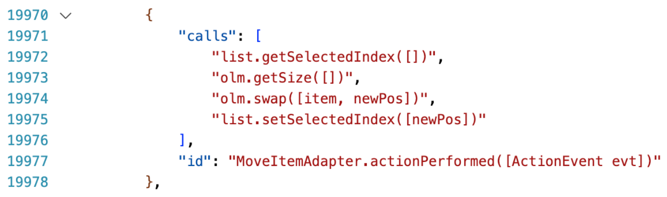
- The number of incoming and outgoing calls,
- The presence of self-calls, and
- The number of unique calling methods.

Class-level aggregation is performed, which summarizes method-level features using statistical operations such as sum, mean, maximum, and minimum. This enables a holistic view of each class based on the behavior of its constituent methods. A sample of the numerical feature extraction is shown in Figure 3. A sample of the CG extraction is depicted in Figure 4.



```
            "file": "./JRefactory_v2.6.24/src/org/acm/seguin/awt/OrderableList.java",
            "class_features": {
                "has_static_field": false,
                "num_methods": 4,
                "has_static_method": true,
                "methods_returning_interface": 0,
                "num_fields": 1,
                "num_abstract_methods": 0,
                "num_extended_types": 1,
                "has_private_constructor": false,
                "methods_returning_self_type": 0,
                "num_interfaces": 0
            },
            "methods": {
                "OrderableList.getData([])": {
                    "return_type": "Object[]",
                    "all_instantiations": [],
                    "exception_handlers": [],
                    "max_decision_node_depth": 0,
                    "unique_instantiations": [],
                    "local_variables": [],
                    "modifiers": ["public"],
                    "decision_nodes": [],
                    "parameters": []
                },
```

Figure 3. JSON snippet showing feature extraction for OrderableList within the JRefactory project.



```
        {
            "calls": [
                "list.getSelectedIndex([])",
                "olm.getSize([])",
                "olm.swap([item, newPos])",
                "list.setSelectedIndex([newPos])"
            ],
            "id": "MoveItemAdapter.actionPerformed([ActionEvent evt])"
        },
```

Figure 4. JSON snippet of extracted call graph data for MoveItemAdapter within the JRefactory project.

*2) Feature Standardization and Preprocessing*: Feature standardization scales a feature x to the interval [0,1] applying MinMax scaling. The process operates in two modes:

*a) Training Mode:* computes the global minimum and maximum values across all repositories, fits MinMaxScaler parameters, and serializes them as JSON for reproducibility.

*b) Usage Mode:* loads previously saved scaling parameters to ensure consistent feature transformation, allowing values to exceed [0,1] when input data exceeds training distribution bounds.

*3) Graph Feature Processing and Aggregation:* Graph feature processing transforms method-level CGs to class-level representations suitable for GNN processing.

*a) Method-level CGs to Class-level Transformation:* The process extracts class identifiers from method signatures using regular expressions as shown in Figure 5.

```
def extract_class_id(method_id):
    core = re.sub(r"\(.*\)$", "", method_id)
    idx  = core.rfind('.')
    return core[:idx] if idx > 0 else core
```

Figure 5. Class identifier extraction via regular expressions

The transformation proceeds in three steps:

*1. Node Consolidation*: Creates one node per unique class ID containing the class identifier and feature vector from standardized numerical features, pattern labels available in training mode, method counts indicating class complexity, and additional structural metadata.

*2. Edge Aggregation*: Transforms method-to-method call relationships into class-to-class dependencies using set operations to eliminate duplicates and filtering self-loops automatically. The aggregation process includes class-level feature aggregation and distribution, feature consistency validation and quality checks, and support for missing feature imputation and default values.

*3. Label Propagation*: In training mode, pattern labels are propagated from class-level annotations to individual method nodes. This enables: ground truth establishment for supervised learning, label consistency validation across method-class hierarchies, support for partial labeling and semi-supervised approaches, and quality assurance for label accuracy and completeness.

*b) Class-Level Graph Construction:* This execution step represents the core transformation that supports higher-level DPD. It involves two primary operations: node consolidation and edge aggregation.

Node Consolidation Strategy: For each unique class ID, a consolidated node is created using aggregation strategies that preserve essential information while reducing graph complexity.

*Class Node Creation:* Each class node contains:

- A unique class identifier preserving full namespace information.
- Aggregated numerical features computed from all methods within the class using statistical summaries (mean, sum, max, min).
- Ground truth pattern labels (in training mode) for supervised learning.

- Method counts to indicate class complexity and architectural role.
- Structural metadata on class hierarchy, interface implementation, and architectural roles.

*Feature Aggregation Strategies:* Multiple approaches capture different aspects of class behavior:

- Statistical aggregation (mean, median, std, quartiles) to describe feature distribution.
- Structural aggregation (counts of constructors, getters, setters, etc.) for structural insights.
- Behavioral aggregation analyzing method interaction patterns.
- Complexity aggregation combining individual method complexities into class-level measures.

Edge Aggregation and Relationship Modeling: Method-to-method call relations are transformed into class-to-class dependencies via structured aggregation:

*Dependency Extraction:* dependencies are extracted from

- Direct dependencies: from inter-class method calls.
- Indirect dependencies: from multi-hop paths indicating complex patterns.
- Bidirectional relationships: signaling mutual dependencies (e.g., Observer, Mediator).
- Hierarchical dependencies: derived from inheritance.

*Edge Weight Computation:* these are calculated from:

- Call frequency weights representing the intensity of class interactions.
- Method diversity weights indicating the variety of methods involved.

## C. ML Training Module (M2)

The ML training module encompasses multiple complementary approaches, each designed to capture different aspects of design pattern characteristics through specialized architectures and training strategies. Three are currently realized: 1) GNN, 2) NGDB, and 3) SVM.

*1) GNN Training:* The GNN implementation (in TrainGNN.py) is a DPD approach that leverages the structural relationships inherent in software architectures. The implementation supports multiple GNN variants, each offering different strengths for capturing various aspects of design pattern implementations.

*a) GNN Architecture Design:* A modular and configurable architecture that supports diverse GNN approaches was implemented in the Python Class GNNModel:

Supported GNN Variants: The system supports:

- Graph Convolutional Networks (GCN) provide an implementation of spectral graph convolutions that can capture local neighborhood information and hierarchical patterns characteristic of structural design patterns.
- Graph Attention Networks (GAT) offer attention-based mechanisms that dynamically weight the importance of different neighbors, which could support identifying key relationships inherent in behavioral design patterns.
- Graph Isomorphism Networks (GIN) offer the ability to distinguish between different graph structures that could be used to detect subtle variations in pattern implementations.
- Graph Sample and Aggregation (GraphSAGE) offers a scalable inductive learning approach that can generalize to unseen graph structures, which can support DPD across diverse codebases.

Configurable Architecture Parameters:

- Supports flexible specification of network capacity with hidden channels and layer depth for deep architectures that can capture complex hierarchical patterns.
- For GAT models, configurable multi-head attention provides customizable attention head counts and attention dropout rates.
- Multiple aggregation functions, including mean, max, sum, and attention-weighted approaches can be used to combine neighborhood information.
- Dropout strategies, batch normalization, and weight decay options prevent overfitting through regularization mechanisms.
- Configurable multi-layer perceptrons for final classification offer customizable hidden dimensions and activation functions.

*b) Data Loading and Preprocessing Pipeline:* The training pipeline begins with a data loading and preprocessing workflow:

Graph Data Processing: The system processes class-level graph features from JSON files through the following stages:

1. *File Access and Error Handling*: file system traversal ensures resilience to missing or corrupted files via error handling mechanisms.
2. Schema Validation: Parsed JSON content undergoes schema validation and integrity checks to guarantee consistency across heterogeneous data sources.
3. *Graph Conversion*: JSON representations are transformed into internal graph data structures, including validation of graph properties for structural correctness.
4. *Feature Extraction and Preprocessing*: Node features are extracted and preprocessed through missing value imputation, outlier detection, and normalization for subsequent learning stages.

PyTorch Geometric Integration: integration with PyTorch Geometric is implemented as a series of structured steps:

1. *Tensor Conversion*: transformation of feature matrices into GPU-compatible tensor formats for integration with PyTorch and memory-efficient representation for downstream processing.
2. *Edge Index Construction*: The construction of the edge_index structure includes error-checking to identify and handle invalid or malformed edges, ensuring graph connectivity and correctness.
3. *Label Encoding*: Label encoding is performed using scikit-learn's LabelEncoder, with support for handling unseen classes and mitigating class imbalance during supervised learning tasks.
4. *Graph Validation*: A validation step includes removal of self-loops, detection of disconnected components, and verification of structural graph properties to ensure integrity and suitability for geometric learning.

Data Quality Assurance: Data quality is addressed via multiple validation and control procedures:

- *Feature Range Validation*: Detection and handling of extreme values, infinite values, and NaN entries is performed to ensure numerical stability and consistency of the feature space.
- *Graph Connectivity Analysis*: Structural issues such as isolated nodes and disconnected components are identified via graph connectivity analysis to preserve semantic coherence of the graph.
- *Label Distribution Assessment*: The distribution of class labels is analyzed to identify class imbalance and detect severely underrepresented classes, which could negatively impact supervised learning performance.
- *Data Partitioning Strategy*: Data splits for training, validation, and testing are performed using partitioning strategies that preserve the underlying graph structure, ensuring representativeness and validity in cross-validation setups.

*c) Training Strategies:* The training implementation incorporates strategies designed to address DPD challenges:

Cross-Validation Framework: our cross-validation implementation supports multiple strategies. Configurable K-fold strategies employ stratified sampling to ensure balanced representation across folds. Evaluation strategies for smaller datasets provide validation of generalization performance through Leave-One-Out (LOO) Cross-Validation (CV). For datasets with a temporal structure, specialized validation strategies respect temporal dependencies. Graph-aware CV employs specialized partitioning strategies that account for graph structure and avoid information leakage between training and validation sets.

Applying K-fold CV to DPD settings presents unique challenges. Software projects often cannot be arbitrarily divided, as dependencies and architecture coherence must be preserved. For this reason, a special approach such as Leave-One-Project-Out Cross-Validation (LOPO-CV) is recommended [50]. Here, each project is treated as a single validation fold. The model is trained on all other projects and tested on the one left out. While LOPO-CV increases the evaluation for cross-project scenarios, it also introduces higher variance due to project heterogeneity. Normalization of features, abstraction layers, and embedding representations can help to mitigate these issues.

Optimizer Configuration and Tuning: Optimizer support includes configuration options. Advanced Adam implementations provide configurable learning rates, weight decay, and momentum parameters, proving particularly effective for graph neural network training. Classical stochastic gradient descent with momentum enables stable training on large graphs. Adaptive learning rate methods handle the sparse gradient updates common in graph neural networks through RMSprop. Sophisticated learning rate scheduling encompasses step decay, exponential decay, and cosine annealing strategies.

*d) Focal Loss Implementation for Class Imbalance:* Focal Loss was specifically implemented and adapted for DPD [51]. Our implementation with the class FocalLoss is shown in Figure 6.

```
class FocalLoss(torch.nn.Module):
    def __init__(self, alpha=None, gamma=2):
        super().__init__()
        self.gamma = gamma  # Focusing parameter
        self.alpha = alpha  # Class balancing parameter

    def forward(self, inputs, targets):
        #Computes the Negative Log-Likelihood loss ce
        ce = F.nll_loss(inputs, targets, reduction="none")
        #Convert to probability p_t
        p_t = torch.exp(-ce)
        #Loss modulated by focussing parameter gamma
        loss = (1 - p_t) ** self.gamma * ce
        #Correct imbalance in datasets
        if self.alpha is not None:
            loss = self.alpha[targets] * loss
        #Return the mean loss across all samples
        return loss.mean()
```

Figure 6. Focal Loss implementation.

This loss function provides several advantages for DPD. The $(1 - p_t)^\gamma$ term down-weights easy examples and focuses learning on difficult cases, which is particularly important for distinguishing between similar design patterns. The α parameter provides explicit class balancing to address the severe imbalance between different design patterns in typical datasets. The loss automatically adjusts focus based on prediction confidence, enabling the model to concentrate on boundary cases and ambiguous pattern implementations. The formulation provides stable gradients even with extreme class imbalance, enabling effective training on highly skewed datasets.

*e) Training Monitoring and Control:* The training implementation includes various monitoring and control mechanisms:

Early Stopping Implementation: early stopping employs multiple criteria including:

- Configurable patience parameters with sophisticated validation loss tracking,
- Multiple metrics including F1-score, precision, and recall,
- Overfitting detection through training-validation loss divergence analysis, and
- Automatic retention of the best models based on multiple criteria with comprehensive metadata.

Metrics Tracking: Real-time monitoring of training progress encompasses:

- Detailed tracking of accuracy, precision, recall, F1-scores, and custom metrics for DPD,
- Real-time confusion matrix computation and analysis to identify specific DPD challenges,
- Detailed analysis of loss components to understand model learning dynamics, and
- Learning curve tracking for detecting convergence issues and optimization problems.

*f) Model Artifact Management:* Model artifact management supports reproducibility and enables detailed analysis:

Model Serialization: Model state preservation encompasses full model state dictionaries saved in .pth format with version compatibility, architecture specifications saved as JSON for reproducibility, training hyperparameters and configuration settings, and optimizer state preservation for training resumption.

Visualization and Analysis: Visualization artifacts include Seaborn-based confusion matrix visualizations with statistical annotations, analysis of feature distributions and their relationship to pattern detection performance, and network visualizations of class-level graphs with pattern highlighting.

*2) NGDB Training Pipeline:* The NGDB training approach, implemented in the Python module TrainPatternPipeline.py is shown in Figure 7. It uses Memgraph as an NGDB to store, process, and analyze heterogeneous code graphs in combination with different GNN models. Memgraph is an in-memory, Cypher-compatible graph database that natively integrates GNN modules through its MAGE (Memgraph Advanced Graph Extensions) library [52]. This approach supports analysis by combining the NGDB with NN architectures, enabling graph-based DPD analysis of software code structures.

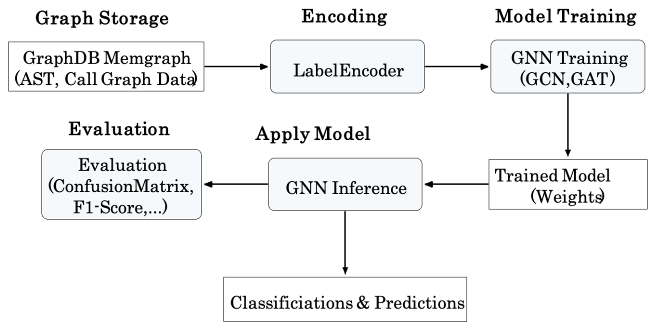
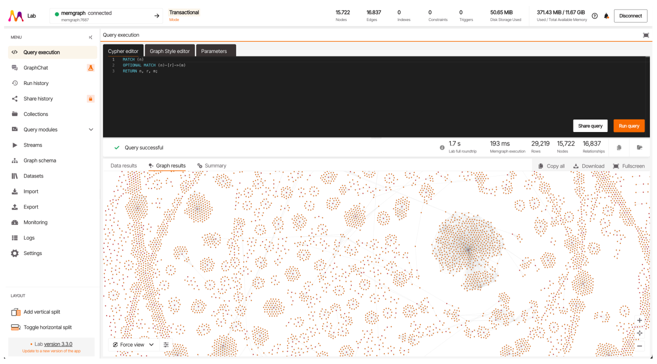


Figure 7. NGDB Training Pipeline

*a)* Graph Storage: The system employs Memgraph as the underlying graph database, chosen for its performance characteristics, Cypher Query Language (CQL) support, and integration capabilities with ML workflows. The database stores a heterogeneous graph representation of the software structure that captures multiple types of entities and relationships. A sample visualization of the CG import of the PMD project is shown in Figure 8. A closeup of the CG is shown in Figure 9.



15.722 Nodes | 16.837 Edges | 0 Indexes | 0 Constraints | 0 Triggers | 50.65 MiB Disk Storage Used | 371.43 MiB / 11.67 GiB Used / Total Available Memory

```
MATCH (n)
OPTIONAL MATCH (n)-[r]->(m)
RETURN n, r, m;
```

Query successful | 1.7 s Lab full roundtrip | 193 ms Memgraph execution | 29,219 Rows | 15,722 Nodes | 16,837 Relationships

Figure 8. Screenshot of Memgraph for entire DPD dataset showing metrics above and clustering of class and method nodes by relationships.

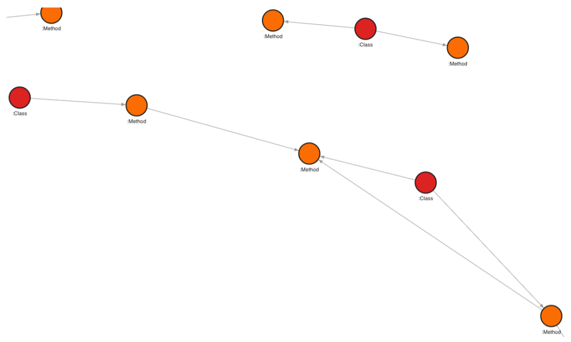


Figure 9. Closeup screenshot of Memgraph for PMD project dataset showing class (red) and method (orange) nodes and directed relationships.

Node Type Hierarchy: The graph database maintains a sophisticated node type system:

*Class Nodes:* These nodes represent software classes and are enriched with metadata including:

- Numerical feature vectors aggregated from method-level analysis,
- Pattern labels for supervised learning (available in training mode),
- Method counts and complexity metrics,
- Inheritance hierarchy information and interface implementation details,
- Package and namespace associations.

*Method Nodes:* These nodes represent individual methods and capture detailed characteristics such as:

- Method signatures and parameter types,
- Access modifiers and method-level flags (e.g., static, final),
- Cyclomatic complexity and code quality metrics,
- Call frequency, usage patterns, and invocation context,
- Exception handling structures and error management indicators.

*Package Nodes:* These nodes encapsulate software packages or modules, including:

- Hierarchical package organization and naming structure,
- Inter-package dependency relationships,
- Package-level metrics and aggregated characteristics.

Relationship Type System: The graph-based relationship model captures the semantic and structural interactions between software entities through multiple distinct relationship types:

*DECLARES Relationships:* These edges connect class nodes to their declared method nodes, encoding:

- Ownership and structural organization of methods within classes,
- Method accessibility (e.g., public, private) and scope information,
- Declaration context and compilation metadata,
- Indicators of method overrides and inheritance-related declarations.

*CALLS Relationships:* These edges represent dynamic or static method invocations and include:

- Call frequency and temporal invocation patterns,
- Parameter passing strategies and intra-procedural data flow,

- Exception propagation paths and error handling behavior,
- Conditional invocation patterns and control flow dependencies.

*EXTENDS Relationships:* These relationships model inheritance between classes, capturing:

- Inheritance depth and hierarchy complexity,
- Method override patterns and specialization,
- Abstract class associations and interface inheritance details.

*IMPLEMENTS Relationships:* These edges represent interface implementation and reflect:

- Interface compliance and contract fulfillment status,
- Patterns of multiple interface implementations,
- Usage of default methods and override behavior.

*USES Relationships:* These capture various forms of software usage dependencies, including:

- Field access and data dependency relations,
- Type usage, generic type associations, and class instantiations,
- Annotation usage and metadata-driven relationships.

*b) Data Ingestion Pipeline:* The data ingestion process implements a pipeline that loads and integrates multiple data sources into the unified graph representation:

Multi-Source Data Integration: The system systematically processes multiple JSON-based data sources, each contributing distinct structural and semantic information:

- *Graph Features Integration*: Incorporates class-level graph representations enriched with aggregated metrics and pattern labels, enabling structural learning at the class granularity.
- *Call Graph Processing*: Extracts method-level call relationships, capturing invocation frequency and contextual information to support interprocedural analysis.
- *Abstract Syntax Tree Data Integration*: Provides structural information about class hierarchies, inheritance relationships, and interface implementation through AST parsing.
- *Numerical Features Loading*: Supplies quantitative metrics for both classes and methods and offers statistical validation.

Graph Database Population: Ingestion functions manage the transformation and insertion of structured JSON nodes into the graph database, ingesting class-level data into a Memgraph instance. A Memgraph instance executes Cypher's MERGE clause to ensure idempotent insertion of class nodes based on a unique identifier. Node properties are assigned using dynamic key-value pairs passed via a props parameter. Metadata such as the ingestion timestamp and data version are appended to each node to support traceability, versioning, and data lineage.

Relationship Establishment: The system supports relationship creation between software entities with integrated validation and error handling mechanisms:

- *Inheritance Relationships*: Processing of EXTENDS clauses includes validation of multiple inheritance constraints and semantic correctness within the class hierarchy.
- *Method Declarations*: Methods are associated with their declaring classes through validated DECLARES relationships, incorporating access control checks and declaration context verification.
- *Call Relationships*: CALLS relationships are created between method nodes, enriched with invocation context, frequency metadata, and control flow annotations.
- *Interface Implementations*: IMPLEMENTS edges are formed based on parsed IMPLEMEMTS clauses, including validation of interface compliance and contract fulfillment semantics.

Data Consistency and Validation: To ensure semantic and structural correctness, the system performs validation procedures throughout the ingestion process:

- *Referential Integrity*: All referenced nodes and relationships are checked for existence to maintain consistency across the graph.
- *Schema Compliance*: Nodes and relationships are validated against predefined schema constraints to enforce type and structure conformity.
- *Data Quality Checks*: Inconsistencies, corrupted entries, and malformed properties are detected and handled using systematic quality control measures.
- *Duplicate Detection*: Redundant entities and relationships are identified and resolved through deduplication mechanisms to prevent semantic ambiguity.

*c) Graph-Derived Feature Computation:* the NGDB approach enables the computation of graph-derived features using Cypher queries:

Cypher Query Implementation: The system utilizes Cypher queries to derive composite structural metrics directly from the graph database. The query below (Figure 10) extends each Class node with additional structural and interaction-based features, enabling enhanced downstream analysis and learning.

```
mg.execute(
    """
    MATCH (c:Class)
    WHERE c.features IS NOT NULL AND c.pattern IS NOT NULL
    OPTIONAL MATCH
       (c)-[:DECLARES]->(m:Method)-[:CALLS]->(other_m:Method)
               <-[:DECLARES]-(other_c:Class)
    WITH c, collect(DISTINCT other_c.id) as
       called_classes_ids
    OPTIONAL MATCH (sub:Class)-[:EXTENDS]->(c)
    WITH c, called_classes_ids, count(DISTINCT sub) as
       in_degree_inheritance
    OPTIONAL MATCH (c)-[:EXTENDS]->(super:Class)
    WITH c, called_classes_ids, in_degree_inheritance,
         count(DISTINCT super) as out_degree_inheritance
    OPTIONAL MATCH (c)-[:DECLARES]->(meth:Method)
    WITH c, called_classes_ids, in_degree_inheritance,
       out_degree_inheritance,
         count(DISTINCT meth) as num_methods
    SET c.extended_features = c.features + [
        in_degree_inheritance, out_degree_inheritance,
        num_methods, size(called_classes_ids)
    ]
    """
)
```

Figure 10. A CQL query extends each Class node with additional structural and interaction-based features.

Specifically, the query proceeds in several stages:

1. *Filtering Relevant Classes*: Only classes for which both feature vectors (features) and pattern labels (pattern) are present are included in the analysis.
2. *CG Expansion*: The query traverses DECLARES and CALLS relationships to collect identifiers of all classes indirectly referenced via method calls, capturing inter-class interaction patterns (called_classes_ids).
3. *Inheritance Analysis (In-degree)*: By matching EXTENDS relationships pointing to the current class, the number of direct subclasses (i.e., inheritance in-degree) is computed.
4. *Inheritance Analysis (Out-degree)*: The number of direct superclasses extended by the current class (i.e., inheritance out-degree) is also calculated, supporting analysis of hierarchical complexity.
5. *Method Aggregation*: The number of methods declared by each class is counted using the DECLARES relationship to quantify class-level behavioral encapsulation.

- *Feature Augmentation*: A new property extended_features is appended to each class node. It combines the existing features vector with the four newly computed metrics: Inheritance in-degree, Inheritance out-degree, Number of declared methods, and Number of referenced external classes via method calls.

Following the execution of the Cypher query, the system augments each Class node with enriched architectural descriptors that enable a deeper analysis of software design structures and interaction patterns. These derived feature categories are extracted or inferred from graph-topological and semantic relationships:

Inheritance Metrics: captures the hierarchical properties of the class design, including:

- *In-degree inheritance*: The number of classes that extend a given class, indicating its centrality as a base class or abstract interface.
- *Out-degree inheritance*: The number of direct superclass relationships a class possesses, indicating hierarchy depth and potential misuse of multiple inheritance.
- *Inheritance tree depth*: The maximal depth from the root of the inheritance chain, used to detect deep or overly complex hierarchies.
- *Interface implementation count*: The number of interfaces implemented by a class, reflecting its abstraction adherence and flexibility.

Method Declaration Patterns: describes the intra-class behavioral structure:

- *Total method count per class:* providing a proxy for behavioral richness.
- *Distribution of method types:* distinguishing between constructors, accessors, mutators, and core logic methods.
- *Access modifier patterns:* such as the public-to-private method ratio, which may suggest encapsulation quality.
- *Abstract method statistics:* relevant for identifying abstract base classes or template patterns.

Inter-Class Communication: Encodes communication behavior in the CG:

- *Outgoing communication count*: i.e., the number of distinct classes this class calls.
- *Incoming communication count:* i.e., the number of classes calling this class.
- *Communication intensity and frequency:* providing insight into dependencies and possible code smells.
- *Bidirectional call detection:* to identify tight coupling or cyclic dependencies.

Architectural Complexity: Captures structural roles and architectural health:

- *Dependency fan-in and fan-out metrics:* used for impact and stability analysis.
- *Coupling and cohesion indicators:* inferred from communication and method sharing patterns.
- *Architectural layer classification:* by analyzing depth and connection types.
- *Pattern-specific structural indicators:* such as those associated with Singleton or Factory design patterns.

As shown in the CQL query above, a feature vector enhancement process is finally applied:

Feature Concatenation: The derived graph-based features are concatenated with the original static numerical features extracted from source code. Normalization and scaling are applied to ensure numerical stability.

Dimensionality Management: Strategies such as dimensionality capping, regularization, or feature selection are employed to prevent overfitting and ensure computational tractability.

Feature Correlation Analysis: Correlation matrices and mutual information metrics are used to detect and eliminate redundant or collinear features.

Quality Validation: The resulting feature vectors are validated for completeness, consistency, and compatibility with downstream models.

This multi-level feature integration enables the model to capture a rich representation of both static and relational program semantics, supporting tasks such as design pattern classification, anomaly detection, and architecture recommendation.

*d) Class Imbalance Mitigation Strategies:* Our DPD-NGDB approach integrates the following techniques to address the inherent class imbalance and memory constraints associated with design pattern datasets.

Minority Class Oversampling is used to mitigate the severe class imbalance in supervised learning tasks. The system employs graph-specific oversampling strategies that preserve structural integrity:

- *Configurable Oversampling Factor*: Typically, 2x oversampling with configurable parameters based on class distribution analysis
- *Graph-Aware Sampling*: Oversampling strategies that preserve graph structure and neighborhood characteristics
- *Synthetic Graph Generation*: Advanced techniques for generating synthetic graph samples that maintain pattern characteristics
- *Stratified Sampling*: Ensuring representative sampling across different pattern types and complexity levels

Memory Management and Scalability is used to support high-volume graph data. The system incorporates memory management mechanisms and scalable processing techniques:

- *Batch Processing:* Intelligent batching strategies to prevent out-of-memory errors during training.
- *Memory Monitoring:* Real-time memory usage monitoring using psutil with automatic garbage collection.
- *Incremental Processing:* Support for incremental processing of large graphs with checkpoint recovery.
- *Resource Optimization:* Dynamic resource allocation and optimization based on available system resources.

*e) Heterogeneous Graph Neural Network Architecture (HeteroGNN):* this constitutes a specialized neural architecture engineered to process and learn from heterogeneous graph data.

Multi-Head Attention Implementation: This is used to model the diverse semantic relationships in heterogeneous graphs. The architecture leverages advanced Graph Attention Convolution (GATConv) layers with the following design considerations:

- *Attention Head Configuration*: Multiple attention heads (typically 4-8) for capturing diverse relationship patterns.
- *Edge-Type-Specific Attention*: Specialized attention mechanisms for different relationship types (DECLARES, CALLS, EXTENDS).
- *Dynamic Attention Weighting*: Adaptive attention weights that adjust based on graph structure and the learning progress.
- *Attention Dropout*: Dropout strategies (typically 0.5) for regularization and generalization.

Message Passing for Heterogeneous Structures: The message passing paradigm is extended to accommodate structural heterogeneity, enabling effective information flow across diverse node and edge types:

- *Type-Specific Message Functions*: Different message computation functions for each relationship type.
- *Hierarchical Message Aggregation*: Multi-level aggregation strategies that capture both local and global graph patterns.
- *Temporal Message Patterns*: Support for temporal relationship patterns where available.
- *Bidirectional Message Passing*: Advanced bidirectional message passing for capturing complex dependency patterns

*f) Advanced Training Configuration:* To ensure model convergence, a multi-faceted training strategy is employed. This includes configured optimizers, loss functions tailored to class imbalance, and long-term training protocols with adaptive control mechanisms.

Optimizer Configuration: The optimization process is guided by adaptive strategies that enhance convergence stability and learning performance:

- *Adam Optimizer:* Advanced Adam implementation with learning rate 0.001 and adaptive moment estimation.
- *Learning Rate Scheduling*: ReduceLROnPlateau scheduler with sophisticated plateau detection and learning rate adaptation.
- *Gradient Clipping*: Advanced gradient clipping strategies to prevent gradient explosion in deep graph networks.
- *Weight Initialization*: Weight matrices are initialized using graph-aware strategies.

Loss Function Integration: To address severe class imbalance and optimize learning for minority classes, a sophisticated loss configuration is applied:

- *Focal Loss Implementation*: Gamma parameter of 2 for hard example mining with adaptive class weighting.
- *Balanced Class Weights*: Dynamic class weight computation based on current class distributions.
- *Loss Regularization*: Additional regularization terms for graph structure preservation.
- *Multi-Task Learning*: Support for auxiliary tasks that improve pattern detection performance

Extended Training Protocol: A training loop with fault-tolerance and performance tracking ensures sustainable learning across extended sessions:

- *Extended Epoch Training:* Up to 1000 epochs with careful overfitting monitoring.
- *Early Stopping Strategy:* Patience of 50 epochs with multiple stopping criteria.
- *Model Checkpointing:* Regular model checkpointing with best model preservation.
- *Training Resumption:* Support for training interruption and resumption with full state preservation.

*g) Evaluation and Analysis:* The evaluation framework enables an analysis of model performance through both metric-based assessment and visualization-based interpretability.

Multi-Metric Evaluation: A set of performance metrics is employed to account for the challenges posed by class imbalance and varying pattern complexities:

- *Weighted Metrics*: Class-weighted accuracy, precision, recall, and F1-scores to handle class imbalance.
- *Per-Class Analysis*: Detailed per-class performance metrics for identifying pattern-specific detection capabilities.
- *Confusion Matrix Analysis*: Confusion matrix analysis with statistical significance testing.
- *Confidence Score Analysis*: Distribution analysis of prediction confidence scores.

Visualization: The framework integrates visualizations to support interpretability and diagnostic analysis:

- *Confusion Matrix Heatmaps:* Confusion matrix visualizations with statistical annotations are offered
- *Graph Structure Visualization:* Network visualizations of learned graph representations can be shown.

*3) SVM Integration:* The Support Vector Machine implementation TrainSVM.py provides a complementary approach to the graph-based methods via high-dimensional feature space analysis. It focuses on leveraging rich numerical feature representations for pattern classification.

## *D. DPD and Ensembles Module (M3)*

The DPD and Ensembles Module implements the ensemble technique combining predictions from multiple trained models, or if desired utilizing a single model. It currently applies soft voting and confidence-based weighting, but can apply any other EM technique.

*4) Architecture:* a stacked ensemble learning technique is applied in order to improve predictive performance by combining multiple base learners through a higher-level model known as a meta-classifier. This can be effective when the individual base models capture different aspects of the data or exhibit complementary strengths and weaknesses. The Detection.py module implements the stacked ensemble learning technique for DPD.

The process begins with a set of base models, which are independently trained on the same training data. Each model generates an individual model detection (i.e., predictions) for the input instances, resulting in a vector of intermediate outputs. These predictions are then passed to a meta-classifier, which is trained to learn a higher-order decision function based on the outputs of the base models (currently soft voting is utilized, but this can be adjusted). The meta-classifier effectively integrates the diverse perspectives of the underlying models and produces an ensemble-based detection (i.e., ensemble prediction) $P_f$ with generalization capabilities.

*5) Individual Model Detection:* The detection system provides unified interfaces for applying trained models. The pipeline orchestrates: numerical feature extraction, standardization using pre-computed parameters, and graph feature merging. The following individual models are used:

SVM Detection: the function loads standardized features, applies the trained model, and handles edge cases (e.g., zero-feature vectors assigned "No Pattern", low-confidence predictions marked "Unknown").

GNN Detection: the function converts class-level graph features to PyTorch Geometric Data objects and applies trained GNN models.

NGDB Detection: the function leverages Memgraph for real-time graph queries and applies trained HeteroGNN models.

*6) Ensemble-Based Detection:* The ensemble methodology combines multi-model predictions (our current implementation utilizes SVM, GNN, and NGDB models via soft voting), while addressing different label spaces through label alignment. The ensemble process is as follows:

i. Common Class Identification: Finds classes that exist in all model outputs.
ii. Label Space Unification: Creates a unified label space containing all unique pattern classes.
iii. Probability Alignment: Maps each model's outputs to the unified space.
iv. Apply ensemble technique: Various techniques can be applied to combine the predictions of the base models, include stacking, bagging, boosting, etc. Soft voting is currently used.
v. Post-processing: Applies confidence thresholding and special-case handling. A function generates aligned probability matrices in which each row corresponds to a class instance and each column to a specific design pattern type. This alignment ensures that the probabilistic outputs from different models are directly comparable. Soft voting prevents bias toward patterns in a subset of models.

## *E. System Integration and API Module (M4)*

*7) Web API:* The complete system is orchestrated through a FastAPI backend, providing RESTful endpoints. The system provides several key endpoints, including:

- POST /process-features for complete feature processing pipeline,
- POST /train-gnn, /train-pattern, and /train-svm for model training,
- POST /start-detection for model inference with ensemble support, and
- GET /get-metrics for model evaluation data.

The API includes error handling, asynchronous processing support, and Cross-Origin Resource Sharing (CORS) middleware for frontend integration.

*8) Robustness and Integrity:* Robustness, reproducibility, and overall system integrity are addressed via several supporting mechanisms in the pipeline. These include:

- Error Handling: includes file I/O validation, memory management - particularly for graph processing, and proper cross-validation data partitioning.
- Configuration Management: All configurations are serialized as JSON, including hyperparameters, training settings, and timing information to ensure reproducibility.
- Validation Framework: Multi-stage validation encompasses feature extraction consistency checks, standardization validation, and model validation with convergence analysis and overfitting detection.
- Feature Consistency: Validation of feature consistency across the graph includes range checks for feature values and detection of outliers, consistency validation between different feature sources, missing feature detection and imputation strategies, and feature distribution analysis with normalization validation.
- Label Quality: In training mode, label quality assurance encompasses label consistency validation across related classes, detection of potentially mislabeled instances, analysis of label distribution and balance, and validation of ground truth quality and completeness.

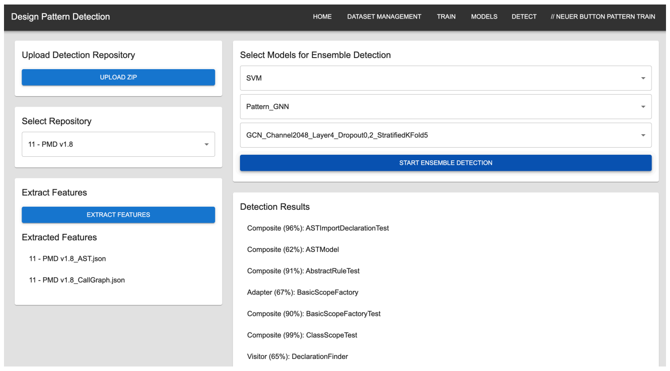
*9) Output Generation and Format Optimization:* The output generation process creates JSON files containing the class-level graph structure and essential metadata. This stage ensures that both the structural and contextual information are retained and formatted for efficient downstream use. Key aspects include:

Format Optimization: The JSON output is optimized for downstream processing through efficient encoding of graph structures for fast loading, preservation of metadata for debugging and analysis, compatibility with PyTorch Geometric data formats, and support for incremental loading and processing of large graphs.

Metadata Preservation: Metadata is preserved to support analysis and debugging, including original method-level information for traceability, transformation parameters and configuration settings, quality metrics and validation results, and processing timestamps with version information.

The resulting output enables efficient graph-based ML operations while maintaining the semantic relationships present in the original code structure, creating a foundation for DPD using GNN and other techniques.

*10) User Interface (UI):* The frontend is implemented via Node.js with React, Vite.js, the Material UI React component, and Axios for Web APIs. Once a DPD repository is uploaded as a zip, it can be selected via dropdown (here PMD), as shown on the left in Figure 11. Thereafter, extract features can be executed (left bottom). Then the various ensemble models can be selected on the right, and then the ensemble detection can be started. At the top, the menu offers dataset management, training, models, and detection management.



11 - PMD v1.8

Extracted Features

11 - PMD v1.8_AST.json

11 - PMD v1.8_CallGraph.json

SVM

Pattern_GNN

GCN_Channel2048_Layer4_Dropout0,2_StratifiedKFold5

Detection Results

```
Composite (96%): ASTImportDeclarationTest
Composite (62%): ASTModel
Composite (91%): AbstractRuleTest
Adapter (67%): BasicScopeFactory
Composite (90%): BasicScopeFactoryTest
Composite (99%): ClassScopeTest
Visitor (65%): DeclarationFinder
```

Figure 11. DPD user interface.

## VI. Evaluation

The evaluation focuses on the effectiveness of our DPD-NGDB approach, analyzing GNN variants and DPD performance.

As described previously, our DPD approach is structured to inherently support ensemble methods for maximum flexibility. However, currently, all other models we attempted (GNN, SVM) for the ensemble in preliminary evaluations performed far worse or lacked consistency, and thus did not complement DPD-NGDB. Hence, our DPD-EM results are not evaluated here. As EM depends on the other models improving results, future work will focus on finding and tuning alternative models such that they address the misclassifications found in our DPD-NGDB approach.

All experiments were conducted on an Apple MacBook with M2 Pro with 32GB RAM.

### A. Dataset Description

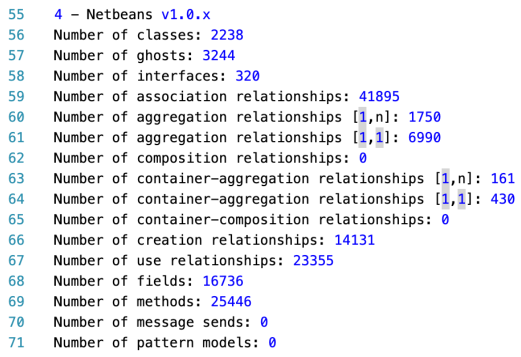
Both the training and evaluation are conducted on a dataset consisting exclusively of 23 canonical GoF design patterns and other non-labeled code. The dataset utilizes 9 projects from the Pattern-like Micro-Architecture Repository (P-MARt) repository [7]. As not all 23 GoF design patterns were exemplified in P-MARt, the dataset was supplemented with 23 pattern implementation examples from Refactoring Guru [53], which, while isolated examples without a larger project context, at least provide some training data.

The evaluation focuses on classifying any single class as a pattern instance to emphasize discriminative performance among the design patterns, and unlabeled classes are assumed to be "No Pattern" or unknown. Note that for a single pattern, multiple classes may participate (e.g., Observer), while a single class might participate in or utilize multiple patterns simultaneously (e.g., Factory and Observer). For simplicity and the labeled ground truth basis, classes are assumed to be associated with only a single pattern (e.g., either Factory Method or Observer, not both). Thus, the actual number of static pattern instantiations could be far less than the number of classes identified (labeled) as participating in a certain pattern (e.g., Observer, Broker).

The dataset comprises 30 projects with 417 unique samples across the 23 patterns, with varying distribution across pattern categories (121 creational, 182 structural, and 114 behavioral) as shown in TABLE I. Hence, the creational, structural, and behavioral design pattern types are included in the training and detection dataset with varying degrees of pattern sample frequency.

TABLE I. GoF Dataset

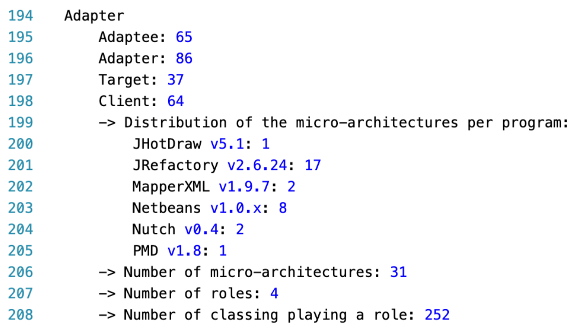
| Pattern | Samples |
|---|---|
| ***Creational*** | **121** |
| Abstract Factory | 72 |
| Builder | 28 |
| Factory Method | 8 |
| Singleton | 12 |
| Prototype | 1 |
| ***Structural*** | **182** |
| Adapter | 62 |
| Bridge | 28 |
| Composite | 58 |
| Decorator | 8 |
| Facade | 17 |
| Flyweight | 4 |
| Proxy | 5 |
| ***Behavioral*** | **114** |
| Chain of Responsibility | 5 |
| Command | 16 |
| Interpreter | 5 |
| Iterator | 19 |
| Mediator | 9 |
| Memento | 12 |
| Observer | 20 |
| State | 6 |
| Strategy | 7 |
| Template Method | 2 |
| Visitor | 13 |
| **Total** | **417** |



```
4 - Netbeans v1.0.x
Number of classes: 2238
Number of ghosts: 3244
Number of interfaces: 320
Number of association relationships: 41895
Number of aggregation relationships [1,n]: 1750
Number of aggregation relationships [1,1]: 6990
Number of composition relationships: 0
Number of container-aggregation relationships [1,n]: 161
Number of container-aggregation relationships [1,1]: 430
Number of container-composition relationships: 0
Number of creation relationships: 14131
Number of use relationships: 23355
Number of fields: 16736
Number of methods: 25446
Number of message sends: 0
Number of pattern models: 0
```

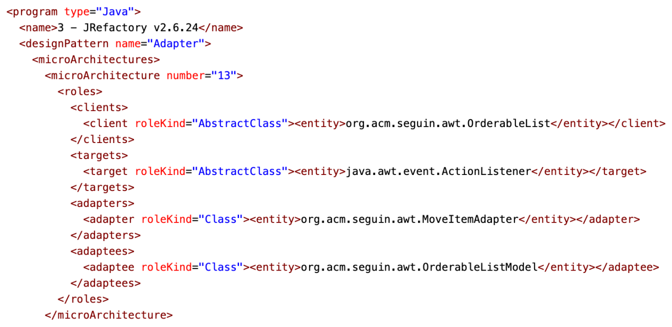
Figure 12. Snippet of P-MARt Netbeans project summary metrics.

An example of the pattern summary metrics per project in P-MARt is shown in Figure 12. Pattern-specific metrics are also summarized in XML as shown in Figure 13. The XML-based documentation describes on a per-project basis the classes involved in a micro-architecture (pattern), as shown in Figure 14. We extracted the information per project to use for class labeling as our ground truth as shown in Figure 15.



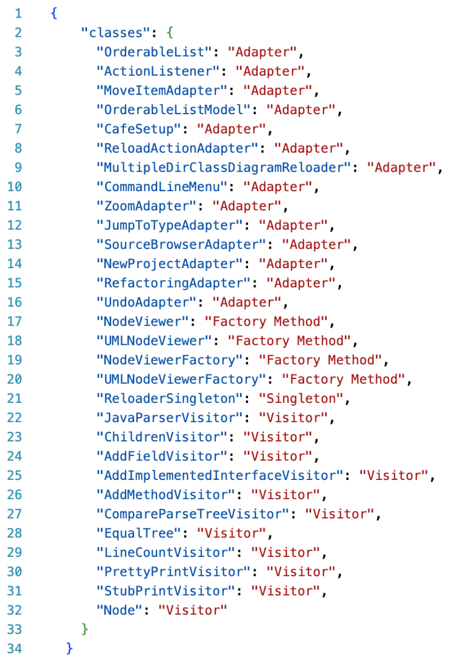
```
Adapter
    Adaptee: 65
    Adapter: 86
    Target: 37
    Client: 64
    -> Distribution of the micro-architectures per program:
        JHotDraw v5.1: 1
        JRefactory v2.6.24: 17
        MapperXML v1.9.7: 2
        Netbeans v1.0.x: 8
        Nutch v0.4: 2
        PMD v1.8: 1
    -> Number of micro-architectures: 31
    -> Number of roles: 4
    -> Number of classing playing a role: 252
```

Figure 13. Snippet of P-MARt Adapter pattern summary.



```
<program type="Java">
  <name>3 - JRefactory v2.6.24</name>
  <designPattern name="Adapter">
    <microArchitectures>
      <microArchitecture number="13">
        <roles>
          <clients>
            <client roleKind="AbstractClass"><entity>org.acm.seguin.awt.OrderableList</entity></client>
          </clients>
          <targets>
            <target roleKind="AbstractClass"><entity>java.awt.event.ActionListener</entity></target>
          </targets>
          <adapters>
            <adapter roleKind="Class"><entity>org.acm.seguin.awt.MoveItemAdapter</entity></adapter>
          </adapters>
          <adaptees>
            <adaptee roleKind="Class"><entity>org.acm.seguin.awt.OrderableListModel</entity></adaptee>
          </adaptees>
        </roles>
      </microArchitecture>
```

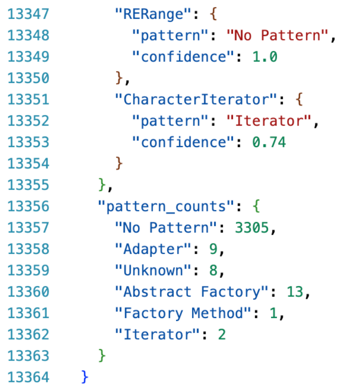
Figure 14. Snippet of XML-based P-MARt [7] documentation of Adapter pattern in JRefactory project involving the MoveItemAdapter class.



```
{
    "classes": {
      "OrderableList": "Adapter",
      "ActionListener": "Adapter",
      "MoveItemAdapter": "Adapter",
      "OrderableListModel": "Adapter",
      "CafeSetup": "Adapter",
      "ReloadActionAdapter": "Adapter",
      "MultipleDirClassDiagramReloader": "Adapter",
      "CommandLineMenu": "Adapter",
      "ZoomAdapter": "Adapter",
      "JumpToTypeAdapter": "Adapter",
      "SourceBrowserAdapter": "Adapter",
      "NewProjectAdapter": "Adapter",
      "RefactoringAdapter": "Adapter",
      "UndoAdapter": "Adapter",
      "NodeViewer": "Factory Method",
      "UMLNodeViewer": "Factory Method",
      "NodeViewerFactory": "Factory Method",
      "UMLNodeViewerFactory": "Factory Method",
      "ReloaderSingleton": "Singleton",
      "JavaParserVisitor": "Visitor",
      "ChildrenVisitor": "Visitor",
      "AddFieldVisitor": "Visitor",
      "AddImplementedInterfaceVisitor": "Visitor",
      "AddMethodVisitor": "Visitor",
      "CompareParseTreeVisitor": "Visitor",
      "EqualTree": "Visitor",
      "LineCountVisitor": "Visitor",
      "PrettyPrintVisitor": "Visitor",
      "StubPrintVisitor": "Visitor",
      "Node": "Visitor"
    }
  }
```

Figure 15. Snippet of our extracted JSON labels for the JRefactory project labeling MoveItemAdapter as class participating in Adapter pattern.

DPD results are output in our JSON format per project listing all classes, the primary pattern detected, a confidence value, and a project summary, as shown in Figure 16.



```
      "RERange": {
        "pattern": "No Pattern",
        "confidence": 1.0
      },
      "CharacterIterator": {
        "pattern": "Iterator",
        "confidence": 0.74
      }
    },
    "pattern_counts": {
      "No Pattern": 3305,
      "Adapter": 9,
      "Unknown": 8,
      "Abstract Factory": 13,
      "Factory Method": 1,
      "Iterator": 2
    }
  }
```

Figure 16. Snippet of our detection results per class and project summary.

Class imbalance across DPs was addressed via oversampling during training to ensure balanced representation. K-Fold cross-validation (K=5) is used for internal validation, with stratification to ensure balanced folds across the 23 pattern types.

## *B. NGDB GNN Variant Evaluation*

The NGDB GNN model, implemented using Memgraph, integrates heterogeneous graph representations with GNN inference for real-time pattern detection, focusing on graph-derived features (e.g., inheritance metrics, call patterns) computed via Cypher queries. NGDB GNN variant performance was evaluated using Accuracy, Precision, Recall, and F1-Score. F1-Score was used due to dataset imbalance across pattern types. The overall performance across all patterns for each variant, averaged over the 5-fold CV is summarized in TABLE II.

TABLE II. NGDB GNN VARIANT PERFORMANCE (STRATIFIED 5-FOLD CV)

| Model | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) |
|---|---|---|---|---|
| GCN | 93.45 | 90.67 | 93.45 | 91.13 |
| GAT | 97.72 | 97.73 | 97.72 | 97.68 |
| GINSAGE | 97.66 | 97.61 | 97.66 | 97.44 |

GAT achieved the highest F1-Score (97.68%) and precision (97.73%), followed closely by GINSAGE (F1: 97.44%, precision: 97.61%). GCN performed slightly lower, with an F1-Score of 91.13%. These results show substantial improvement over the standard GNN variants, meeting or exceeding the target F1-Score of 0.80 (80%) specified in the non-functional requirements (NFR3).

Pattern-specific performance from the classification reports shows high F1-scores for most patterns, such as Singleton (1.0 across variants), Proxy (1.0), and Chain of Responsibility (1.0). Abstract Factory achieved strong scores (GAT: 0.86, GINSAGE: 0.80, GCN: 0.15), while Adapter had lower performance in GCN (0.0) but improved in GAT (0.65) and GINSAGE (0.61). Behavioral patterns like Observer (GAT: 0.86, GINSAGE: 0.67, GCN: 0.0) and Command

(GAT: 0.75, GINSAGE: 0.86, GCN: 0.22) showed variability, reflecting challenges in capturing dynamic interactions.

Analysis: The results indicate that GAT outperforms other NGDB variants, likely due to its attention mechanism effectively weighting graph-derived features like call patterns and inheritance degrees. GINSAGE also performed well, combining isomorphism testing with inductive learning for robust generalization across diverse pattern implementations. GCN, while solid, lagged slightly, possibly due to less sophisticated aggregation in heterogeneous graphs, which limited its ability to capture complex relationships in patterns like Observer (F1: 0.0).

The high overall F1-Scores (91.13% – 97.68%) demonstrate the value of NGDB's Cypher-derived features in capturing relational aspects missed by standard GNNs, particularly for behavioral patterns. For instance, Observer achieved an F1-Score of 0.86 in GAT, reflecting the benefit of inter-class communication metrics. Despite excluding 'No Pattern' instances during training, the evaluation metrics reflect strong discrimination among the 23 patterns, with oversampling mitigating imbalance effectively.

Confidence metrics show reduced overconfidence compared to standard GNNs, with wrong predictions having notably lower scores (e.g., GAT: 64.79% for wrong vs. 97.16% for correct). Patterns like Singleton, Proxy, and Chain of Responsibility achieved perfect F1-scores (1.0), benefiting from distinct structural signatures enhanced by NGDB features, such as inheritance hierarchies and method declarations. Lower performance on patterns like Command (GAT: 0.75, GINSAGE: 0.86) and Adapter (GAT: 0.65) may stem from variability in implementations or insufficient feature representation for nuanced behavioral interactions.

## C. DPD Evaluation of DPD-NGDB

As GINSAGE and GAT performance was equivalent, DPD-NGDB utilized GINSAGE for the rest of the evaluation.

*1) DPD Performance:* The confusion matrix for the DPD-NGDB model across the entire dataset using 5-fold CV demonstrates strong discrimination across the 23 GoF patterns, as shown in Figure 17. The matrix reveals minimal misclassification, with most patterns correctly identified along the diagonal. Analysis of misclassifications requires a case-by-case deeper analysis as explained in the following discussion.

Project-specific DPD tests were performed on the various P-MARt projects on which it had been trained (except for Netbeans) as listed in TABLE IV. The DPD results for the projects in the training showed an accuracy range from 0.17 to .91, precision from 0.83 to 1, recall from 0.20 to 0.91, and F1 scores from 0.55 to 0.94.

When including the Netbeans test project (left out of training), the overall values were 0.45 for accuracy, 0.96 for precision, 0.58 for recall, and 0.72 for F1. Each project has a diverse set of patterns which were detected, as shown on the right of the table, covering 18 of the 23 GoF. The testing of the remaining 5 patterns was performed in the 5-fold CV and is shown in the confusion matrix. In addition, a confidence diagram for the 23 GoF patterns (including no-pattern and unknown) is shown in Figure 18. It shows that confidence values rarely go below 0.5, and that many or most of the misclassifications occur below complete confidence and have to do with determining a class is not involved in a pattern (No pattern). This determination can be difficult even for seasoned software engineers.

*2) Leave-One-Project-Out Cross-Validation (LOPO-CV):* The Netbeans project was left out of the training set which used 5-fold CV. Withholding Netbeans for testing evaluates DPD for unseen test data, with the results shown in TABLE III. The F1 score of 0.17 shows relatively poor performance over the four patterns with an overall accuracy of 0.03. This shows potential issues with the generalization of our DPD results as described in the following discussion, and our relatively small training dataset may be a factor.

TABLE III. NETBEANS TEST RESULT (LOPO-CV)

| Pattern | Classes | TP | FP | FN | Precision | Recall | F1 |
|---|---|---|---|---|---|---|---|
| Abstract Factory | 171 | 1 | 3 | 170 | 0.25 | 0.01 | 0.01 |
| Adapter | 81 | 7 | 3 | 74 | 0.70 | 0.09 | 0.15 |
| Command | 3 | 1 | 2 | 2 | 0.33 | 0.33 | 0.33 |
| Iterator | 12 | 0 | 1 | 12 | 0 | 0 | - |
| Overall | 267 | 9 | 9 | 258 | 0.32 | 0.11 | 0.17 |

*3) Feature Importance:* The SHAP values can provide insight into which features contributed the most to the model predictions, and is shown in Figure 19. The minimum number of unique instantiations of a class was the strongest, followed by the mean number of outgoing calls, followed by the number of fields in the class. This was followed by a number of call-related metrics. This may be an indication that the graph structure together with these metrics provide influence the ability to distinguish the various patterns.

## D. Discussion

*1) NGDB GNN variants evaluation:* patterns like Singleton, Proxy, and Chain of Responsibility were detected most reliably, achieving perfect F1-scores (1.0) across NGDB variants. This is likely due to their distinct structural signatures such as private constructors and static access methods for Singleton. These features are well-captured by graph-derived metrics like inheritance relationships and call patterns computed via Cypher queries in Memgraph. Structural patterns like Composite and Abstract Factory also showed strong performance in NGDB (e.g., GAT F1: 0.86 for Abstract Factory), benefiting from the heterogeneous graph representations that emphasize class-to-class dependencies and method aggregations. In contrast, behavioral patterns such as Observer and Command exhibited more variability, with lower F1-scores in GCN (0.0 for Observer, 0.22 for Command) but improvements in GAT (0.86 for Observer, 0.75 for Command) and GINSAGE (0.67 for Observer, 0.86 for Command). This suggests that attention mechanisms in GAT and inductive learning in GINSAGE better handle dynamic interactions, though the static nature of the analysis limits full capture of runtime behaviors. The standard GNN variants (GCN, GAT, GINSAGE) performed poorly overall (F1: 3.54% – 7.19%), with GINSAGE slightly outperforming others due to its ability to generalize to unseen graphs, but still struggling with imbalance and feature noise.

In contrast, standard non-NGDB GNN variants (GCN, GAT, GIN, GraphSAGE) yielded poor performance far below the target, due to dataset imbalance and limited feature representation. Thus, the ensemble method, combining SVM, GNN, and NGDB via soft voting, underperformed compared to the NGDB base model alone, primarily due to the suboptimal results of GNN and SVM, which introduced noise and biased predictions. This validates the hypothesis that ensemble methods can enhance robustness only when base models are sufficiently strong, highlighting NGDB's critical role in achieving high accuracy.

*2) Ensemble Methods evaluation:* The methodology successfully extended the ensemble method, intended to combine SVM's feature separation, GNN's structural resilience, and NGDB's graph-derived insights, but underperformed relative to the standalone NGDB approach. It was thus not included in this evaluation. This highlights a key strength of ensembles in theory – leveraging complementary models to reduce variance and improve robustness – but in practice, the weak base models (perhaps due to the sparse training datasets) diluted NGDB's high performance. Further multi-model investigation and fine-tuning of ensembles is included in future work, as we believe it to hold promise for addressing NGDB misclassifications once their causes are apparent.

*3) Requirements coverage:*

- FR1: All 23 GoF DPs were detected across 9 open-source Java projects with documented DPs from the P-MARt dataset with Refactoring Guru implementations, which supporting class-level labeling and benchmarking.
- FR2: Multi-modal features including numerical, graph-based (ASTs, CGs), and derived structural metrics were extracted from the source code. from the base implementation and incorporating multi-modal features (numerical, graph-based, structural).
- FR3: Training pipelines for individual base models (e.g., SVM, GNN, NGDB) and an ensemble combiner (e.g., soft voting) were implemented.
- FR4: The P-MARt dataset was extended with additional patterns to cover all GoF patterns, class-level labeling was applied, and imbalance handling was addressed via oversampling.
- FR5: An evaluation framework was implemented that offers cross-validation (K-Fold, Leave-One-Project-Out (LOPO)) and calculates performance metrics (F1-Score, confidence). The use of stratified K-Fold (K=5) cross-validation supports the generalization of the approach, though challenges like dataset imbalance and implementation variability persisted.
- FR6: The pipelines were implemented to support batch processing of multiple repositories and real-time pattern detection for integration into development workflows was enabled via memory monitoring.
- NR1: DPD model execution performance was reasonable, within minutes on standard hardware for medium-size projects. Model training involved more time and resources but was within reasonable expectations.
- NR2: Scalability was supported via memory and resource management and optimization, e.g., via batching strategies that account for resource limitations (avoiding out-of-memory errors during training), incremental processing of large graphs with checkpoint recovery, etc.
- NR3: While F1-scores above the target of 0.80 were achieved for various P-MARt projects, the overall score was 0.72 for the GoF spectrum on over 5300 classes; robustness against code variations for unseen projects needs further work, exemplified with the large 3347 class Netbeans project that had 267 classes participating in four different DP types with a resulting F1 score of 0.17.
- NF4: Extensibility was achieved via a modular architecture that allows for the automatic addition of new pattern types, automated feature extractors, and a flexible ensemble inclusion of further base models.
- NF5: Reproducibility was addressed via logging of parameters (including random seed management) and versioning of models, datasets, and intermediate and final output results.

*4) Summary:* The DPD-NGDB evaluation showed that it is feasible to automatically train DPD-NGDB on a labeled training set of the 23 GoF patterns, and for it to detect the spectrum of patterns when it comes across these again. Overall, the results validate the use of NGDB for real-time, relational pattern detection, aligning with functional requirements for covering all 23 patterns and achieving high accuracy. The AST and CG features proved resilient for the different pattern types (creational, structural, and behavioral).

As to limitations, as encountered with LOPO-CV using Netbeans, DPD on unseen datasets can be challenging and further investigation regarding misclassification factors and testing on diverse datasets is needed. The relatively sparse training set and static-only analysis limit performance. Furthermore, discriminative challenges are presented among similar patterns, some differing primarily in intention. The reliance on static analysis (ASTs, CGs) limits detection of dynamic behaviors in patterns (e.g., Observer, Strategy), where runtime interactions (e.g., method invocations via reflection) might provide better hints. While dynamic analysis is promising for revealing hidden dependencies, it was not incorporated due to execution risks, high-overhead setup and execution costs, and coverage issues. In particular, pattern dataset limitations further constrain generalizability: the examples offer textbook implementations or intentional pattern-centric projects, lacking real-world diversity, partial realizations, or obfuscation variants (as initially planned but not fully evaluated). Imbalance among patterns persisted despite oversampling, skewing performance toward common patterns like Singleton and underrepresenting rare ones like Flyweight or Interpreter. Deeper analysis is required to understand any pattern misclassifications, as this could be due to multiple causes. E.g., the same object participating in multiple patterns, cases where the pattern structure differs primarily by its intent or purpose (highly context-sensitive), information about dynamic interactions is missing (e.g., due to reflection), etc. This deeper case-by-case analysis is included in future work. Environment limitations include the dependency on the Memgraph setup, which adds complexity and potential memory overhead for large heterogeneous graphs.

## VII. Conclusion

This paper developed and evaluated our automated design pattern detection approach DPD-NGDB, based on a neural graph database and modular pipeline architecture, and made available as an ensemble base model in our DPD-EM ensemble model approach. The evaluation was benchmarked against a dataset with the 23 GoF design patterns contained in over 5300 Java classes spanning 9 open-source realistic practical Java projects (plus an additional 23 single example patterns). The dataset offered independently documented DPs and were used for the automated DPD training and testing. With 5-fold cross-validation and leave one project out, an overall F1 score of 0.72 was achieved, while the large unseen test project achieved 0.17. Furthermore, three NGDB GNN variants were evaluated, with GAT and GINSAGE showing similarly high F1 scores, and GCN somewhat lower.

The use of multi-modal features (numerical from ASTs, graph-based from CGs, structural like inheritance) outperforms traditional ML approaches (e.g., Uchiyama et al. [29], Dwivedi et al. [30]) by capturing both syntactic and relational aspects. DPD-NGDB's high F1-scores (up to 94%) surpass reported benchmarks for tools like PINOT, which achieve high precision only for well-structured code. The DPD-EM ensemble approach, while not as effective as the NGDB base model yet, provides an ongoing framework for the combination of diverse base models, contributing to the exploration of ensemble methods for DPD in software engineering [48].

Future work includes experimentation with various ensemble base models and methods; integrating runtime tracing for dynamic analysis; combining static features with execution traces to better detect behavioral patterns; full obfuscation; scalability evaluation; variant detection; usage on generalized open-source projects; expansion beyond the GoF patterns; an evaluation across multiple programming languages; and a comprehensive industrial case study.

## Acknowledgment

The authors would like to thank Natalie Böhm, Marius Mühleck, and Sandro Moser for their assistance with various aspects of the design, implementation, and evaluation.

## References

[1] R. Oberhauser and S. Moser, "Design Pattern Detection in Code: A Hybrid Approach Utilizing a Bayesian Network, Machine Learning with Graph Embeddings, and Micropattern Rules," Proceedings of the Eighteenth International Conference on Software Engineering Advances (ICSEA 2023), IARIA, 2023, pp. 122-129.

[2] R. Minelli, A. Mocci, and M. Lanza, "I know what you did last summer: an investigation of how developers spend their time," In: Proceedings of the 2015 IEEE 23rd International Conference on Program Comprehension, IEEE Press, 2015, pp. 25-35.

[3] M. J. Pacione, M. Roper, and M. Wood, "A novel software visualisation model to support software comprehension," In: Proc. 11th Working Conference on Reverse Engineering, IEEE, 2004, pp. 70-79

[4] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Pearson, 1995.

[5] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, Pattern-Oriented Software Architecture: A System of Patterns, Vol. 1. John Wiley & Sons, 2008.

[6] Y. Guéhéneuc and G. Antoniol. "Fingerprinting Design Patterns". In: 11th Working Conference on Reverse Engineering (WCRE), IEEE, 2004, pp. 172–181. DOI: 10.1109/WCRE.2004.3

[7] Y. G. Guéhéneuc, "P-mart: Pattern-like micro architecture repository," Proceedings of the 1st EuroPLoP Focus Group on pattern repositories, pp. 1-3, 2007.

[8] P-MARt Pattern-like Micro-Architecture Repository. [Online]. Available from: https://www.ptidej.net/tools/designpatterns 2025.11.28

[9] M. G. Al-Obeidallah, M. Petridis, and S. Kapetanakis, "A survey on design pattern detection approaches," International Journal of Software Engineering (IJSE), 7(3), 2016, pp. 41-59.

[10] H. Yarahmadi and S. M. H. Hasheminejad, "Design pattern detection approaches: A systematic review of the literature," Artificial Intelligence Review, 53, 2020, pp. 5789-5846.

[11] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner and G. Monfardini, "The Graph Neural Network Model," in IEEE Transactions on Neural Networks, vol. 20, no. 1, 2009, pp. 61-80, doi: 10.1109/TNN.2008.2005605.

[12] M. Besta et al., "Neural Graph Databases," Proceedings of the First Learning on Graphs Conference (LoG 2022), PMLR 198, Virtual Event, 2022.

[13] H. Ren, M. Galkin, M. Cochez, Z. Zhu, and J. Leskovec, "Neural Graph Reasoning: Complex Logical Query Answering Meets Graph Databases," In: ArXiv abs/2303.14617, 2023.

[14] N. Francis et al., "Cypher: An evolving query language for property graphs," Proc. 2018 International Conference on Management of Data, 2018, pp. 1433-1445.

[15] T. G. Dietterich, "Ensemble Methods in Machine Learning," In: Multiple Classifier Systems. Springer, 2000, pp. 1-15. DOI: 10.1007/3-54045014-9_1.

[16] G. Seni and J. F. Elder, "Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions," In: Synthesis Lectures on Data Mining and Knowledge Discovery, Morgan & Claypool Publishers, 2010. DOI: 10.2200/S00240ED1V01Y200912DMK002.

[17] R. Oberhauser, "A Machine Learning Approach Towards Automatic Software Design Pattern Recognition Across Multiple Programming Languages," Proc. of the Fifteenth International Conference on Software Engineering Advances (ICSEA 2020), IARIA XPS Press, 2020, pp. 27-32.

[18] R. Oberhauser, "A Hybrid Graph Analysis and Machine Learning Approach Towards Automatic Software Design Pattern Recognition Across Multiple Programming Languages, " International Journal on Advances in Software, vol. 15, no. 1 & 2, year 2022, pp. 28-42. ISSN: 1942-2628.

[19] D. Yu, Y. Zhang, and Z. Chen, "A comprehensive approach to the recovery of design pattern instances based on sub-patterns and method signatures," Journal of Systems and Software, vol. 103, pp. 1-16, 2015.

[20] B. Mayvan and A. Rasoolzadegan, "Design pattern detection based on the graph theory," Knowledge-Based Systems, vol. 120, pp. 211-225, 2017.

[21] M. L. Bernardi, M. Cimitile, and G. Di Lucca, "Design pattern detection using a DSL-driven graph matching approach," Journal of Software: Evolution and Process, 26(12), pp. 1233-1266, 2014.

[22] M. Oruc, F. Akal, and H. Sever, "Detecting design patterns in object-oriented design models by using a graph mining approach," 4th International Conference in Software Engineering Research and Innovation (CONISOFT 2016), IEEE, 2016, pp. 115-121.

[23] A. Pande, M. Gupta, and A. K. Tripathi, “A new approach for detecting design patterns by graph decomposition and graph isomorphism,” International Conference on Contemporary Computing, Springer, Berlin, Heidelberg, 2010, pp. 108-119.

[24] P. Pradhan, A. K. Dwivedi, and S. K. Rath, “Detection of design pattern using graph isomorphism and normalized cross correlation,” Eighth International Conf. on Contemporary Computing (IC3 2015), IEEE, 2015, pp. 208-213.

[25] S. Alhusain, S. Coupland, R. John, and M. Kavanagh, “Design pattern recognition by using adaptive neuro fuzzy inference system,” 2013 IEEE 25th International Conference on Tools with Artificial Intelligence, IEEE, 2013, pp. 581-587.

[26] M. Zanoni, F. A. Fontana, and F. Stella, “On applying machine learning techniques for design pattern detection,” J. of Systems & Software, vol. 103, no. C, pp. 102-117, 2015.

[27] L. Galli, P. Lanzi, and D. Loiacono, “Applying data mining to extract design patterns from Unreal Tournament levels,” Computational Intelligence and Games. IEEE, 2014, pp. 1-8.

[28] R. Ferenc, A. Beszedes, L. Fulop, and J. Lele, “Design pattern mining enhanced by machine learning,” 21st IEEE International Conference on Software Maintenance (ICSM'05), IEEE, 2005, pp. 295-304.

[29] S. Uchiyama, A. Kubo, H. Washizaki, and Y. Fukazawa, “Detecting design patterns in object-oriented program source code by using metrics and machine learning,” Journal of Software Engineering and Applications, 7(12), pp. 983-998, 2014.

[30] A. K., Dwivedi, A. Tirkey, and S. K. Rath, “Software design pattern mining using classification-based techniques,” Frontiers of Computer Science, 12(5), pp. 908-922, 2018.

[31] H. Thaller, L. Linsbauer, and A. Egyed, “Feature maps: A comprehensible software representation for design pattern detection,” IEEE 26th international conference on software analysis, evolution and reengineering (SANER 2019), IEEE, 2019, pp. 207-217.

[32] A. Chihada, S. Jalili, S. M. H. Hasheminejad, and M. H. Zangooei, “Source code and design conformance, design pattern detection from source code by classification approach,” Applied Soft Computing, 26, pp. 357-367, 2015.

[33] Y. Wang, H. Guo, H. Liu, and A. Abraham, “A fuzzy matching approach for design pattern mining,” J. Intelligent & Fuzzy Systems, vol. 23, nos. 2-3, pp. 53-60, 2012.

[34] A. Alnusair, T. Zhao, and G. Yan, “Rule-based detection of design patterns in program code,” Int'l J. on Software Tools for Technology Transfer, vol. 16, no. 3, pp. 315-334, 2014.

[35] M. Lebon and V. Tzerpos, “Fine-grained design pattern detection,” IEEE 36th Annual Computer Software and Applications Conference, IEEE, 2012, pp. 267-272.

[36] I. Issaoui, N. Bouassida, and H. Ben-Abdallah, “Using metric-based filtering to improve design pattern detection approaches,” Innovations in Systems and Software Engineering, vol. 11, no. 1, pp. 39-53, 2015.

[37] A. K. Dwivedi, A. Tirkey, and S. K. Rath, “Software design pattern mining using classification-based techniques,” Frontiers of Computer Science, 12(5), pp. 908-922, 2018.

[38] F. A. Fontana, S. Maggioni, and C. Raibulet, “Understanding the relevance of micro-structures for design patterns detection,” Journal of Systems and Software, vol. 84, no. 12, pp. 2334-2347, 2011

[39] I. Issaoui, N. Bouassida, and H. Ben-Abdallah, “Using metric-based filtering to improve design pattern detection approaches. Innovations in Systems and Software Engineering,” vol. 11, no. 1, pp. 39-53, 2015.

[40] J. Dong, Y. Zhao, and Y. Sun, “A matrix-based approach to recovering design patterns,” IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 39, no. 6, pp. 1271-1282, 2009.

[41] J. Singh, S. R. Chowdhuri, G. Bethany, and M. Gupta, “Detecting design patterns: a hybrid approach based on graph matching and static analysis,” Information Technology and Management, 23(3), pp. 139-150, 2022.

[42] R. Barbudo, A. Ramírez, F. Servant, and J. R. Romero, “GEML: A grammar-based evolutionary machine learning approach for design-pattern detection,” Journal of Systems and Software, 175, p. 110919, 2021.

[43] M. Kouli and A. Rasoolzadegan, “A Feature-Based Method for Detecting Design Patterns in Source Code,” Symmetry, 14(7), p. 1491, 2022.

[44] J. Liu et al., "Learning Graph-based Code Representations for Source-level Functional Similarity Detection," In: Proc. IEEE/ACM 45th International Conference on Software Engineering (ICSE). IEEE, 2023, pp. 345-357. DOI:10.1109/ICSE48619.2023.00040.

[45] A. Ampatzoglou et al., "Design patterns mining using neural subgraph matching". In: Proc. 37th ACM/SIGAPP Symposium on Applied Computing, ACM, 2022, pp. 1435-1444. DOI: 10.1145/3477314. 3507073.

[46] M. Li, Y. Wang, and W. Zhang, "A Multi-Feature Fusion Approach for Design Pattern Detection Based on Graph Neural Networks," In: 2024 IEEE International Conference on Software Services Engineering (SSE), IEEE, 2024, pp. 1-10. DOI: 10.1109/SSE62679.2024.10633498

[47] Memgraph. [Online]. Available from: https://memgraph.com 2025.11.28

[48] M. Y. Mhawish and M. Gupta, "Software Metrics and tree-based machine learning algorithms for distinguishing and detecting similar structure design patterns". In: SN Applied Sciences 2(1) p.11, 2020. DOI: 10.1007/s42452-019-1815-3.

[49] N. Shi and R. A. Olsson, "Reverse Engineering of Design Patterns from Java Source Code," 21st IEEE/ACM International Conference on Automated Software Engineering (ASE'06), Tokyo, Japan, 2006, pp. 123-134, doi: 10.1109/ASE.2006.57.

[50] M. Tufano et al., "An Empirical Study on Learning Bug-Fixing Patches in the Wild via Neural Machine Translation," In: ACM Transactions on Software Engineering and Methodology (TOSEM), 30(4), 2021, pp. 1-37. DOI: 10.1145/3450284.

[51] J. Dong, "Focal Loss Improves the Model Performance on Multi-Label Image Classifications with Imbalanced Data". In: Proceedings of the 2nd International Conference on Industrial Control Network and System Engineering Research, 2020, pp. 18-21.

[52] R. S. Pienta et al., "MAGE: Matching approximate patterns in richly attributed graphs". In: 2014 IEEE International Conference on Big Data (Big Data), IEEE, 2014, pp. 585-590.

[53] Refactoring.Guru. [Online]. Available from: https://refactoring.guru/design-patterns/java 2025.11.28

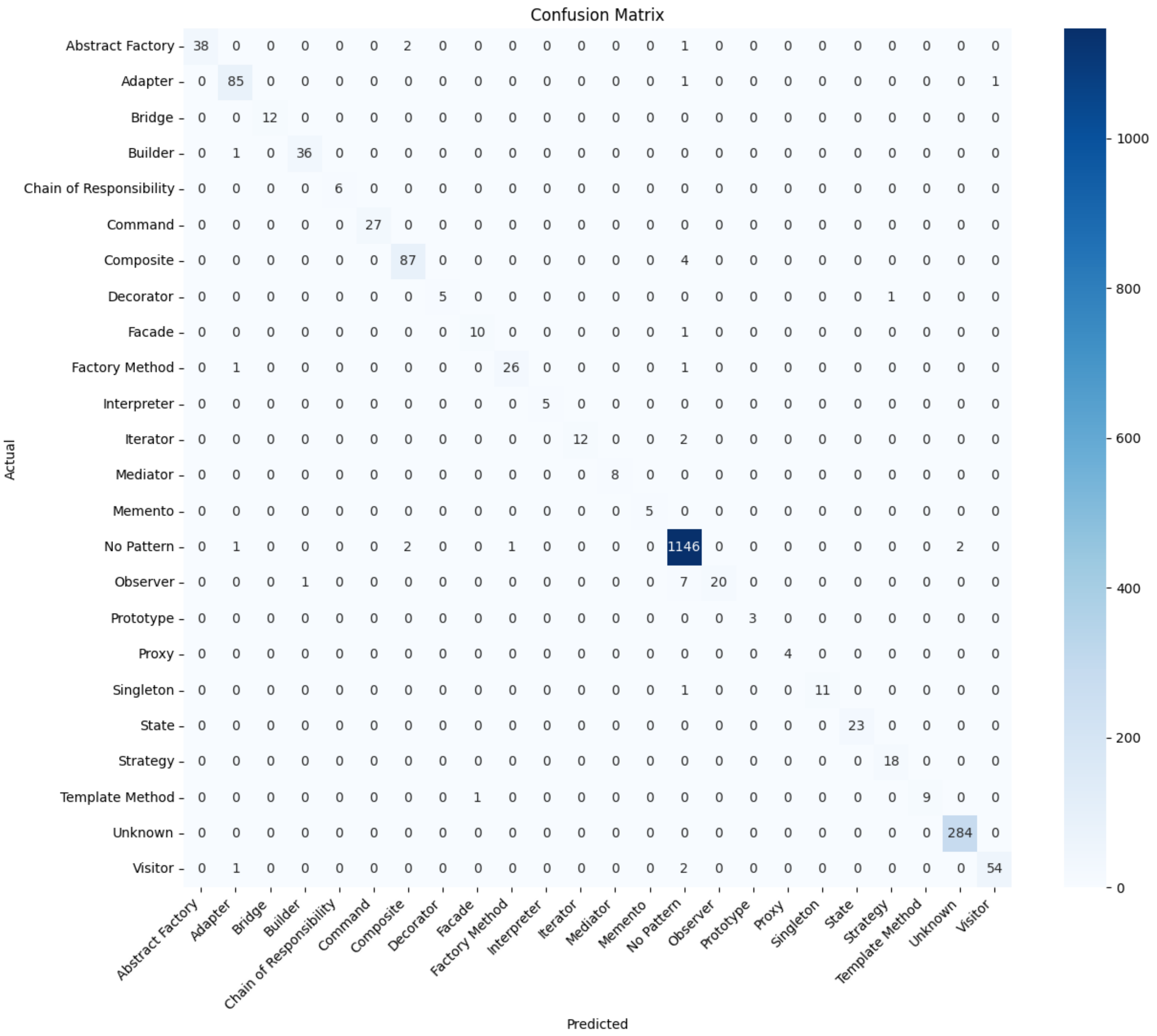


Figure 17. Confusion Matrix for DPD-NGDB GINSAGE showing strong DPD performance (diagonal) across the entire pattern dataset.

TABLE IV. DPD RESULTS USING P-MART TEST PROJECTS (LOPO-CV NETBEANS)

| Project | Classes | Scored | Accuracy | Precision | Recall | F1 | Pattern Types | Abstract Fac- | Adapter | Bridge | Builder | Command | Composite | Decorator | Facade | Factory | Iterator | Memento | Observer | Prototype | Singleton | State | Strategy | Template | Visitor |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **JHotDraw v5.1** | 153 | 116 | 0.91 | 0.99 | 0.91 | 0.94 | 10 | x | | | | | x | x | | x | | | x | x | x | x | x | x | |
| **JRefactory v2.6.24** | 568 | 206 | 0.50 | 1.00 | 0.69 | 0.79 | 6 | | x | | x | | | | | x | | | | | x | x | | | x |
| **JUnit v3.7** | 89 | 55 | 0.69 | 1.00 | 0.54 | 0.68 | 4 | | | | | | x | | | | x | | x | | x | | | | |
| **Lexi v0.1.1 alpha** | 97 | 16 | 0.31 | 1.00 | 0.53 | 0.88 | 3 | | | | x | | | | | | | | x | | x | | | | |
| **MapperXML v1.9.7** | 222 | 63 | 0.59 | 0.98 | 0.65 | 0.74 | 8 | x | x | | | | x | | x | | | | x | | x | | x | x | |
| **Netbeans v1.0.x** | 3347 | 267 | 0.03 | 0.32 | 0.11 | 0.17 | 4 | x | x | | | x | | | | | x | | | | | | | | |
| **Nutch v0.4** | 244 | 46 | 0.17 | 1.00 | 0.20 | 0.55 | 7 | | x | x | | x | | | | | x | x | | | x | | x | | |
| **PMD v1.8** | 472 | 55 | 0.71 | 0.94 | 0.61 | 0.79 | 8 | | x | | x | | x | | | x | x | | x | | | | | x | x |
| **QuickUML 2001** | 193 | 64 | 0.86 | 0.83 | 0.69 | 0.88 | 6 | x | | | x | x | x | | | | | | x | | x | | | | |
| **OVERALL** | 5385 | 888 | 0.45 | 0.96 | 0.58 | 0.72 | 18 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |

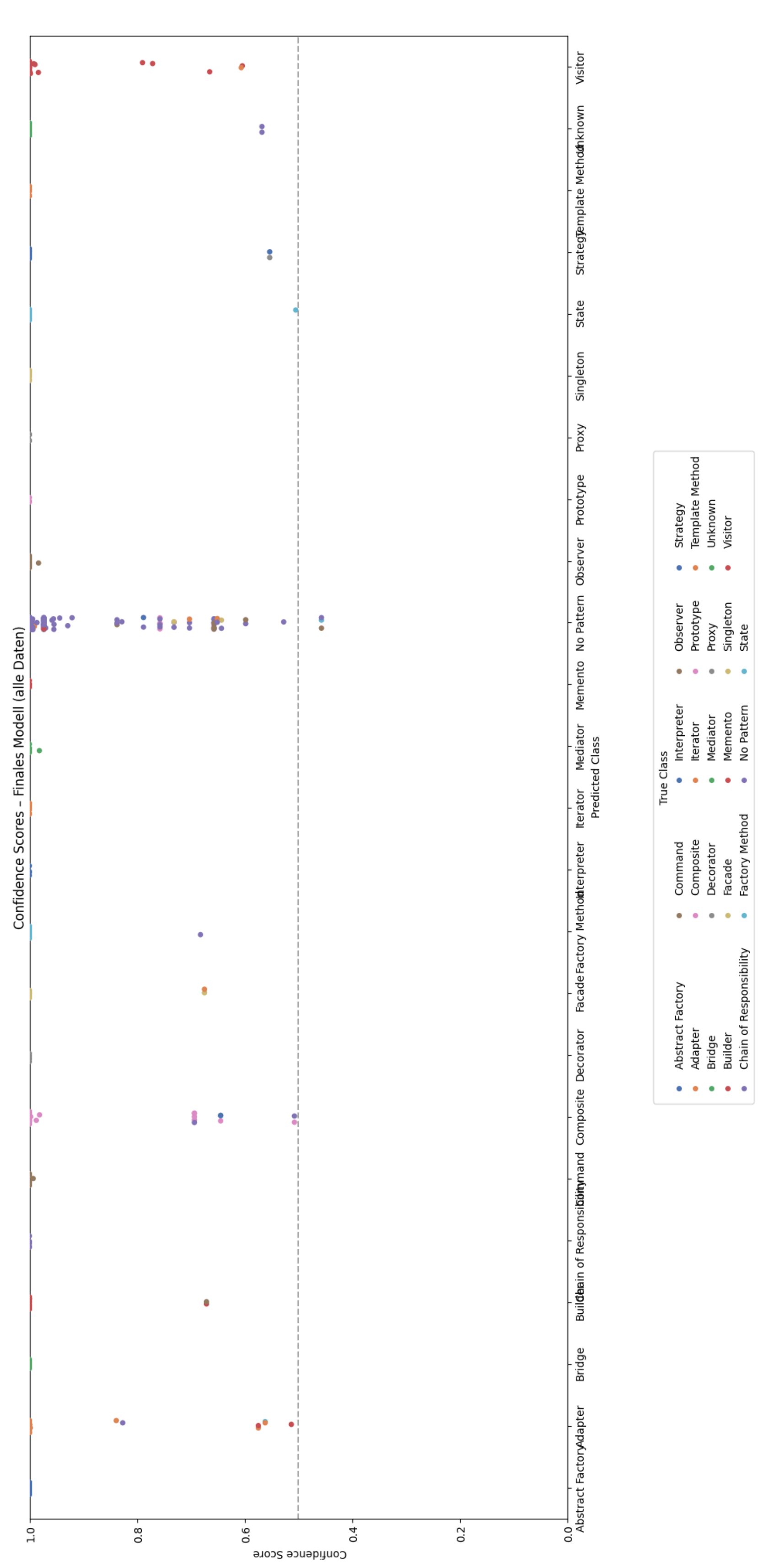


Figure 18. DPD-NGDB confidence scores for the entire dataset (NetBeans left out of training set).

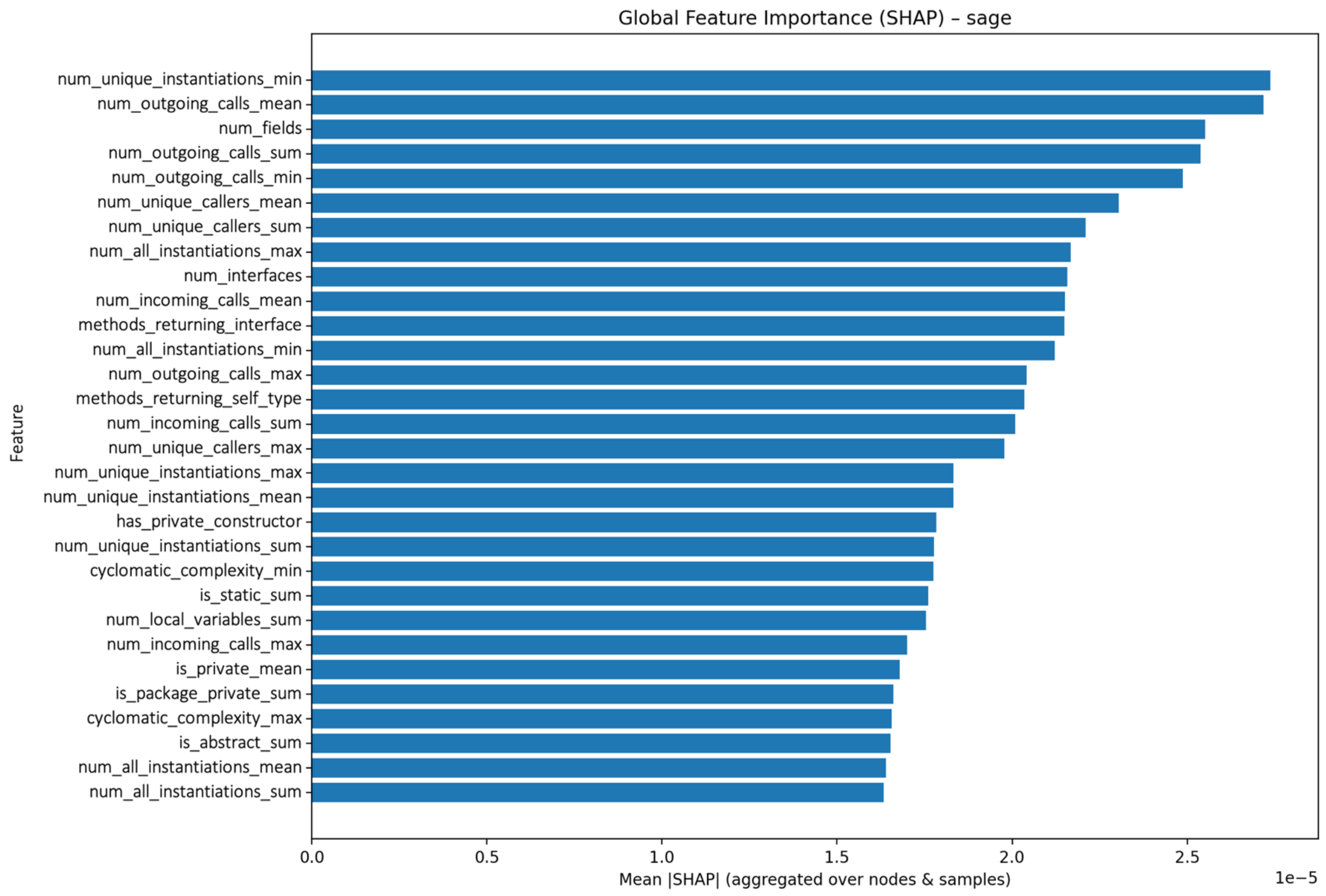


Figure 19. SHAP feature importance for DPD-NGDB across the entire dataset (generated diagram edited to replace feature numbers with names).

# GDPR Compliance Verification through Data Provenance

Alba Martinez Anton
Aix-Marseille Univ., LIS, CNRS
e-mail: alba.martinez-anton@univ-amu.fr

Clara Bertolissi
INSA Centre Val de Loire, LIFO, France
e-mail: clara.bertolissi@insa-cvl.fr

***Abstract*—With the exponential growth in the collection of personal data, ensuring compliance with data protection regulations such as the General Data Protection Regulation (GDPR) has become a critical challenge. In this work, we present a formal approach to GDPR compliance verification based on data provenance. We model the behavior of systems as provenance graphs, capturing the lifecycle of personal data across processes. Compliance patterns corresponding to key GDPR principles and rights are expressed as Prolog rules, enabling logical inference over these graphs. We present a verification tool which evaluates whether system executions meet legal obligations, and provides interpretable feedback in case of violations. We provide experimental validation on synthetic graphs to demonstrates the feasibility of our approach.**

***Keywords-GDPR; provenance; verification; Prolog.***

## I. Introduction

With the increase in the number of computer systems and the proliferation of online services, the collection and processing of personal data has grown at an unprecedented scale. This growing dependence on data-intensive services makes the protection of personal data a critical concern. In response, legal frameworks such as the General Data Protection Regulation (GDPR) have been introduced to establish rights for individuals and responsibilities for data controllers and processors.

While the GDPR provides a clear legal foundation, its effective implementation remains a significant challenge. Verifying that a system complies with GDPR requirements involves analyzing both the system's specifications and its actual behavior during runtime. Being capable of enforcing, monitoring, and verifying compliance within the complex and dynamic ecosystems of modern digital infrastructures is still a challenge today. Indeed, compliance must be ensured at several levels: the design of processes, the configuration of data storage systems, and the execution of actions such as consent collection, data sharing, and deletion.

Manual compliance verification is not only tedious and error-prone, but often infeasible for complex systems with high data volumes and dynamic behaviors. Furthermore, existing auditing tools tend to focus on static policies or incomplete logs, and do not capture the full context of data usage over time. There is thus a growing need for formal, automated methods to evaluate GDPR compliance.

Fortunately, the increasing computerization of data processing offers an opportunity to support, and in some cases automate, parts of the compliance verification process. In particular, by leveraging structured representations of system behavior and data lifecycles, based on provenance models, it becomes possible to formalize and check compliance requirements seen as patterns in provenance graphs.

**Approach.** Our work proposes a formal approach to represent and verify compliance with key GDPR principles using provenance graphs. This representation enables us to track data processing chains and apply rule-based checks, thus bridging the gap between legal requirements and executable system behaviors. Our method complements existing compliance-by-design efforts by offering a verification mechanism capable of evaluating compliance post hoc, based on actual data processing traces.

Provenance provides a natural foundation for compliance verification because it captures who performed which operations on which data, when, and under which dependencies. Leveraging this representation, we identify the legal primitives relevant to GDPR, such as data items, data subjects, processing activities, purposes and rights-related actions, and map them to provenance entities, activities, and relations following the Open Provenance Model. This creates a uniform vocabulary in which legal requirements can be expressed independently of the underlying system. Then, we formalize GDPR principles, such as lawfulness of processing, the right to be forgotten, data minimization, and access rights, as declarative logic patterns over the provenance graph. These patterns specify constraints on the provenance graph structure, often involving specific sequences or combinations of events. For example, purpose limitation is formalized as a relationship between consent events, permitted purposes, and subsequent uses of personal data. Patterns are declarative and reusable: once defined, they apply to any provenance graph that adheres to the underlying schema, regardless of the system domain. This shifts the focus from tool-specific implementation details to a general, principle-driven method for analysing GDPR compliance through provenance.

The formal patterns constitute the foundation of our approach: they map regulatory concepts to precise constraints on the provenance graph, define reusable templates for expressing GDPR obligations, and support principled analyses of real system behaviors. To evaluate these patterns over concrete provenance data, we rely on a Prolog-based engine. The logical specifications are encoded directly as Prolog predicates. When the provenance graph is loaded, these predicates are instantiated with the corresponding entities, activities, and timestamps, allowing the engine to systematically traverse the graph and determine whether the constraints are satisfied or violated. This approach ensures coherence between the abstract legal formalization and the executable checks performed on real execution traces.

To assess scalability, we also developed a provenance graph generator based on modular building blocks (called "bricks") representing common user interactions. These building blocks can be assembled in various configurations to simulate realistic data flows in different application domains (e.g., social networks, online shopping, public administration). This enables us to test our approach on large and diverse graphs under controlled conditions.

**Contributions.** This work presents a comprehensive framework for the automated verification of GDPR compliance based on data provenance. It extends [1] by providing a detailed presentation of the generalized provenance model and by formalizing a significant subset of GDPR principles in the form of provenance patterns. Moreover, a graph generator was implemented, and an experimental evaluation using the Prolog prototype was carried out to assess performance results.

Our key contributions are:

- A formal representation of selected GDPR principles and data subject rights as logic-based patterns that can be evaluated over provenance graphs.
- A verification tool, implemented in Java and Prolog, that allows auditors to configure compliance checks, select specific rights or principles to verify, and receive detailed feedback on potential non-compliance or pending verifications.
- A method to produce synthetic but realistic datasets for experimentation. We generate realistic provenance graphs based on reusable components.
- An experimental validation showing the feasibility of our method on both manually crafted and large, synthetic provenance graphs, with performance analysis and scalability insights.

Overall, our work demonstrates how logical reasoning over provenance data can provide a powerful foundation for automated, explainable, and customizable compliance verification tools. This contributes to the broader effort of making digital systems not only more transparent, but also more accountable with respect to data protection regulations.

The structure of this paper is as follows. Section II discussed the state of the art. Section III introduces the context of the General Data Protection Regulation (GDPR) and motivates our work. Section IV presents the necessary preliminaries on provenance modeling and describes our extensions to capture GDPR requirements. In Section V, we detail our formal approach for modeling compliance requirements. Section VI outlines our methodology for semi-automated compliance verification together with a prototype tool implementation. Section VII reports on the experiments conducted to validate our approach. Finally, we conclude with the perspectives for future research in Section VIII.

## II. Related Work

The General Data Protection Regulation has triggered a broad range of research efforts aimed at understanding, supporting, and enforcing data protection principles within digital systems.

Several studies have sought to analyze whether real-world systems comply with GDPR requirements [2][3][4]. These works often rely on case studies, audits, or systematic evaluations of privacy policies and data processing practices. While valuable for revealing compliance gaps, such approaches are mostly retrospective and provide limited automation or generalizability.

Another line of research focuses on developing technical solutions to support compliance checking in a more systematic way, see for instance [5][6][7]. These works are based on the formalization of legal norms themselves. This effort is crucial to support both design- and runtime compliance, as it provides the formal foundation upon which reasoning, verification, and automation can be built. Translating legal texts into formal logic is a complex and delicate task that requires capturing the subtleties of legal language, its contextual dependencies and normative intent in a structured, machine-interpretable form.

Various approaches have been proposed to bridge the gap between natural legal language and logical formalism. A widely adopted strategy involves the use of ontologies to model legal knowledge. As discussed in [8], ontologies provide a structured representation of legal concepts and their interrelations, making it easier to translate legal provisions into sets of logical rules. This modeling facilitates knowledge sharing, consistency checking, and reasoning over legal domains. While ontology-based approaches offer rich semantic models of GDPR concepts, they primarily support reasoning over declared system properties. In contrast, our approach focuses on operational compliance: provenance graphs capture the actual flow of personal data and the causal structure of system executions. This makes it possible to detect violations that arise from concrete system behavior, rather than from inconsistencies in the system's specification. Ontology-based models and provenance-based verification may be complementary: ontologies can supply high-level semantic annotations, whereas provenance provides the fine-grained, event-level evidence required for auditing real system executions.

Another notable development is LegalRuleML [9], an XML-based extension specifically designed to represent legal norms, their conditions of applicability, and the structure of legal arguments. LegalRuleML aims to support automated reasoning and interoperability between legal information systems, offering a standardized way to encode normative content.

Complementary to these formalization efforts, some research has focused on making logical representations accessible and verifiable by legal experts. For instance, Bartolini et al. [10] propose using RIO logic to express legal provisions in a structured but human-readable format. This representation allows legal practitioners to validate the formalization of regulations while preserving the rigor needed for automated compliance checking. Such approaches enhance the transparency and trustworthiness of logic-based systems used in legal contexts.

Since the adoption of the GDPR in 2016, considerable research has focused on designing systems that are compliant by design, in line with Article 25 of the regulation. In [11], the authors extract Technical and Organizational Measures (TOMs) directly from the GDPR text to support the development of a compliance-assisting tool. This tool relies on contractual relationships between users and organizations, implemented through knowledge graphs. This formalization allows the system to reason over contractual obligations and compliance requirements in a machine-interpretable way.

Building upon this foundation, [12] extends the previous work by incorporating the notion of consent and adapting the tool to the context of smart cities and IoT environments. The primary goal of this enhancement is to ensure GDPR compliance for resource-constrained or distributed infrastructures.

Other studies, such as [13], approach compliance by focusing on business process modeling. The authors propose a methodology for aligning business workflows with the obligations imposed on data controllers by the GDPR. While the work offers a structured approach to incorporating privacy by design, it remains theoretical, demonstrated only through a case study involving a telecommunications operator, and does not include a dedicated tool or automation component.

Similarly, [14] employs business process models to embed compliance features in IoT systems. Their work stands out by introducing smart contracts and blockchain technologies to facilitate automated compliance verification. Drawing on the methodology of [15], they address several key GDPR principles, including confidentiality, consent, data minimization, data protection, and data transfer.

Also, numerous frameworks and guidelines have been proposed to support developers in incorporating privacy requirements throughout the system lifecycle. For instance, Saltarella et al. [16] survey approaches that support system designers in aligning their architectures with GDPR principles from the outset. Similarly, Rhahla et al. [17] present a comprehensive study of academic and industrial tools intended to facilitate GDPR compliance, with a focus on development-time integration.

Compared to these approaches, that emphasizes design-time compliance or high-level modeling, our contribution aims to formalize and verify that runtime behaviors, specifically, actions performed on personal data, adhere to the GDPR's principles and constraints. In contrast to the above efforts, relatively few works have addressed the runtime verification of actions and behaviors performed by systems, especially from the perspective of compliance with GDPR obligations. Ensuring that concrete system operations, such as data access, sharing, deletion, or processing, respect legal constraints in practice remains a largely open challenge. Existing approaches often lack the granularity or formality needed to assess compliance at the level of individual actions or data flows.

In [6], the authors translate a subset of GDPR articles into temporal logic formulas to automate compliance verification. Their approach leverages MonPoly, a monitoring tool that processes system event logs by translating them into MFOTL logic and compares them against GDPR rules to detect potential violations. While effective for specific cases, this work remains limited to a small subset of GDPR provisions and does not address key principles such as purpose limitation.

Building on a broader scope, [18] introduces a framework developed within the European project Cloud4EU [19], which combines different techniques to verify GDPR compliance in cloud environments for the public sector. Their contribution lies in the semi-automated translation of regulatory texts into logical rules, enabling dynamic updates as new laws or amendments are introduced. These rules can be applied both to runtime log verification and to compliance-by-design approaches, where traces generated from business models are checked against GDPR requirements. Nevertheless, the use of LegalRuleML within this framework remains relatively basic. As the authors themselves highlight, the current metamodel is primarily geared toward legal representation and requires significant extensions to support advanced compliance reasoning.

Other lines of research explore the use of machine learning to automate parts of this process, especially the translation of legal texts. For example, [20] investigates text processing techniques to facilitate compliance reasoning. Similarly, Zimmeck et al. [21] apply natural language processing to automatically extract GDPR-relevant requirements from privacy policies, demonstrating the potential of NLP for large-scale analysis. However, these approaches still face challenges in accurately capturing certain GDPR concepts, whose semantic complexity and context-dependence are difficult to formalize.

Our work follows the same general line of formal specification of [6], which relies on temporal logic to reason about consent and event sequences, but we adopt a broader provenance-based perspective. By introducing a set of purpose-oriented and constraint-based provenance patterns, our framework supports the expression of GDPR principles that involve conditional or evolving requirements. This makes it suitable for capturing a wider range of compliance scenarios, including the verification not only of whether consent exists for a given piece of data, but also of whether the specific purpose stated in the consent request is actually respected. Moreover, in contrast with approaches based on temporal logic or high-level ordering constraints between events, our provenance-based representation provides explicit temporal information attached to provenance activities. Therefore, we can determine exactly when a consent was granted, how long it remained valid, and during which period a given piece of personal data was accessed, transmitted, or stored. By using in the analysis concrete time intervals extracted in the provenance graph directly from system logs, our approach captures forms of temporal misuse that cannot be detected using purely qualitative or symbolic temporal relations.

## III. Context and Motivations

### A. The General Data Protection Regulation

In order to safeguard the privacy of its citizens, the European Union adopted a directive in 1995 [22], introducing the notions of informed consent, the right of access and rectification, and the obligations of data controllers with respect to the collection, processing, and storage of personal data. This directive laid the foundation for the General Data Protection Regulation

(GDPR), which came into force in 2018 and has since become the cornerstone of European data protection law.

The GDPR establishes a comprehensive framework that regulates the collection, processing, and dissemination of personal data, and it applies to any organization, whether public or private, operating within the EU or targeting European residents. In its Article 4, the regulation defines key concepts to delineate its scope. Personal data is understood as any information relating to an identified or identifiable natural person, the data subject. Processing is broadly defined as any operation performed on personal data, from collection and recording to storage, modification, or dissemination. The notion of data controller refers to the person or organization that determines the purposes and means of the processing and is ultimately responsible for compliance. Consent is defined as a freely given, specific, informed, and unambiguous indication by the data subject of their agreement for personal data to be processed for a clearly stated purpose.

The regulation is articulated around a set of fundamental principles that ensure lawful, fair, and transparent processing, restrict the use of data to explicit and legitimate purposes, and require that only the necessary amount of information be collected and maintained accurately over time. It also limits storage to what is strictly necessary, obliges organizations to guarantee the confidentiality and integrity of data through appropriate safeguards, and finally, introduces the principle of accountability, by which controllers must be able to demonstrate their compliance at all times. In addition to these principles, the GDPR includes a wide range of rights for individuals, including the right of access, rectification, erasure (often referred to as the "right to be forgotten") as well as the rights to restrict processing, to data portability, and to object to certain forms of processing. Together, these provisions aim to give individuals effective control over their personal information in the digital society.

## B. Motivations

Although the objectives of the GDPR are clear, ensuring compliance in practice represents a considerable challenge for organizations. This challenge is both organizational and technical, since it requires translating often complex legal requirements into operational practices and embedding them within the functioning of information systems. A particularly delicate issue concerns the notion of purpose. Organizations must not only declare the reasons why they collect and process personal data, but also guarantee throughout the entire lifecycle of the data that these purposes are respected and remain aligned with the consent initially given by individuals. Tracking and documenting these purposes across complex systems is a demanding task, especially when data is reused in multiple contexts.

Compliance also demands meticulous documentation of processing activities, the legal bases invoked, the consents obtained, and the technical and organizational measures put in place to ensure security. These obligations are generally verified through audits, which assess the adequacy of procedures. However, such audits are often labor-intensive and retrospective in nature. Automating them, at least partially, is a pressing need if organizations are to sustain compliance in a scalable and efficient manner.

Further difficulties arise from the necessity to ensure that modifications such as rectification or erasure are consistently propagated across all replicas. This issue of replication consistency is critical for respecting principles such as accuracy and storage limitation, as well as rights such as the right to erasure.

These different challenges leave traces within the systems themselves, in the form of logs or event records that capture the sequence of operations performed on data. By analyzing these traces, it becomes possible to assess whether certain principles or rights have been respected, thereby opening the way to automated forms of compliance verification. However, it is important to recognize that not all principles or rights lend themselves to such automation. Certain notions embedded in the GDPR, such as fairness and transparency, cannot be automatically verified because they depend on communication methods and contextual nuances that system traces cannot capture. The automation of legal verification therefore has limits: while it can significantly reduce human error, accelerate verification processes, and help maintain continuous compliance, it cannot fully replace human judgment when it comes to subjective assessments. For instance, the evaluation of whether a request for consent is expressed in clear and understandable terms (article 12 to 14) remains outside the reach of automated systems. Similarly, the principle of data minimization raises complex challenges, since determining whether a dataset is “necessary” (article 5.1.c) for a given purpose often requires domain expertise and contextual knowledge. In such cases, compliance is better supported by establishing precise specifications of procedures and processes, and then verifying that these specifications are respected. Furthermore, many parts of the GDPR, such as Chapters 6, 7, and 8, dealing respectively with supervisory authorities, cooperation mechanisms, and remedies and penalties, regulate institutional, procedural and enforcement aspects rather than the technical or operational rules of data processing.

Given these limitations, our work focuses on the subset of GDPR provisions that can be addressed through system-level analysis. More specifically, we concentrate on the principles of lawfulness, purpose limitation, and storage limitation, as well as the rights of access and erasure. These aspects are particularly amenable to verification through system traces and provenance data, since they can be formalized as constraints on the occurrence, timing, and consistency of processing events. Our approach therefore emphasizes the automation of articles directly concerned with event compliance, while considering semi-automated methods for those that require complementary verification.

In this way, we aim to contribute to the broader goal of bridging the gap between legal requirements and system implementation, by providing methods that support both automation where feasible and human oversight where necessary.

## C. Examples

In this work, we concentrate on three aspects that we consider fundamental: the traceability of purposes, the enforcement of storage limitations, and the consistency of data erasure across replicas.

Purpose traceability requires that consented purposes accompany personal data throughout all processing activities, ensuring that data is not diverted from its legitimate use.

*Example 1 (Purpose traceability):* A system respects the purpose limitation principle when personal data is processed exclusively for the purposes explicitly consented to by the data subject. Consider the following situation in an online forum: Bob joins a discussion group, and as part of this action, his personal data is associated with the creation of cookies. Two different scenarios illustrate compliance and non-compliance.

**Compliant scenario:** Bob joins a cooking interest group. In this context, the forum generates an analytics cookie used solely to improve the performance of the website. The cookie remains internal to the platform and is accessible only to the maintenance team, fully aligned with the purpose to which Bob initially consented.

**Non-compliant scenario:** Bob joins the same cooking interest group, and the forum once again creates an analytics cookie. However, despite Bob having explicitly withdrawn his consent for any sharing with third parties, the cookie is nonetheless transmitted to an external kitchenware vendor. This constitutes a clear violation of the purpose limitation principle, as the data is processed beyond the scope of the consent provided.

Concerning storage limitations, temporal constraints must be verified to check that data is retained no longer than necessary and that requests from data subjects are honored within prescribed deadlines.

*Example 2 (Data storage limitation):* The storage limitation principle requires that personal data be retained only for as long as necessary in relation to the purposes for which it was collected. Automating compliance with this principle typically involves checking that data subject information is deleted or anonymized once the retention period has expired. Consider the case of Alice, who creates an account on an online forum and contributes content. The following scenarios illustrate compliance and non-compliance.

**Compliant scenario:** Alice registers on the forum and actively participates by posting reviews. After five years of inactivity, the system enforces its retention policy and deletes Alice's account data. Two years later, when Alice attempts to log in again, she receives an error message indicating that the account no longer exists. This demonstrates compliance with the predefined five-year retention limit.

**Non-compliant scenario:** Alice registers on the forum and posts several reviews. After seven years without activity, she attempts to log in again and is still able to access her account with all of her personal data intact. This persistence of data beyond the declared five-year limit constitutes a violation of the storage limitation principle.

Data erasure consistency, finally, is needed to guarantee that compliance-related actions such as deletion are properly applied across all copies of the data within the system.

*Example 3 (Consistency of data deletion across copies):* Compliance with the right to erasure requires that when a deletion request is made, all copies of the corresponding data within the system are removed. These copies may exist in different modules of the system, such as the main database, user activity logs, or temporary caches. Consider the following scenarios for David, who has an account in an online forum.

**Compliant scenario:** David requests the deletion of his account. The system processes the request by erasing his personal data from the main user database, cleaning the activity logs, and clearing cached data related to his profile. When David later attempts to log in, his account no longer exists, confirming that all copies of his data have been consistently deleted.

**Non-compliant scenario:** David deletes his account, and the system removes it from the main user database. However, his data remains in the activity logs used by the forum administrators. Weeks later, traces of David's activity are still accessible, revealing that the deletion request was not fully enforced across all system copies, thus violating the GDPR.

Together, these challenges highlight the necessity of systematic and automated approaches to compliance verification, which we aim to address through the use of provenance information.

## IV. Preliminaries

In this section we present the Open Provenance model [23] and its generalization to represent GDPR requirements.

### A. The Open provenance model

Provenance information is usually represented as a labeled direct graph (called *provenance graph*) that expresses how objects evolve in the system. A provenance graph is a triple $G = (\mathcal{N}, \mathcal{E}, L)$, where $\mathcal{N}$ is a set of nodes, $\mathcal{E}$ is a set of directed edges between nodes, and $L$ is a set of labels. A node in the graph can represent an *artifact*, which is an immutable piece of data, a *process*, which is used to denote the action performed on an artifact and resulting in a new artifact; or an *agent*, which is used to denote the entity controlling or affecting the execution of a process. We denote the sets of artifacts, processes, and agents as $\mathcal{O}$, $\mathcal{P}$, and $\mathcal{A}$, respectively (with $\mathcal{N} = \mathcal{O} \cup \mathcal{P} \cup \mathcal{A}$ and $\mathcal{O}, \mathcal{P}, \mathcal{A}$ pairwise disjoint). Edges represent labeled casual dependencies between nodes. Role labels $\mathcal{R} \subseteq L$ define the function of an agent or an artifact in a process.

The Open Provenance model defines three main causal dependencies (see Table I): used, wasGeneratedBy, and wasControlledBy. The first two dependencies indicate a link between an artifact and a process: used connects a process to the artifacts used in the process and wasGeneratedBy links a process to the resulting artifact. Dependency wasControlledBy indicates which agent controls a process. The Open Provenance model also includes additional dependencies that can be derived from a provenance graph by composing dependencies. For instance, composed dependency wasTriggeredBy indicates that the execution of a process was triggered by another process, and dependency wasDerivedFrom denotes that an artifact was derived from another artifact. As done in [24], we make the role parameter explicit to be able to specify what role the first artifact played in the creation of the new one. Multi-step dependencies can be defined by transitive closure. For instance, wasDerivedFrom$^+$ captures, for an artifact, all artifacts used to derive it, possibly indirectly. Similarly, wasTriggeredBy$^+$ captures, for a process, all its direct and indirect triggering processes. The latter is also used to define the indirect use (used$^+$) and generation of artifacts (wasGeneratedBy$^+$).

To reason over temporal aspects and information evolution in a system, we extend the Open Provenance model with time observations, following [23]. We assume time is measured by an observer according to a single clock (or synchronized clocks), in such a way that time observations may be comparable. Moreover, observed time is expected to be compatible with causal dependencies, e.g., an artifact must exist before it is being used, a process generates artifacts before it ends, etc. In other words, causality implies time precedence. Provenance causal dependencies are decorated with time information expressed as timestamps associated with edges. We consider a series of ordered time points denoted as $ts_1, \ldots, ts_n \in \mathcal{T}$, with $ts_1 < ts_2 < \ldots < ts_n$, which represent specific points in time. We define a time interval $[ts_s, ts_e]$, as the interval of time between $ts_s \in \mathcal{T}$, starting time point, and $ts_e \in \mathcal{T}$, ending time point. We assume that a process runs for a time interval $[ts_s, ts_e]$, with $ts_s, ts_e \in \mathcal{T}$. All used or generated artifacts are used or created during this time window. Accordingly, in the provenance model with time observations, edges "wasControlledBy" are extended with the time interval $[ts_s, ts_e]$, and edges "used" and "wasGeneratedBy" are extended with a timestamp $ts \in [ts_s, ts_e]$ indicating that the artifact was used by the process (in the case of used) or that the artifact was created by the process (in the case of wasGeneratedBy) at a given time $ts$.

Following [24], we introduce two custom dependencies to capture the owner and the contributors of an artifact, as presented in the third block of Table I. In particular, owns captures the classical assumption in discretionary access control where the agent creating an artifact owns that artifact; contributedTo extends dependency wasControlledBy to identify the agents that have contributed, directly or indirectly, to the creation of an artifact. While in [24], the dependency wasControlledBy and, in turn, the dependency contributedTo were used only to link a process with an agent involved actively in a process, here we extend these dependencies to link a process to an agent involved actively or passively in the creation of an artifact. To this end, we explicitly model a role in contributedTo to denote in which way a subject was involved in the creation. If the role in contributedTo is *owner*, the casual dependency has the same meaning as the one in [24].

We also introduce the dependency notAvailable to capture that a previously created artifact is no longer available in the system (i.e., the artifact has been deleted).

As done previously, we can extend these dependencies with a timestamp. We suppose that an agent has owned an artifact since its generation (i.e., the timestamp associated with the edge owns matches the time parameter in the composing wasGeneratedBy dependency). Similarly, for the dependency contributedTo, its timestamp coincides with the time associated with the generation of the artifact. The notAvailable dependency has a timestamp indicating the time when the artifact has been deleted (thus matching the timestamp of the composing used dependency). It is worth noting that the definition of this last dependency uses a predicate **action** that, given a specific process instance, returns its type. This is needed because policies typically specify "types" of actions rather than process instances, as demanded by the provenance graph.

Provenance information can be represented graphically, as illustrated in Fig. 1. Following the standard notation for provenance graphs [23], artifacts are represented as circles, processes as rectangles, and agents as octagons. Edges are annotated with the type of dependency, the role of artifacts and agents in processes and the temporal information. For instance, in the graph of Figure 1, one can see that the artifact `data report` has been generated by the process `sendData` using the artifact `data request`.

## B. Extending provenance information for compliance

In the context of the GDPR, it is crucial to represent personal data in a way that allows one to identify the processes acting upon it, the purposes for which it is used, the consent given by data subjects, and the degree of compliance with user rights.

To achieve this, we extend the provenance model so that it explicitly distinguishes personal data and better captures consent mechanisms. This extension is implemented through the introduction of additional artifact nodes and enriched node attributes.

*a) Specific nodes.:* When personal data is involved, individuals are typically presented with a template of possible choices for granting or denying consent (a privacy policy template). In the provenance graph, such a template is represented as an artifact node. Once the individual makes a decision, the corresponding consent is also modeled as an artifact. If consent is later modified, a new artifact is created to represent the updated state. This new artifact is linked to the previous one via a wasDerivedFrom edge, ensuring versioning of consent. For instance, in Figure 1, Bob's consent evolves from *consent_bob_v0* to *consent_bob_v1* after he adjusts the purposes he had initially authorized.

Purposes themselves are captured as attributes associated with consent artifacts. It is important to note that artifacts in the provenance model are immutable objects: any modification to data or consent does not overwrite the existing artifact but

| **Basic casual dependencies from [23]** |
|---|
| $\mathsf{used}(p:\mathcal{P}, d:\mathcal{O}, r:\mathcal{R}, ts_u:\mathcal{T})$ |
| $\mathsf{wasGeneratedBy}(d:\mathcal{O}, p:\mathcal{P}, r:\mathcal{R}, ts_g:\mathcal{T})$ |
| $\mathsf{wasControlledBy}(p:\mathcal{P}, s:\mathcal{A}, r:\mathcal{R}, ts_b \in \mathcal{T}, ts_e:\mathcal{T})$ |
| $\mathsf{wasTriggeredBy}(p_1:\mathcal{P}, p_2:\mathcal{P}, ts_u:\mathcal{T}) = \exists d \in \mathcal{O}, r_1, r_2 \in \mathcal{R}, ts_g \in \mathcal{T} : \mathsf{used}(p_1, d, r_1, ts_u) \wedge \mathsf{wasGeneratedBy}(d, p_2, r_2, ts_g)$ |
| $\mathsf{wasDerivedFrom}(d_1:\mathcal{O}, d_2:\mathcal{O}, r:\mathcal{R}, ts_g:\mathcal{T}) = \exists p \in \mathcal{P}, r_i \in \mathcal{R}, ts_u \in \mathcal{T} : \mathsf{wasGeneratedBy}(d_1, p, r_i, ts_g) \wedge \mathsf{used}(p, d_2, r, ts_u)$ |
| **Multi-step casual dependencies from [23]** |
| $\mathsf{used}^+(p:\mathcal{P}, d:\mathcal{O}, ts_b:\mathcal{T}, ts_e:\mathcal{T}) = \exists p_i \in \mathcal{P}, r \in \mathcal{R}, ts_{b'} \in \mathcal{T} : \mathsf{wasTriggeredBy}^+(p, p_i, ts_{b'}, ts_e) \wedge \mathsf{used}(p_i, d, r, ts_b)$ |
| $\mathsf{wasGeneratedBy}^+(d:\mathcal{O}, p:\mathcal{P}, ts_b:\mathcal{T}, ts_e:\mathcal{T}) = \exists p_i \in \mathcal{P}, r \in \mathcal{R}, ts'_e \in \mathcal{T} : \mathsf{wasGeneratedBy}(d, p_i, r, ts_e) \wedge \mathsf{wasTriggeredBy}^+(p_i, p, ts_b, ts'_e)$ |
| $\mathsf{wasTriggeredBy}^+(p_1:\mathcal{P}, p_2:\mathcal{P}, ts_b:\mathcal{T}, ts_e:\mathcal{T}) = (\mathsf{wasTriggeredBy}(p_1, p_2, ts_e) \wedge (ts_b = ts_e)) \vee (\exists p_i \in \mathcal{P}, ts_{e'} \in \mathcal{T} : \mathsf{wasTriggeredBy}(p_1, p_i, ts_e)$ $\wedge \mathsf{wasTriggeredBy}^+(p_i, p_2, ts_b, ts'_e))$ |
| $\mathsf{wasDerivedFrom}^+(d_1:\mathcal{O}, d_2:\mathcal{O}, ts_b:\mathcal{T}, ts_e:\mathcal{T}) = (\mathsf{wasDerivedFrom}(d_1, d_2, r, ts_e) \wedge (ts_b = ts_e)) \vee (\exists d_i \in \mathcal{O}, r_i \in \mathcal{R}, ts_{e'} \in \mathcal{T} : \mathsf{wasDerivedFrom}(d_1, d_i, r_i, ts_e)$ $\wedge \mathsf{wasDerivedFrom}^+(d_i, d_2, ts_b, ts_{e'}))$ |
| **Additional casual dependencies** |
| $\mathsf{owns}(s:\mathcal{A}, d:\mathcal{O}, ts_g:\mathcal{T}) = \exists p \in \mathcal{P}, r \in \mathcal{R}, ts_b, ts_e \in \mathcal{T} : \mathsf{wasGeneratedBy}(d, p, r, ts_g) \wedge \mathsf{wasControlledBy}(p, s, \mathtt{owner}, ts_b, ts_e)$ |
| $\mathsf{contributedTo}(s:\mathcal{A}, d:\mathcal{O}, r:\mathcal{R}, ts_g:\mathcal{T}) = \exists p \in \mathcal{P}, ts_b, ts_e \in \mathcal{T} : \mathsf{wasGeneratedBy}^+(d, p, ts_g, ts) \wedge \mathsf{wasControlledBy}(p, s, r, ts_b, ts_e)$ |
| $\mathsf{notAvailable}(d:\mathcal{O}, ts:\mathcal{T}) = \exists p \in \mathcal{P}, r \in \mathcal{R} : \mathsf{used}(p, d, r, ts) \wedge \mathsf{action}(p) = \mathtt{delete}$ |

TABLE I. CASUAL DEPENDENCIES FOR OUR MODEL

instead gives rise to a new artifact. This new version is linked back to its predecessor through a wasDerivedFrom dependency with the role *update*, thereby maintaining the integrity of the evolution history.

*b) Specific attributes:* Our model relies on two main types of attributes. The first serves to distinguish artifacts that constitute personal data. Since the GDPR explicitly governs the processing of personal data, only these artifacts are subject to compliance checks; other data in the system remains outside its scope unless it embeds personal information. To capture this distinction, we introduce the Boolean attribute $dp$. For example, $dp(email\_bob) = True$ indicates that the artifact $email\_bob$ represents personal data.

The second type of attribute encodes the purposes for which consent has been granted. A purpose corresponds to an action performed by a process (e.g., purchase, information request) captured by the predicate *action*, introduced in the previous section.

Since consent links personal data to authorized purposes, this relation is captured within a dedicated $purposes$ attribute on consent artifacts. This attribute can be formalized as a list of tuples, each associating an artifact (personal data) with the set of authorized purposes. We denote the $purposes$ attribute of the $cons$ artifact as $purposes(cons) = [(artefact_1, list_1), ..., (artefact_k, list_k)]$.

When the system does not allow purpose specification at the level of individual artifacts, a simplified representation is used: $k = 1$ and $artifact_1 = \_$ , meaning that the consent purposes apply globally to all personal data.

## C. Running example

To illustrate our approach, we introduce a running example that will be used throughout the paper. Figure 1 depicts an online forum modeled as a provenance graph. In this system, users can create accounts to join discussion groups, where they are able to post and reply to messages. The forum adopts a global approach to purposes: consent is defined at the account level and does not depend on individual data items.

In the example, Bob creates an account (*create_account* process) and provides several pieces of personal information such as his e-mail address and telephone number, which are stored by the system. At registration, an identity number is automatically assigned, along with a personal wall and a friends list, each of these elements being considered personal data under the GDPR. Bob must also specify the purposes for which his data may be processed. Initially, he authorizes its use for both marketing (including third-party sharing) and analytics (service improvement), a choice represented in the graph by the artifact $consent\_bob\_v0$. Later, he revises his preferences by withdrawing consent for marketing while keeping analytics enabled. This update produces a new artifact, $consent\_bob\_v1$, linked to the previous one as described in the preceding section.

Subsequently, Bob joins a cooking interest group. At this point, the system generates a cookie, which is sent to an external vendor of kitchenware, despite Bob's updated consent. This illustrates a non-compliant behavior with respect to the principle of purpose limitation.

The graph also shows the role of the Data Controller ($DC$ agent). For example, Bob requests access to his personal data, and the controller provides a response. He later asks for the deletion of his phone number (*ask_erase* process), but this request remains unfulfilled, highlighting another case of non-compliance.

Notice that the graph, timestamps are expressed in minutes and, for readability reasons, node attributes (such as purpose or personal data markers) are not explicitly visualized in the provenance graph, even though they are formally associated with the corresponding artifacts.

*Example 4 (Purpose attribute):* In the scenario illustrated in Figure 1, purposes are defined globally. The $purpose$ attribute of the artifact $consent\_bob\_v0$ is expressed as:

$$purposes(consent\_bob\_v0) = \\ [(\_, ["thirdParties", "analysis", "improvement"])]$$

This notation means that Bob's consent applies uniformly to all his personal data, authorizing its use for third-party sharing, analysis, and service improvement.

Now consider an extended version of the forum that includes an integrated online shop. To better align with the principle of data minimization, the system allows users to grant different consent for each category of personal data. For instance, Bob may agree that his physical address, e-mail, and bank account can be used when purchasing products, but restrict refunds to rely solely on his bank account number. In this case, the consent attribute would be represented as follows:

$$purposes(consent\_bob\_v0) = [(name\_bob, ["buy", "refund"]), (address\_bob, ["buy"]), (email\_bob, ["buy"]), (bank\_account\_bob, ["buy", "refund"])]$$

This finer-grained modeling provides a more precise representation of user preferences, ensuring that data is processed strictly within the scope for which explicit consent has been granted.

Our work proposes to use provenance graphs to represent the evolution of the system over time and to model the principles and rights defined in the GDPR using patterns, defined as path expressions in the next section.

## V. Modeling principles as provenance patterns

Some of the GDPR principles introduced in Section III are directly related to data processing, which makes the compliance verification easier to be automated or semi-automated.

In this section we formalize the selected GDPR principles as patterns in a provenance graph representing the system execution history, then we focus on user rights verification. A pattern is a path expression composed of conjunctions and/or disjunctions of causal dependencies. Possibly, temporal constraints or constraints on attribute values can be included in patterns. The constraints on attribute values ensure that the value of the attribute in the graph corresponds to the value in the pattern. If the attribute value is a list, as in the case of the *purposes* attribute, the constraint is verified if the value belongs to the list.

When we use negation in our patterns, we follow the negation by failure principle, similar to what is used in Prolog. This means that a pattern is considered negated (i.e., false) if the system fails to find any instance that satisfies it. In other words, rather than proving that the negated condition is logically false, we assume it is false because no matching example can be found.

### A. Principles of the GDPR

In this section we start by formalizing the lawfulness principle, related to the notions of consent and purpose. Then, we model the storage limitation principle, related to the notion of legal retention limit.

*1) Consent and purpose limitation:* As previously introduced, consent and purpose are closely linked. Indeed, the GDPR stipulates in its basic principles that any data processing must be consented to and imposes purpose limitation as a principle. We present next how to formally define the patterns related to the purpose and the presence or absence of consent.

We write $\mathsf{isPurpose}(PU, D, C)$ if the purpose $PU$ has been consented for the personal data $D$ in the consent artifact $C$. If the piece of data is not specified, we simply write $\mathsf{isPurpose}(PU, \_, C)$, where _ stands for any personal data and the purpose is called *global*.

Personal data cannot be used if the data subject has not given consent for the intended purpose (Art. 6.1.a). To determine whether the data subject has consented to the use of his or her personal data $D$ for the purpose $PU$, there must be a consent artifact $C$ for which $PU$ is a purpose for $D$. The following path expression formalizes this:

$$\mathsf{consent}(C, D, PU, T) = \mathsf{wasControlledBy}(P1, S, "owner", TB, TE) \wedge \mathsf{wasGeneratedBy}(C, P1, "consent", T) \wedge \mathsf{isPurpose}(PU, D, C)$$

where $C$ represents the consent (artifact) created by the process "consent" $P1$ controlled by the data subject $S$ as owner of the personal data $D$. The timestamps $T$, $TB$, and $TE$ are included in the pattern because there may be multiple instances of the consent $C$ within the graph. The consent may correspond either to the one initially provided by the data subject (as represented by the `consent` process in Figure 1) or to a subsequent modification (the `update` process in the same figure). Once the value of $T$ is instantiated, it uniquely identifies a specific occurrence of $C$, allowing it to refer to a single artifact.

*Example 5 (Pattern for consent):*

In the scenario presented in Figure 1, the pattern $\mathsf{consent}(consent\_bob\_v0, email\_bob, "analysis", T)$ is evaluated to true.

This is because the path

$$\mathsf{wasControlledBy}(\texttt{consent}, Bob, "\texttt{owner}", 16, 20) \wedge \mathsf{wasGeneratedBy}(consent\_bob\_v0, \texttt{consent}, "\texttt{consent}", 17)$$

exists in the provenance graph. The consent artifact *consent_bob_v0* contains the global purpose *"analysis"* and the timestamp of its generation is indeed 17.

*2) Consent revocation and update:* As stated in Article 7.3 of the GDPR, the data subject must be able to withdraw his/her consent at any time. Consent revocation is modeled using the predicate *revoke* and the following pattern:

$$\mathsf{revoke}(C, T) = \mathsf{used}(P, C, "revokeConsent", T)$$

where $C$ refers to the consent artifact, $T$ is the timestamp indicating the time of revocation, and $P$ denotes the process responsible for revoking the consent. Since the revocation applies to all purposes of all associated artifacts, the predicate is independent of any specific data or purpose.

Instead of being entirely revoked, consent preferences can be modified by the data subject, either to add or remove purposes. In such cases, a new artifact is created in the provenance graph, derived from the previous consent artifact, through a process that performs an update initiated by the data subject.

The following pattern allows us to verify whether a consent artifact $C$ has been updated by the data subject $S$ at a given time $T$:

$$\mathsf{nextConsent}(C, C1, T) = \mathsf{wasControlledBy}(P1, S, "owner", TB, TE) \wedge \mathsf{used}(P1, C, "updateConsent", TU) \wedge \mathsf{wasGeneratedBy}(C1, P1, "consent", T)$$

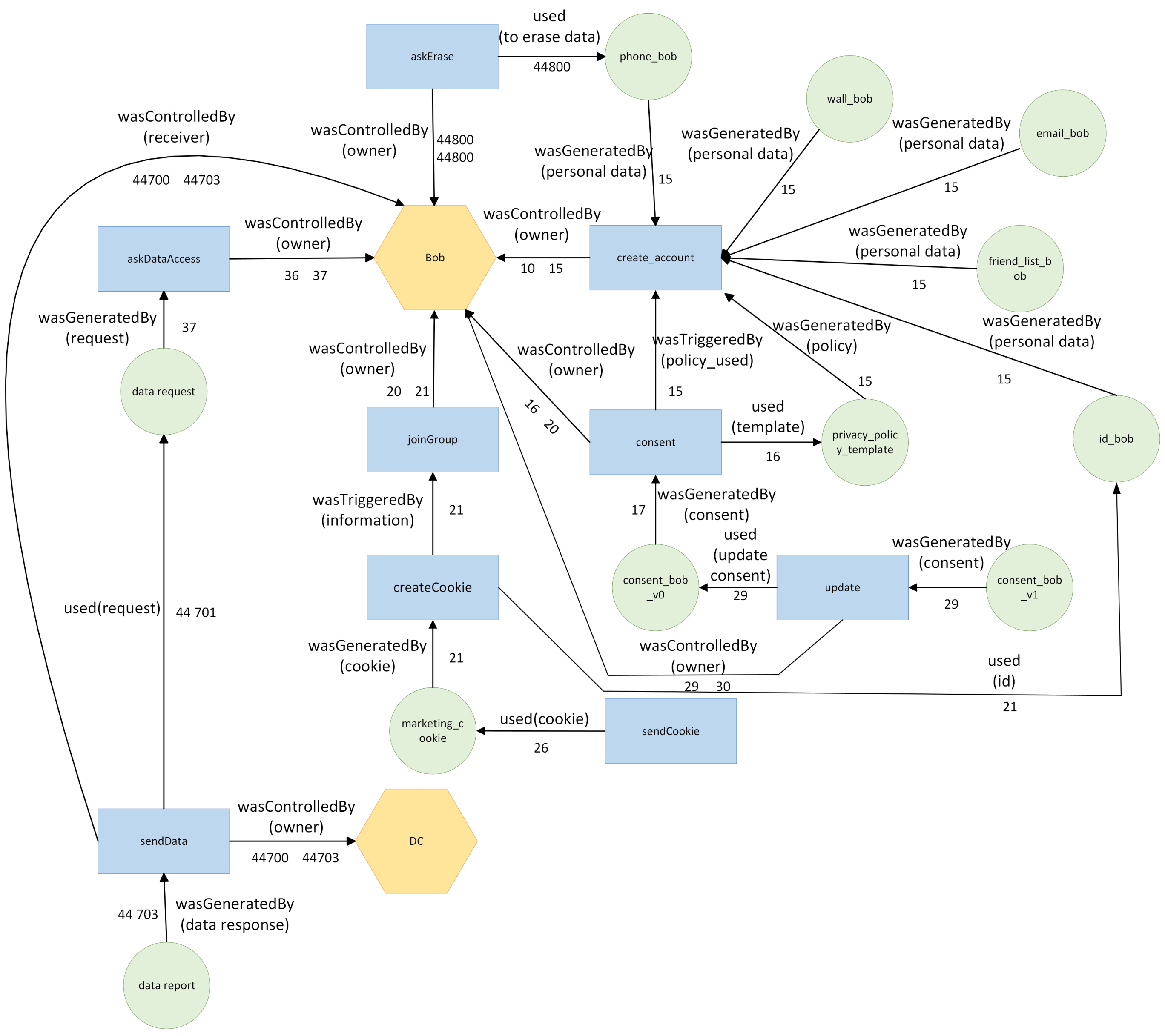


Figure 1. Extract from the provenance graph of an online forum.

Notice that $\mathsf{nextConsent}(C, C1, T)$ refers to the artifact representing the update that directly follows the consent version denoted by $C$. It is important to note that since artifacts represent immutable objects, any modification to the original object leads to the creation of a new artifact. Consequently, for a given $C$, there can be only one corresponding instance of $C1$.

We can easily verify whether a consent artifact is the latest existing version, using the predicate *lastConsent* and the following pattern:

$$\mathsf{lastConsent}(C) = \mathsf{consent}(C, D, PU, T) \wedge \neg(\mathsf{nextConsent}(C, C1, TU))$$

As explained earlier, $\neg(\mathsf{nextConsent}(C, C1, TU))$ is interpreted as negation by failure. This means the pattern $\mathsf{lastConsent}(C)$ holds true if $C$ represents a consent and there is no pair $(C1, TU)$ such that $\mathsf{nextConsent}(C, C1, TU)$ exists in the graph. In other words, $C$ is considered the latest consent definition when no subsequent update artifact follows it.

*Example 6 (Consent update):*

In the provenance graph shown in Figure 1, Bob has updated his consent once, resulting in the creation of a new artifact, $consent\_bob\_v1$. The pattern $\mathsf{nextConsent}(C, C1, T)$ can be instantiated with $C = consent\_bob\_v0$, $C1 = consent\_bob\_v1$, and $T = 29$. Consequently, $\mathsf{lastConsent}(consent\_bob\_v0)$ evaluates to False, while $\mathsf{lastConsent}(consent\_bob\_v1)$ evaluates to True.

*3) Lawfulness:* As stated in Article 5 of the GDPR, an artifact is considered to have been used lawfully if it was

| Article | Pattern | Description |
|---|---|---|
| Principle: Purpose Limitation | | |
| 6.1(a) | $\mathsf{consent}(C, D, PU, T)$ | consent C given for data D to be used for PU purpose at time T |
| 7.3 | $\mathsf{revoke}(C, T)$ | consent C revoked at time T |
| 7.3 | $\mathsf{nextConsent}(C, C1, T)$ | updated consent C into C1 at time T |
| 7.3 | $\mathsf{lastConsent}(C)$ | C has no updates |
| Principle: Lawfullness | | |
| - | $\neg\mathsf{isPersonal}(\mathsf{D})$ | artifact D is not personal data and doesn't derived from personal data |
| 5.1(a) | $\mathsf{validConsent}(D, PU, C, T, TG, TU)$ | consent C for D and PU valid at time T |
| 5.1(a) | $\mathsf{validConsent}^{+}(D, PU, C, T, TG, TU)$ | consent C for personal data from which D is derived and PU valid at time T |
| 5.1.(a) | $\mathsf{legal}(P, D, C, TG, TU)$ | treatement of D by process P lawfull |
| Principle: Storage Limitation | | |
| - | $\mathsf{usageInLimit}(D, P)$ | process P used D during the authorized interval |
| - | $\mathsf{delationComplete}(D)$ | D has been deleted before the end of the authorized period |
| - | $\mathsf{retentionValid}(D)$ | authorized period to use D not expired |
| 5.1(e) | $\mathsf{storageLimitation}(D)$ | compliance of the storage limitation for D |
| Rigth: To Be Forgotten | | |
| - | $\mathsf{askErase}(S, D, T)$ | request from S to delete D at time T |
| - | $\mathsf{eraseNotDoneYet}(D)$ | D not deleted yet but still in the allowed time |
| - | $\mathsf{erased}(\mathsf{D})$ | D erased during authorized period |
| - | $\mathsf{copiesErased}(\mathsf{D})$ | D replications erased during authorized period |
| 17(a) | $\mathsf{eraseComplete}(D)$ | compliant to the erase principle for the request on D |
| Right: To Access | | |
| 15.1 | $\mathsf{askAccess}(U, TB, TE)$ | user U asked for access at time TE |
| 15.1 | $\mathsf{rigthAccess}(U)$ | U's request answered in time |

TABLE II. PATTERNS EXPRESSING GDPR PRINCIPLES AND RIGHTS

processed solely for purposes to which the data subject has given valid consent. In the context of a provenance graph, this means that all paths leading from a consent artifact to a process must demonstrate that the processing was authorized.

More precisely, for each process $P$ that uses a data artifact, the provenance path must confirm two conditions: the data subject had provided valid consent for the specific purpose at the time of processing, and this consent had not been revoked during the execution of $P$.

From the structure of the graph, two lawful processing scenarios can be distinguished:

- Non-personal data path: For the artefact used by $P$ not to contain personal data, it is required that the artefact is not classified as personal data, with the attribute $dp = \mathit{False}$. it is also necessary that the artefact was derived only from data nodes annotated with the attribute $dp = \mathit{False}$. In this case, no personal data is involved, and thus no consent is required. This scenario corresponds to the path:

$$\neg\mathsf{isPersonal}(\mathsf{D}) = (\mathsf{dp}(\mathsf{D}) = \mathsf{False})\ \wedge \\ \forall D'\big(\mathsf{wasDerivedFrom}^{+}(D, D', TS, TE) \\ \wedge\ (\mathsf{dp}(\mathsf{D}') = \mathsf{False})\big)$$

- The artifact used by process $P$ is personal data $D$, processed for a specific purpose $PU$, and:
  i) There exists a consent artifact $C$ such that the data subject has given valid consent for purpose $PU$, and this consent was still valid (i.e., not revoked) at the time $D$ was used by $P$. In this case, the data $D$ is personal and is processed for a specific purpose $PU$ by a process $P$. The processing is lawful if there exists a valid consent $C$ that applies at the time of processing $T$.
  Consent is considered valid at time $T$ if any of the following hold: consent $C$ was given and never revoked; consent $C$ was given and revoked, but after time $T$; consent $C$ was updated after time $T$, meaning the version valid at $T$ was still in force. This is expressed by the following path expression:

$$\mathsf{validConsent}(D, PU, C, T, TG, TU) = \\ \big(\mathsf{consent}(C, D, PU, TG) \wedge (TG < T) \wedge \mathsf{lastConsent}(C) \\ \wedge\neg\big(\mathsf{revoke}(C, TU)\big)\big)\vee \\ \big(\mathsf{consent}(C, D, PU, TG) \wedge (TG < T) \wedge \mathsf{lastConsent}(C) \\ \wedge\mathsf{revoke}(C, TU) \wedge (TU > T)\big)\vee \\ \big(\mathsf{consent}(C, D, PU, TG) \\ \wedge(TG < T) \wedge \mathsf{nextConsent}(C, C1, TG1) \wedge (TG1 > T)\big)$$

  ii) $D$ was derived from other personal data artifacts $D'$, and each of those $D'$ has an associated valid consent for the same purpose $PU$ at the time $D$ is processed by $P$. The corresponding path expression is as follows:

$$\mathsf{validConsent}^{+}(D, PU, C, T, TG, TU) = \forall D' \\ (\mathsf{wasDerivedFrom}^{+}(D, D', TS, TE) \wedge dp(D') = \mathtt{False})\vee \\ (\mathsf{wasDerivedFrom}^{+}(D, D', TS, TE) \wedge\ dp(D') = \mathtt{True}\ \wedge \\ \mathsf{validConsent}(D', PU, C, T, TG, TU))$$

In both cases, the provenance graph provides a traceable path that allows us to verify compliance by following the data lineage and matching it with the consent.

Summarizing, the process $P$ on an artifact $D$ has been carried out legally if and only if the following pattern finds an instantiation in the provenance graph:

$$\neg\mathsf{isPersonal}(\mathsf{D}) \vee (\mathsf{used}(\mathsf{P}, \mathsf{D}, \mathsf{R}, \mathsf{T}) \wedge \mathsf{action}(\mathsf{P}) = \mathsf{PU} \\ \wedge\ \mathsf{validConsent}^{+}(D, PU, C, T, TG, TU)\ \wedge \\ dp(D) = \mathtt{True}\ \ \wedge \mathsf{validConsent}(D, PU, C, T, TG, TU))$$

Here $P$ is the process that performed an action $PU$ on the artifact $D$, $C$ is the consent of the artifact containing the consent to perform $PU$, $R$ represents a role and

$T, TG, TU, TG1$ are timestamps. This path expression is denoted $\mathsf{legal\_}DP(P, D, C, TG, TU)$. An action is lawful if consent has been given for all personal data used by $P$ before the action is performed. If the data is not of a personal nature (first line of the expression), the action is automatically considered to comply with this principle.

*Example 7 (Lawful processing):* In the scenario depicted in Figure 1, a marketing cookie is created at time $T = 21$ by the process $createCookie$. This process operates under the purpose *thirdParties*, which refers to the creation of cookies intended for third-party use. To do so, it uses Bob's identity on the website, denoted $id\_bob$. This processing is considered lawful if and only if at least one of the conditions defined previously is satisfied. In this case, it is the third condition (involving updated consent) that applies.

The assignment $P = createCookie$, $PU = thirdParties$, $D = id\_bob$, $T = 21$, $R = id$, $C = consent\_bob\_v0$, $C = consent\_bob\_v1$, $TG = 17$ and $TG1 = 29$ satisfies the third clause of the disjunction, since the original consent $consent\_bob\_v0$ was given at $TG = 17$ and updated by $consent\_bob\_v1$ at $TG1 = 29$, which occurs after the data usage at $T = 21$. As a result, the consent was still valid when the data was used, and the creation of the cookie is therefore lawful.

*4) Storage limitation:* The retention of personal data is governed by Article 5.1(e) of the GDPR, which states that personal data should not be kept longer than necessary for the purposes for which it is processed, except in specific cases such as archiving in the public interest.

In practice, this means that personal data should not be stored without use for a period exceeding the retention limit defined by the system, based on its operational requirements.

A system respects this storage limitation principle for a given piece of personal data $D$ if the data has not remained unused beyond the authorized retention duration. More specifically, there must be no interval between two consecutive uses of $D$ that exceeds the maximum permitted retention time ($TLIMIT$).

This condition can be expressed by the following pattern:

$$\begin{aligned}&\mathsf{usageInLimit}(D, P) = \mathsf{used}(P, D, R, T) \wedge \\ &((\mathsf{used}(P', D, R', T') \wedge (T' > T) \wedge (T' - T < TLIMIT)) \\ &\vee \neg(\mathsf{used}(P', D, R', T') \wedge (T' > T)))\end{aligned}$$

The negation here is interpreted again using negation by failure. That is, for a process $P$ that uses $D$, either there exists another process $P'$ that also uses $D$ within a time interval not exceeding the retention period, or $P$ is the last process to have used $D$. The verification of whether the retention period is still valid after the last use is performed by the predicate. retentionValid.

Between the last use of the personal data artifact $D$ and the current time, two conditions can ensure compliance with the GDPR's storage limitation principle (Article 5.1(e)): either the data has been explicitly deleted, i.e., there exists a deletion event $\mathsf{notAvailable}(D, T)$ recorded in the provenance graph; or the data is still retained, but the defined retention period has not yet been exceeded, meaning the condition $TCURRENT - TU < TLIMIT$ holds, where $TU$ is the last time the data was used.

The path expression confirming that the data was previously deleted is as follows.

$$\begin{aligned}&\mathsf{delationComplete}(D) = \mathsf{used}(P, D, R, TU) \wedge \\ &\mathsf{notAvailable}(D, T) \wedge (T - TU <= TLIMIT)\end{aligned}$$

In the case that the data has not been deleted yet and the retention period has not been exceeded, the following path expression will have a match in the graph.

$$\begin{aligned}&\mathsf{retentionValid}(D) = \\ &\mathsf{used}(P, D, R, TU) \wedge \neg(\mathsf{notAvailable}(D, T)) \\ &\wedge (TCURRENT - TU <= TLIMIT)\end{aligned}$$

In both cases, there is no need to verify that $P$ is the last process to have used $D$. If the path exists for some process $P$, the time condition also holds for the actual last process $P'$, since $P'$ was executed more recently than $P$. This ensures that the data is either used regularly within the permitted retention period or deleted once that period has elapsed, thereby guaranteeing compliance with the storage limitation principle.

A system is compliant with the storage limitation principle for a personal data artifact $D$, if the following pattern has an instantiation:

$$\begin{aligned}&\mathsf{storageLimitation}(D) = \forall P(\mathsf{usageInLimit}(D, P) \wedge \\ &(\mathsf{delationComplete}(D) \vee \mathsf{retentionValid}(D)))\end{aligned}$$

This ensures that personal data is never retained passively beyond its legal retention limit.

*Example 8 (Limitation of personal data storage):* Suppose that the forum shown in Figure 1 has a data retention period of 5 years. Bob has not logged into his account for 7 years. His email address and all his personal data have not been used since his last login. This situation is represented in the graph by the path $\mathsf{used}(P, email\_bob, R, TU) \wedge \neg(\mathsf{used}(P', email\_bob, R', TU') \wedge (TU < TU'))$ If the system is compliant, we would find the dependency $\mathsf{notAvailable}(email\_bob, T)$ with $T \in [TU, TU + TLIMIT]$. The sub-pattern $\mathsf{delationComplete}(email\_bob)$ evaluates to True.

In the case where the system fails to delete the data, the pattern $\mathsf{delationComplete}(emai\_bob)$ would evaluate to false, as would the pattern $\mathsf{retentionValid}(email\_bob)$, because the retention period has been exceeded.

It is important to emphasize that these patterns yield a Boolean result depending on whether a matching instantiation exists in the provenance graph. However, this result must be interpreted with care: there is a difference between actual compliance, meaning that the processing is and will remain compliant over time, and the absence of non-compliance simply because a retention deadline has not yet been reached.

In the latter case, the pattern does not confirm lasting compliance, but rather the current absence of a violation. Therefore, an a posteriori check may be required.

For instance, if the verification in the previous example were performed three years after the last use of Alice's email, the pattern would return true, indicating compliance at that moment. However, it would still be necessary to later verify that the email was either deleted before the retention limit expired, or reused within the allowed time frame.

### *B. User rights*

Just as certain articles related to data protection principles can be automatically verified through pattern-based approaches, the same applies to some rights, or at least to specific provisions within the articles that define them. In this section, we illustrate this with two key rights: the right to be forgotten and the right of access.

As with the principles, pattern-based detection may only identify a single instance of non-compliance, necessitating further verification over time. Moreover, some articles require additional manual checks, especially when they impose conditions on the content to be provided to individuals, or when they involve requirements concerning the clarity, granularity, or comprehensibility of the explanations and notifications to be delivered.

*a) Right to be forgotten:* In accordance with Article 17a of the GDPR, data subjects have the right to request the erasure of certain personal data, and data controllers are obliged to comply with such requests within a specified time frame.

To verify whether the right to erasure (Article 17a of the GDPR) has been correctly exercised, we first identify a relevant pattern in the provenance graph. Specifically, we look for a process $P$ performing the action *askErase* on a data item $D$ under the control of a user $U$ identified as the data owner. This can be captured by the following path in the graph $\mathsf{used}(P, D, R, T) \wedge \mathsf{wasControlledBy}(P, U, "owner", TB, TE)$ where $T, TB, TE$ are timestamps, and the action associated with process $P$ is *askErase*. We define the $askErase$ pattern to formalize this detection as follows:

$$\begin{aligned}&\mathsf{askErase}(S, D, T) = \mathsf{owns}(S, D, TU) \wedge \mathsf{used}(P, D, R, T) \wedge \\ &\mathsf{wasControlledBy}(P, S, R', TB, TE) \wedge \\ &\mathsf{action}(P, "askErase") \wedge \\ &\neg\big(\mathsf{used}(P', D, R, T') \wedge \mathsf{action}(P', "askErase") \wedge (T' < T)\big)\end{aligned}$$

This pattern identifies the earliest request for deletion of data $D$, ensuring that we retrieve the first occurrence of such a request in time, i.e there exists no prior instance of a process $P'$ asking for the erasure of the data. Since a user may issue multiple deletion requests before receiving a response, capturing only the initial request is essential to assess whether the controller met the deadline and thus complied with the regulation.

A system is considered compliant if and only if the requested data and all its copies are deleted within the authorized time frame. The deletion of a data item $D$, following a user's request, is captured by the pattern

$$\begin{aligned}&Erased(D) = \mathsf{askErase}(S, D, T) \wedge \\ &\mathsf{notAvailable}(D, TU) \wedge (TU - T < TLIMIT)\end{aligned}$$

with $TLIMIT$ being the deadline for performing the action in a compliant manner. Here $\mathsf{askErase}(D, T)$ denotes that the first request for erasure of $D$ occurred at time $T$, $\mathsf{notAvailable}(D, TU)$ indicates that $D$ became unavailable (i.e., deleted) at time $TU$, $TLIMIT$ is the regulatory deadline for performing the deletion.

To ensure compliance, it is not sufficient to delete only the original data item; all copies of that data must also be deleted. The deletion of derived or copied data $D$, resulting from a copy operation on $D$, is expressed by the following condition:

$$\begin{aligned}&copiesErased(D) = \mathsf{askErase}(D, T) \wedge \\ &\forall D'(\mathsf{wasDerivedFrom}^{+}(D, D', "copy", TU', TG) \wedge \\ &\mathsf{notAvailable}(D', TU') \wedge (TU' - T < TLIMIT))\end{aligned}$$

This states that for all data artifacts $D'$ derived from $D$ via one or more copy operations (denoted by $\mathsf{wasDerivedFrom}^{+}$), $D'$ must also be made unavailable (i.e., deleted) within the same time constraint relative to the original request time $T$.

We can define full compliance for a data item $D$ with respect to Article $17a$ using the following predicate:

$$\begin{aligned}&\mathsf{eraseComplete}(D) = \\ &Erased(D) \wedge copiesErased(D)\end{aligned}$$

This formalization ensures that both the original data and all its copies are deleted in a timely manner, satisfying the obligations imposed by Article 17a.

However, as previously mentioned, a compliance check can be performed at any time. This implies that there may be cases where a deletion request has been submitted, but the corresponding data has not yet been deleted, while still being within the allowed deadline.

We define the case of a pending deletion request using the pattern :

$$\begin{aligned}&\mathsf{eraseNotDoneYet}(D) = \\ &\mathsf{askErase}(S, D, T) \wedge \neg(\mathsf{notAvailable}(D, TU)) \wedge \\ &(TCURRENT - T < TLIMIT)\end{aligned}$$

where $TCURRENT$ represents the date on which the verification is carried out. $\mathsf{askErase}(S, D, T)$ indicates that a deletion request for data $D$ was made at time $T$ by the owner $S$; $\neg\mathsf{notAvailable}(D, TU)$ indicates that the data $D$ is still available at time $TCURRENT$; $TLIMIT$ is the legal deadline to complete the deletion.

This pattern captures the intermediate state where the data has not yet been deleted, but the deadline has not been exceeded (hence, the system is not yet non-compliant).

Finally, to verify whether the system is globally compliant with respect to the deletion request for a personal data item $D$, we define the following pattern:

$$\begin{aligned}&\mathsf{eraseCompliant}(D) = \\ &(\mathsf{askErase}(S, D, T) \wedge \mathsf{eraseComplete}(D)) \vee \\ &\mathsf{askErase}(S, D, T) \wedge \mathsf{eraseNotDoneYet}(D)\end{aligned}$$

This expression states that compliance is satisfied either because the data and all its derived copies have been deleted within the deadline ($\mathsf{eraseComplete}(D)$), or because the request has been issued but the allowed deadline has not yet expired ($\mathsf{eraseNotDoneYet}(D)$).

Therefore, our framework allows us to distinguish between full compliance, ongoing compliance (pending fulfillment), and non-compliance (which occurs when the deadline has passed without deletion).

*Example 9 (Right to be forgotten):* We consider a scenario derived from the system shown in Figure 1. According to Article 17a of the GDPR, a data controller must delete a data subject's personal data within a specified period after receiving a valid erasure request. Suppose that this deadline is set to 30 days. Given that timestamps are expressed in minutes, this results in a deadline of $TLIMIT = 30 \times 24 \times 60 = 43.200$.

Bob, after receiving his personal data report from the forum, submits a request to delete his phone number $phone\_bob$. The provenance graph records this request at timestamp $T = 44.800$.

At the time of the compliance verification, 12 days have passed since Bob sent the request, i.e., $TCURRENT = 61.983$. We compute the elapsed time: $TCURRENT - T = 61.983 - 44.800 = 17.183 < TLIMIT$. This means that the request has been made and is still within the allowable period for compliance. As a result, the following instantiation holds: $\mathsf{eraseCompliant}(phone\_bob) = \mathsf{askErase}(''bob'', phone\_bob, 44800) \wedge \mathsf{eraseNotDoneYet}(phone\_bob)$.

The system is therefore considered compliant at this stage, even though the deletion has not yet been executed. However, a follow-up check must be performed after the deadline to ensure that the data, and all its copies, were actually erased in time.

*b) Right of access:* The GDPR grants data subjects the right to access all personal data held about them and to understand how their information has been processed. Article 15.1 specifies the obligations of the data controller in responding to such requests. While some aspects of this compliance check can be automated, it cannot be fully verified through event logs alone, as these do not capture the content of the response document. In this context, the defined predicates can verify that a response was issued following a request, but they do not guarantee that the response includes all the required information.

As with the right to be forgotten, an access request has been made if the following path is present in the graph:

$$\begin{aligned}&\mathsf{askAccess}(U, TB, TE) = \\ &\mathsf{wasControlledBy}(P, U, ''owner'', TB, TE) \wedge \\ &\mathsf{action}(P, ''askAccess'') \wedge used(P, D, R, TU) \wedge \\ &dp(D) = True \wedge \mathsf{owns}(U, D, TU') \wedge \\ &\neg\big(\mathsf{wasControlledBy}(P', U, ''owner'', TB', TE') \wedge \\ &\mathsf{action}(P', ''askAccess'') \wedge (TE' < TE)\big)\end{aligned}$$

The pattern $\mathsf{askAccess}(U, TE)$ retrieves the timestamp $TE$ corresponding to the first access request initiated by the data subject $U$.

The system is considered compliant with the right of access if the data controller $S'$ provides a response to the request within a legally defined time limit $TLIMIT$. Compliance can be verified if at least one of the two disjunctive conditions in the following pattern is instantiated in the provenance graph:

$$\begin{aligned}&\mathsf{rigthAccess}(U) = \big(\mathsf{askAccess}(U, TB, TE) \\ &\wedge \mathsf{wasControlledBy}(P', S', R, TB', TE') \\ &\wedge \mathsf{action}(P', ''sendData'') \wedge (TE' - TE < TLIMIT)\big) \vee \\ &\big(\mathsf{askAccess}(U, TB, TE) \wedge \\ &(TCURRENT - TE < TLIMIT)\big)\end{aligned}$$

Here, $P$ and $P'$ denote processes, and $TB$, $TB'$, $TE$, and $TE'$ represent timestamps. The first disjunct checks whether the response has already been sent in time, while the second allows for the possibility that the request has been made but the deadline has not yet passed.

The example that follows illustrates its application to the scenario depicted in Figure 1.

*Example 10 (Right of access):* Suppose that, in the case of the online forum, the data controller has a legal obligation to respond to a data access request within 30 days, which corresponds to a time limit of $TLIMIT = 43.200$ minutes.

According to the provenance graph, Bob submitted a data access request labeled $data\_request$ at timestamp $TE = 37$, and later received a response labeled $data\_report$ at timestamp $TE' = 44.730$. This sequence satisfies the structural requirements of the first conjunction in the pattern $\mathsf{rigthAccess}(U)$, as it includes the expected control and action events.

However, the time constraint is not satisfied: the response was provided $44.730 - 37 = 44.693$ minutes (approximately 32 days) after the initial request, which exceeds the allowed time limit of 30 days. Therefore, the system fails to comply with Article 15 of the GDPR in this instance and Bob's right of access was not respected.

## VI. Compliance verification

In this section, we present a prototype tool designed to assist in the verification of GDPR compliance. In this work, we consider provenance graphs as input. We do not address the problem of constructing provenance graphs from traces or event logs. In practice, this task is non-trivial, as logs differ widely in structure, semantics, and level of detail, and therefore require context-specific parsing and mapping choices. Developing such extraction pipelines is outside the scope of the present work and constitutes an interesting direction for future research. We focus here on the verification aspects of the framework. The tool allows users to select specific principles and rights to examine, offering a flexible and targeted approach to auditing. In addition to verifying compliance, the tool provides informative feedback by flagging situations that may warrant further investigation or future monitoring. To support a more thorough audit process, it also extracts contextual data linked to potential non-compliance, enabling auditors to better understand the origin and scope of detected issues. This facilitates not only initial compliance checks but also ongoing assessments and documentation.

### A. Methodology

To demonstrate and evaluate our approach, we rely on the case study representing a simplified online forum system of Section IV-C. This system is modeled as a provenance graph, where nodes represent data, users, system processes, and consent artifacts, and edges capture data flows and dependencies.

Our methodology proceeds as follows:

- For each relevant GDPR principle or data subject right (e.g., purpose limitation, consent, right of access, right to erasure), we define one or more formal compliance patterns. These patterns express the structural and temporal requirements that must be satisfied within the provenance graph. A summary of the patterns associated with each principle is given in Table II.
- We then encode both the provenance graph and the compliance patterns in a logic programming language (Prolog). The provenance information is translated into a set of facts, while the compliance patterns are encoded as rules using the templates defined in Section V.
- Our verification tool uses a Prolog solver to check whether the patterns are satisfied by the graph. If a pattern is not satisfied, the system identifies the precise point of failure and, when possible, provides suggestions for further analysis or actions needed to reach compliance.
- This approach supports both automated compliance verification and semi-automated auditing. For example, in the forum case study, the system can verify whether Bob's updated consent ($consent\_bob\_v1$) is correctly enforced in subsequent data usage, or whether his request for erasure has been processed within a reasonable timeframe.

This logic-based methodology enables precise, explainable, and extensible analysis of compliance with GDPR requirements, using provenance data as evidence.

## B. Prototype Architecture

We have developed a Java-based auditing tool that verifies GDPR compliance by encoding the formal patterns from the previous section in Prolog [25]. The tool integrates a reasoning engine to analyze data processing workflows and identify potential violations. Our approach allows us to benefit from the efficient reasoning capabilities provided by Prolog solvers for path condition resolution rather than relying on an ad-hoc algorithm. In particular, we implemented rules for resolving path queries based on the obtained provenance graph.

The tool reports non-conformities, along with warnings that suggest further verifications when the analysis cannot be concluded at current time. Auditors can configure the analysis by selecting specific GDPR principles, relevant data types, and processing steps, enabling fine-grained and targeted audits. A preliminary version of this tool was introduced in [1].

The main components of the prototype are illustrated in Figure 2. Through the interface, the auditor selects the system to audit and specifies the GDPR principles, personal data, or data subjects to be analyzed. These inputs are translated into Prolog queries by a dedicated translator module. The solver then executes the queries over the provenance graph representing the system's data flows and returns the results to the interface. The output includes detailed information about any detected non-conformities.

We describe next the architecture of our prototype, explaining how its components interact and how the patterns have been adapted to deliver precise feedback to the auditor.

*a) Interface:* The interface, developed in Java using the JavaFX graphics library, allows auditors to configure the scope of the audit. They can select the system to analyze and optionally restrict the verification to specific processes, personal data, or data subjects whose rights should be checked. The system's activity must be provided as a provenance graph, encoded as a Prolog file, which is loaded through the interface.

The auditor can further refine the audit by choosing the GDPR principles to be verified or focusing on specific components rather than conducting a full-system analysis (which is the default behavior). These configuration options are sent to the translator module (see Figure 2), which generates the corresponding Prolog queries. The available audit options are presented through a menu, as shown in Figure 3.

The results returned by the solver are finally displayed to the auditor in textual form (Figure 4).

*b) Translator:* The translator module serves two main purposes. First, it converts the auditor's selections into parameterized Prolog queries. Second, it extracts relevant elements (such as personal data, data subjects, and processes) from the provenance graph. These extracted entities are used both to populate the interface with appropriate choices and to instantiate variables in the generated queries.

Based on the selected principles and nodes, the translator generates a list of Prolog queries and sends them, along with the system data, to the Prolog solver using the JPL library. For instance, in the case study shown in Figure 1, if the auditor wants to verify all GDPR principles related to Bob's email address, namely the lawfulness of processing, the right to erasure, and storage limitation, the translator produces the following queries:

$$\begin{array}{l}\mathsf{legal}(P, "\texttt{email\_bob}", C, TG, T).\\ \mathsf{eraseComplete}("\texttt{email\_bob}").\\ \mathsf{storageLimitation}("\texttt{email\_bob}").\end{array}$$

In the case of the principle of lawfulness, the verification applies to all processes $P$ involving Bob's email. The translator therefore also sends a list of relevant processes to the solver, enabling it to instantiate the variable $P$ during query evaluation.

*c) Reasoning module:* The reasoning module integrates the Prolog solver, which is responsible for verifying path queries on the provenance graph. It also includes the formal definitions of GDPR predicates, each encoding a principle or a right, as introduced in the previous section.

When a path query is received from the translator, the reasoning module constructs the corresponding Prolog goal and invokes the solver to evaluate it. The solver applies the deduction rules (i.e., the Prolog program encoding causal dependencies and compliance patterns) to prove or refute the query.

Each GDPR principle is implemented as a Prolog rule, systematically decomposed into conjunctions and disjunctions of sub-predicates to enable granular compliance checking and

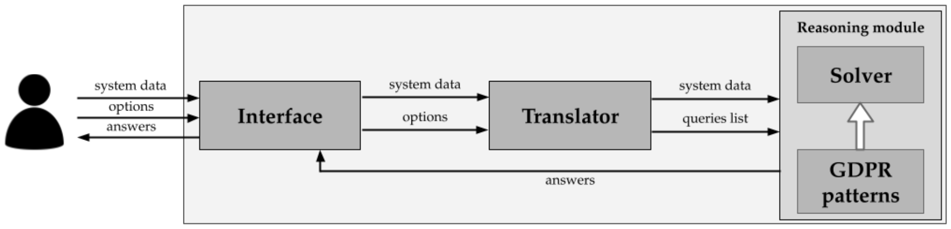
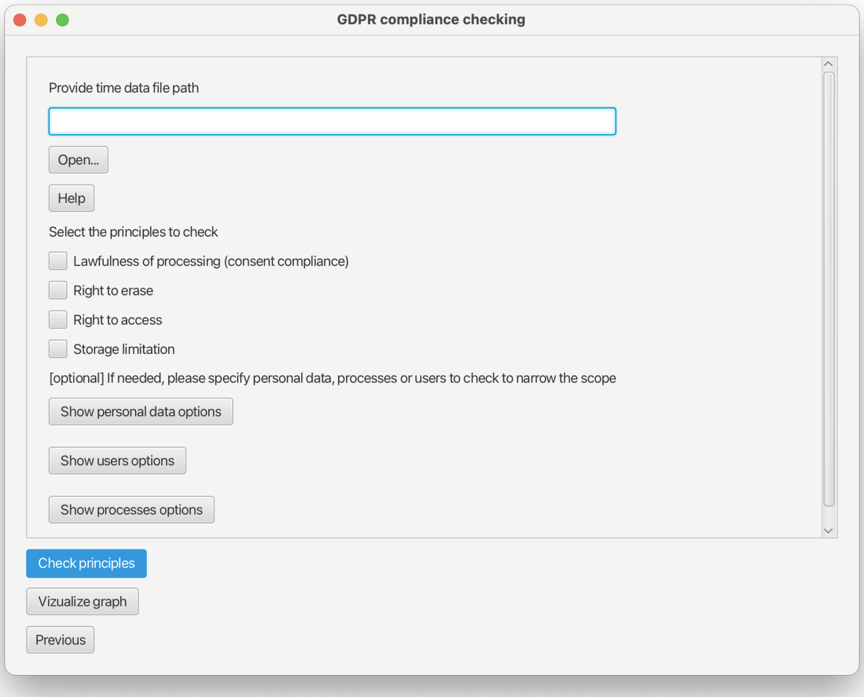


Figure 2. Tool Architecture


GDPR compliance checking

Provide time data file path

Open...

Help

Select the principles to check

- [ ] Lawfulness of processing (consent compliance)
- [ ] Right to erase
- [ ] Right to access
- [ ] Storage limitation

[optional] If needed, please specify personal data, processes or users to check to narrow the scope

Show personal data options

Show users options

Show processes options

Check principles

Vizualize graph

Previous

Figure 3. Prototype interface

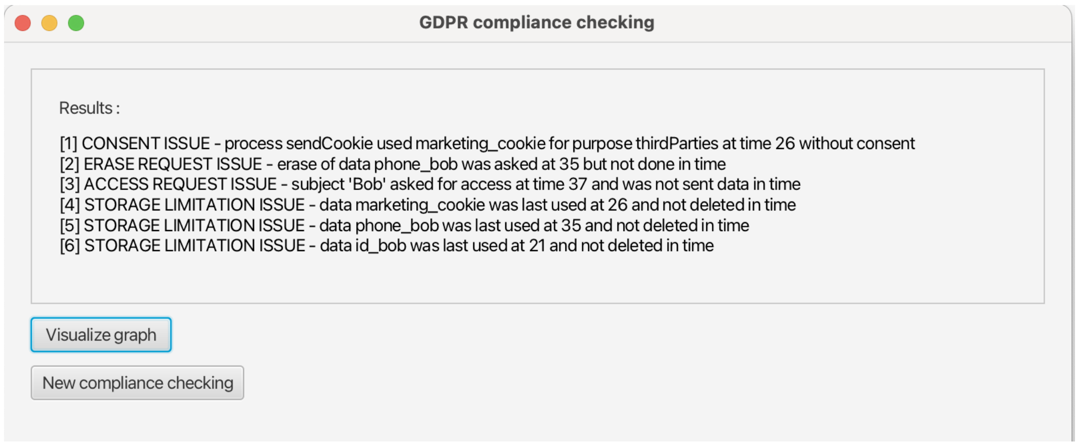
more informative feedback to the auditor. These rules typically follow the structure:

```
predicate(parameters) :-
( parameter check,
( compliance check; (deadline check,
 notify future verification);
(\+ compliance check, \+ deadline check,
notify non-compliance) ));
(\+ parameter check,
 notify missing parameters).
```

where \+ denotes negation. This structure ensures that, for each query, the solver can determine not only whether a principle is satisfied, but also whether additional verification is needed, or whether a compliance failure should be reported.

The patterns introduced in Section V are decomposed into Prolog sub-rules to provide interpretable feedback to the auditor. Each GDPR principle or right is defined through a main predicate composed of several sub-predicates, each serving a specific verification function:

- `check parameters`: This sub-predicate restricts verification to the relevant nodes in the graph. For instance, the principle of lawfulness only applies to personal data. A rule like `used(P,D,R,T), action(P) = PU, isPersonal(D)` ensures that the predicate `legal(P,D,C,TG,T)` is only applied if $D$ is personal data (i.e., `dp(D) = True`) and if $P$ is a process under audit that actually uses $D$.
- `verification compliance` and `verification deadline`: These sub-predicates form the core of the compliance patterns. The first verifies whether the required conditions are satisfied (e.g., whether a request has been processed), while the second checks whether the system is still within the allowed timeframe for fulfilling a request. If the action has not yet occurred but the deadline has not passed, the `display future check` predicate is



```
Results :

[1] CONSENT ISSUE - process sendCookie used marketing_cookie for purpose thirdParties at time 26 without consent
[2] ERASE REQUEST ISSUE - erase of data phone_bob was asked at 35 but not done in time
[3] ACCESS REQUEST ISSUE - subject 'Bob' asked for access at time 37 and was not sent data in time
[4] STORAGE LIMITATION ISSUE - data marketing_cookie was last used at 26 and not deleted in time
[5] STORAGE LIMITATION ISSUE - data phone_bob was last used at 35 and not deleted in time
[6] STORAGE LIMITATION ISSUE - data id_bob was last used at 21 and not deleted in time
```

Figure 4. Compliance results display

triggered to inform the auditor that further verification will be needed later. Not all principles include a deadline check (e.g., lawfulness does not), but rights like access and erasure do.

- `display non-compliant data`: This sub-predicate collects and returns the relevant information when non-compliance is detected, such as the involved data, process, and violated principle or right. It also causes the main predicate to fail (i.e., return `false`), signaling a violation.
- `display no parameters`: This sub-predicate is used when the selected nodes are not subject to the principle being checked. It provides feedback indicating that the rule does not apply in this context.

This modular structure enables precise compliance verification and interpretable output, which helps auditors understand and act on the results more easily.

The full predicate first verifies whether the instantiated parameters are valid. If they are not applicable, it immediately proceeds to the final step, providing feedback through the `display no parameters` predicate.

When the parameters are valid, the predicate returns *true* only if either the `compliance check` or the `deadline check` (which also includes relevant information) succeed. If neither check passes, the predicate triggers `display non-compliant data` to present detailed information about the violation and returns *false*.

The results are sent to the interface for displaying, including information about the personal data, time and processes involved in the possible violation.

## VII. Experimental Validation

To validate our approach, we first tested the correction of the GDPR compliance patterns on small provenance graphs corresponding to the scenarios discussed in Section IV-C. For these initial experiments, we systematically modified key parameters, such as the purposes attached to consent, the timestamps of consent creation, or the timing of actions like data deletion or disclosure, in order to verify that the solver correctly reported all cases of non-compliance. Once the patterns were validated on these controlled examples, we evaluated the performance of the prototype on larger graphs produced by the synthetic graph generator described next.

The results show that the tool maintains acceptable performance for offline audits of graphs of moderate size. Patterns addressing lawfulness, storage limitation, the right to be forgotten, and the right of access are all verified within a few seconds even on the larger generated graphs. However, checks involving the consent principle require noticeably more time: as the graph grows, verifying the complex relationships between personal data, purposes, and consent updates can take significantly longer. Improving the efficiency of these consent-related verifications will therefore be an important focus of future optimization efforts.

### A. Graph generator

In order to generate provenance graphs that reflect realistic scenarios, we developed a generator of provenance graphs. Our approach relies on the assembly of smaller subgraphs, referred to as *bricks*, each representing the execution of a process within the system. These bricks can be either general (e.g., accessing a webpage) or specific to a particular context (e.g., purchasing a product).

Each context is composed of 3 to 5 specific bricks, in addition to a set of 10 general-purpose bricks that are common across contexts. Some bricks are *triggered* bricks, meaning they are conditionally activated by certain actions. To capture this dependency, each standard brick capable of triggering another includes a reference, in its facts, to the corresponding triggered brick (subgraph).

Figure 5 illustrates this mechanism: the general brick sendMail, which defines the subgraph for sending an email, embeds a reference to the triggered brick sendAnalysisCookie, which represents the action of sending an analysis cookie. The variables marked as %VAR% are instantiated during graph generation. The keyword CAN indicates that the triggered brick is optional and will be included based on a random probability between 0.1 and 1.

```
wasControlledBy('%PROCESS:sendMail%','DC', 'owner', %T%, %TC%).
wasGeneratedBy('message','%PROCESS:sendMail%', 'mail to send', %T%).
used('%PROCESS:sendMail%','%DATA:mail%', 'address to send to', %T%).
%CAN:/TRIGGERED/sendAnalysisCookie%
used('%PROCESS:sendMail%', 'message', 'mail sent', %TF%).
```

Figure 5. sendMail brick

To determine the most effective parameters for data storage and retrieval, we considered three distinct usage contexts: a forum or social networking website, an online store, and a public service portal (e.g., a water management platform). For each context, we defined the types of data it may contain (such as email addresses or analytic cookies) as well as the corresponding building blocks (or bricks). Each brick specifies whether it can be reused for the same user. For example, while a user can create an account only once, they may update their information multiple times.

The graph generator takes four parameters, the first of which is mandatory: the name of the output file containing the generated graph encoded as Prolog facts. The other three parameters are optional: the number of users (or a file listing the user names), the desired context, and a repetition *factor*. The factor, i.e., an integer greater than or equal to 1, controls the repetition of reusable bricks, thereby increasing the graph's size. A factor of 1 prevents repetition, while higher values allow bricks to be reused more frequently. If the parameters are not provided, the generator randomly selects values and assigns a default factor of 1.

Bricks are randomly selected, instantiated, and incorporated into the graph. When a triggered brick is added, the generator ensures that all necessary sub-elements for executing the associated action are also included.

Figure 6 illustrates a graph generated with 20 users and a repetition factor of 1, within the public service context. The graph is visualized using the Neo4J tool.

## B. Generated graphs results

First, to estimate verification times, we performed preliminary tests and imposed a maximum duration of four hours per process. For graphs containing up to 200 users, all GDPR principles except *Lawfulness* were verified well within this limit: the principle *Right of access*, for instance, was verified in 4s and the principle *Storage limitation* was verified in 20s. For the *Lawfulness* principle, however, the solver did not consistently complete within the allotted time once graphs reached this scale.

To keep the evaluation consistent and reproducible, we subsequently limited the verification time to five minutes per graph. Under this setting, even graphs of moderate size, such as the example in Figure 6, were fully checked in 18 seconds. Our detailed experiments therefore focus on graphs of up to 50 users, with repetition factors from 1 to 5 across the three defined contexts, which already provide a representative basis for assessing performance.

Our experiments confirmed that the verification time increases with the complexity of the graph, particularly in terms of the number of dependencies. This complexity is influenced both by the repetition factor and the nature of the context-specific bricks, which often introduce additional dependencies. As a result, graphs with larger file sizes tend to require more time for analysis. For example, graphs between 85 and 90 kilobytes (KB) in size were verified in an average of 17 seconds. This time increased to 41 seconds for graphs between 100 and 110 KB, and up to 1.5 minutes for those ranging from 110 to 125 KB.

However, file size alone does not fully determine verification time (see Figure 8). We observed significant variability in execution times for graphs of similar size. In particular, contexts with a high volume of personal data tend to incur longer verification times. This is largely due to the repeated evaluation of the `isPersonal` predicate, which plays a key role in determining whether specific compliance rules are applicable. The more frequently this predicate needs to be resolved, the longer the verification process takes. Indeed, by checking each principle individually, we observe that the time required to verify the principle of *Lawfulness* (and therefore the `validConsent` predicate that uses `isPersonal`) is 100 times higher than for the other principles (see Figure 7). For the graph in Figure 6, the verification time for the *Lawfulness* principle is 17s, whereas the verification of the *Right of access* principle takes only 3ms.

While our experiments did not cover very large provenance graphs, the framework supports a modular verification strategy. The interface allows selecting specific principles, processes, data types, or individual users, enabling the analysis of per-user (or per-principle) graphs. Such a decomposition is, in principle, more scalable and is naturally supported by the tool's interface. All the experiments were performed on a MacBook Pro equipped with an Apple M2 chip and 16 GB of RAM.

## VIII. Conclusion

In this paper, we propose a modeling of the core principles and rights of the GDPR, based on the provenance model. Our representation captures these principles and rights by combining conjunctions and disjunctions of causal dependencies from the OPM, temporal constraints on timestamps, and conditions on node attributes.

These compliance patterns are flexible and can be extended to cover additional GDPR articles or other data protection regulations. For instance, the Health Insurance Portability and Accountability Act (HIPAA) imposes retention periods for medical records, a requirement that can be expressed similarly

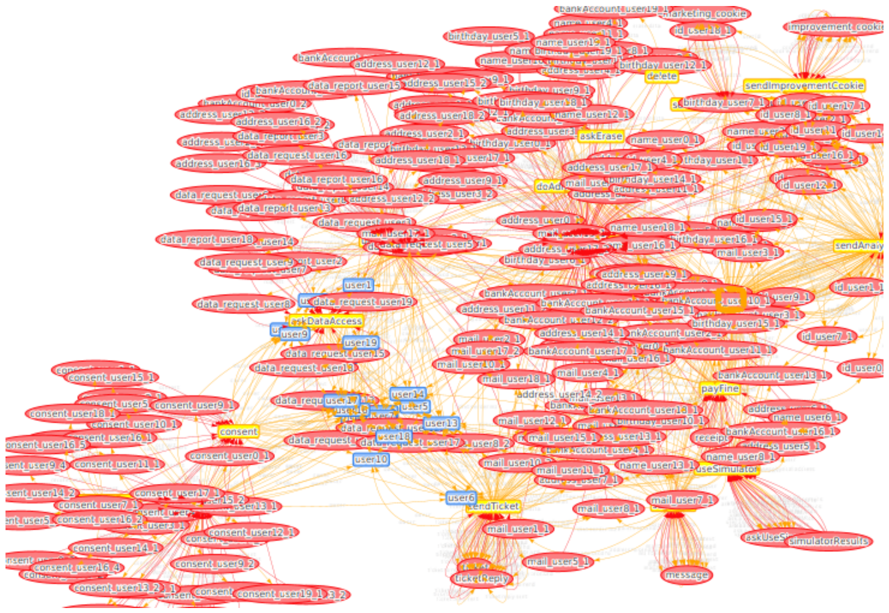


Figure 6. Neo4j representation of a provenance graph extract generated by the generator with 20 users and factor 1

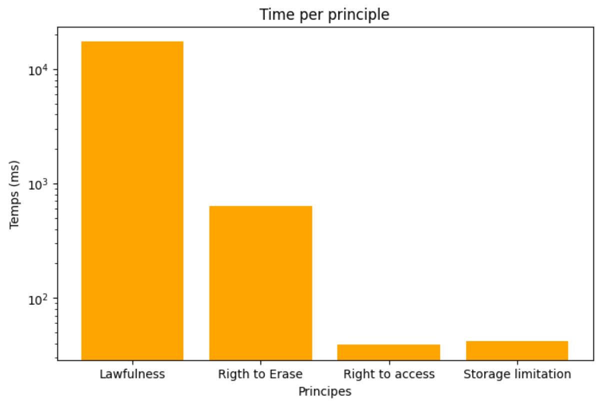


Figure 7. Time per principle for a graph of 20 users

to the right to be forgotten using path expressions with temporal constraints. Likewise, the structural similarity between GDPR and GDPR-UK articles allows our patterns to be easily adapted to ensure compliance with the latter regulation.

We validate our approach through a prototype tool that implements compliance patterns based on causal dependency paths, allowing partial automation of GDPR compliance verification using a Prolog solver. To demonstrate the feasibility of our approach, we conducted experiments on both small- and large-scale scenarios using a graph generator. This generator produces provenance graphs that can incorporate or exclude non-conformities, reflecting actions specific to each type of system. Increasing the variety of building blocks in the graph generator would allow more actions to be represented, thereby expanding the applicability of our approach.

Future work will focus on extending this approach to leverage real system logs or system traces for provenance graph generation, enabling the analysis of real-world scenarios beyond synthetic ones.

## References

[1] P. Di Salvo-Cilia, A. Martinez Anton, and C. Bertolissi, “Towards automated checking of gdpr compliance”, in *Proceedings of the CYBER 2024 Conference*, Extended abstract., IARIA, 2024.

[2] M. Miri, F. H Foomany, and N. Mohammed, “Isaca: Complying with gdpr, an agile case study”, *ISACA Journal*, vol. 2, Apr. 2018.

[3] J. Mohan, M. Wasserman, and V. Chidambaram, “Analyzing GDPR Compliance Through the Lens of Privacy Policy”, *arXiv e-prints*, arXiv:1906.12038, Jun. 2019, Art. no. arXiv:1906.12038. DOI: 10.48550/arXiv.1906.12038. arXiv: 1906.12038.

[4] A. Xiang, W. Pei, and C. Yue, “Policychecker: Analyzing the gdpr completeness of mobile apps’ privacy policies”, in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS ’23, Copenhagen, Denmark: Association for Computing Machinery, 2023, pp. 3373–3387, ISBN: 9798400700507. DOI: 10.1145/3576915.3623067.

[5] S. Chowdhury, L. Coles-Kemp, and G. Kambourakis, “Toward automated gdpr compliance checking”, *Computers & Security*, vol. 100, p. 102 089, 2021. DOI: 10.1016/j.cose.2020.102089.

[6] E. Arfelt, D. Basin, and S. Debois, “Monitoring the gdpr”, in *Computer Security – ESORICS 2019*, Cham: Springer International Publishing, 2019, pp. 681–699, ISBN: 978-3-030-29959-0.

[7] M. Hashem Eiza, V. Thong Ta, Q. Shi, and Y. Cao, “Secure semi-automated gdpr compliance service with restrictive fine-grained access control”, *Security and Privacy*, vol. 7, no. 6, Aug. 2024. DOI: 10.1002/spy2.451.

[8] B. Esteves and V. Rodríguez-Doncel, “Analysis of ontologies and policy languages to represent information flows in gdpr”, *Semantic Web*, vol. 15, no. 3, pp. 709–743, May 2024. DOI: 10.3233/SW-223009.

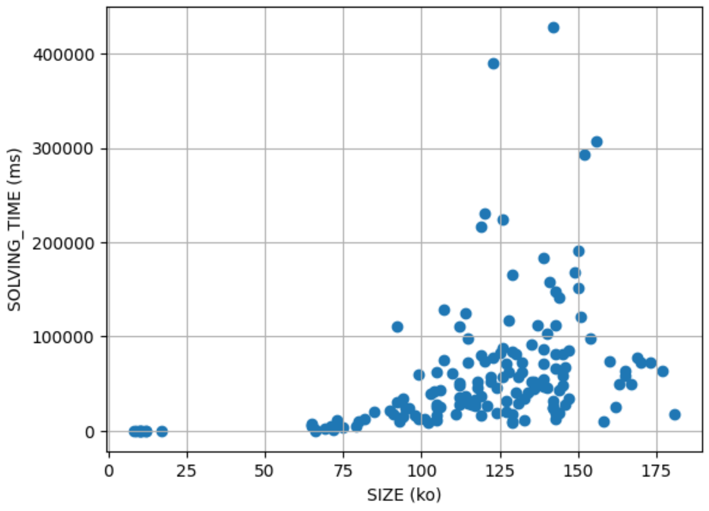


Figure 8. Dependency between graph size and resolution time.

[9] T. Athan, G. Governatori, M. Palmirani, A. Paschke, and A. Wyner, “Legalruleml: Design principles and foundations”, in *Reasoning Web. Web Logic Rules 2015*, ser. Lecture Notes in Computer Science, vol. 9203, Heidelberg: Springer, 2015, pp. 151–188.

[10] C. Bartolini, G. Lenzini, and C. Santos, “A legal validation of a formal representation of articles of the gdpr”, in *TERECOM@JURIX*, 2018.

[11] A. Tauqeer, A. Kurteva, T. R. Chhetri, A. Ahmeti, and A. Fensel, “Automated gdpr contract compliance verification using knowledge graphs”, *Information*, vol. 13, no. 10, 2022, ISSN: 2078-2489. DOI: 10.3390/info13100447.

[12] T. R. Chhetri et al., “Data protection by design tool for automated gdpr compliance verification based on semantically modeled informed consent”, *Sensors*, vol. 22, no. 7, 2022, ISSN: 1424-8220. DOI: 10.3390/s22072763.

[13] S. Agostinelli, F. M. Maggi, A. Marrella, and F. Sapio, “Achieving gdpr compliance of bpmn process models”, in *Information Systems Engineering in Responsible Information Systems*, C. Cappiello and M. Ruiz, Eds., Cham: Springer International Publishing, 2019, pp. 10–22, ISBN: 978-3-030-21297-1.

[14] M. Barati, O. Rana, I. Petri, and G. Theodorakopoulos, “Gdpr compliance verification in internet of things”, *IEEE Access*, vol. 8, pp. 119 697–119 709, 2020. DOI: 10.1109/ACCESS.2020.3005509.

[15] D. Basin, S. Debois, and T. Hildebrandt, “On purpose and by necessity: Compliance under the gdpr”, in *Financial Cryptography and Data Security*, S. Meiklejohn and K. Sako, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, pp. 20–37, ISBN: 978-3-662-58387-6.

[16] M. Saltarella, G. Desolda, and R. Lanzilotti, “Privacy design strategies and the gdpr: A systematic literature review”, in *HCI for Cybersecurity, Privacy and Trust: Third International Conference, HCI-CPT 2021, Held as Part of the 23rd HCI International Conference, HCII 2021, Virtual Event, July 24–29, 2021, Proceedings*, Springer, 2021, pp. 241–257.

[17] M. Rhahla, S. Allegue, and T. Abdellatif, “Guidelines for gdpr compliance in big data systems”, *Journal of Information Security and Applications*, vol. 61, p. 102 896, 2021, ISSN: 2214-2126. DOI: https://doi.org/10.1016/j.jisa.2021.102896.

[18] M. Palmirani and G. Governatori, “Modelling legal knowledge for gdpr compliance checking”, in *International Conference on Legal Knowledge and Information Systems*, 2018.

[19] *Cloud for europe*, https://cordis.europa.eu/project/id/610650, Accessed: 2025-11-25.

[20] R. E. Hamdani et al., “A combined rule-based and machine learning approach for automated gdpr compliance checking”, in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*, ser. ICAIL ’21, São Paulo, Brazil: Association for Computing Machinery, 2021, pp. 40–49, ISBN: 9781450385268. DOI: 10.1145/3462757.3466081.

[21] S. Zimmeck, P. Wang, R. Kang, J. R. Reidenberg, and N. M. Sadeh, “Automated analysis of privacy requirements from privacy policies”, in *2019 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2019, pp. 441–458. DOI: 10.1109/SP.2019.00038.

[22] The European Parliament and the Council, “Directive 1995/46/ec of the european parliament and of the council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data”, *Official Journal of the European Communities*, Oct. 1995.

[23] L. Moreau et al., “The Open Provenance Model core specification (v1.1)”, *Future Generation Computer Systems*, vol. 27, no. 6, pp. 743–756, 2011.

[24] C. Bertolissi, J. Hartog, and N. Zannone, “Using provenance for secure data fusion in cooperative systems”, in *Symposium on Access Control Models and Technologies*, ACM, 2019, pp. 185–194.

[25] Prologia, Ed., *Prolog III : Manuel de référence*. Marseille: Prologia, 1996.

# Learning Together in Global Classrooms: Student Engagement Through Collaborative Activities and Games

Simona Vasilache
Institute of Systems and Information Engineering
University of Tsukuba
Tsukuba, Japan
e-mail: simona@cs.tsukuba.ac.jp

*Abstract* - **Software engineering education increasingly takes place in classrooms where students from diverse cultural backgrounds learn side by side. While this diversity enriches the learning environment, it also introduces challenges in communication, collaboration, and engagement. This paper reports on experiences from teaching a software engineering course with a highly international student cohort, focusing on the role of classroom activities as cross-cultural bridges. Active learning strategies, such as collaborative problem-solving exercises and in-class games, were employed to encourage participation, foster mutual understanding, and develop teamwork skills essential to professional practice. Drawing on classroom observations and student feedback, this work highlights how such activities can mitigate cultural barriers and create opportunities for students to appreciate different perspectives. Findings suggest that collaborative exercises not only improved engagement but also facilitated learning and strengthened students' sense of belonging in a multicultural environment. The paper argues that designing engaging classroom activities, where students from different cultures work together towards achieving a common goal, is crucial for preparing students to work in global software engineering contexts, where cross-cultural collaboration is essential.**

***Keywords - software engineering; student engagement; active and collaborative learning; multicultural environments.***

## I. INTRODUCTION

We live in a world which relies heavily on software systems. Although a relatively young discipline (57 years since it was framed as an engineering discipline in its own right, i.e., in 1968), software engineering provides systematic methods for designing, developing and maintaining software systems that underpin our modern society [1]. In higher education, software engineering related subjects are well established and offered by an increasing number of institutions. According to CS2023 [2], a computer science curricular guidelines document (published by the Joint Task Force of Computer Science Curricula, which comprises ACM, IEEE-CS, and the Association for the Advancement of Artificial Intelligence), "since 2013, the focus of curricular design has moved from what is taught (a knowledge model) to what is learned (a competency model)" [2]. Thus, educators teaching software engineering subjects must equip students not only with technical concepts, the so-called "hard skills", but also with a range of interpersonal and professional "soft skills".

In 2024, a systematic literature review of skills development revealed 33 essential soft skills that educators must teach future software engineers [3]. The top 5 soft skills revealed were communication, teamwork, organization, leadership and learning. They were followed by creativity, critical thinking, analytical skills, problem solving and professionalism. To help with acquiring important abilities like teamwork, communication, and analytical skills, collaborative learning is a very useful tool in a software engineering classroom. This strategy makes sure that students "collaborate" in order to achieve a task and this fosters development of various soft skills. Through collaborative learning, educators can simulate real-world professional settings, where software engineers must work together to develop software applications.

In their learning experience, students' performance is shaped by a wide range of factors, with motivation standing out as one of the most important. In the context of engineering disciplines, motivation has been described as "particularly critical" [4], given the complexity of the skills and knowledge students are expected to acquire. In response, educators have increasingly turned their attention to innovative pedagogical approaches that can increase and sustain motivation. Among these, game-based learning has gained significant momentum in recent years due to its demonstrated potential to enhance engagement, encourage active participation, and ultimately improve learning outcomes [5].

While studying, learners are often part of international classrooms. As a matter of fact, multicultural environments are now a common feature of workplaces, academia, and everyday life. Teaching in such settings presents not only specific challenges [6] but also distinct advantages [7]. A multicultural classroom can serve as a microcosm of distributed development teams, achieving the goal of providing global software engineering education. While students learn together in the same physical space, they simultaneously gain insight into how they may need to collaborate in the future with colleagues from different cultural backgrounds, each bringing diverse values, behaviors, and working styles [7].

In our previous works ([1], [8]), we highlighted some of the challenges of teaching an introductory software engineering course to a multicultural group of graduate students in a Japanese university. This paper extends our work with further exploration of how student engagement was achieved through collaborative activities and game-based learning, along with lessons learned during this course.

The remainder of this paper is organized as follows. Section II describes the background of our work, as well as related work. A description of our course is provided in Section III, followed, in Section IV, by an illustration of collaborative learning and game-based learning as they were employed in our course. Section V includes a discussion and lessons learned during our study. Finally, section VI provides conclusions and directions for future work.

## II. Background and related work

This section examines considerations on collaborative learning, game-based learning and global software engineering, along with related work outlining recent development in these fields.

### A. *Collaborative Learning*

Collaborative learning is an important and useful tool promoting engagement of learners. Despite a generally accepted lack of consensus on the rigorous definition of the term [9], it has certain widely recognized characteristics. As an educational approach to teaching and learning, it involves learners "collaborating" (i.e., working together) to achieve a common goal – solve a problem, complete a task or create a product [10], at the same time progressing individually during the learning process. Participants are challenged socially and emotionally while listening to different perspectives and they may often be required to defend their ideas [9]. Fundamentally, similarly to active learning, collaborative learning promotes active student participation [11].

In international settings such as multicultural classrooms, collaborative learning is particularly valuable for fostering awareness of cultural differences, exposing students to diverse perspectives, and strengthening mutual understanding, as they work together toward solutions acceptable to the entire group. It can also support the development of communication and social skills, by offering a safe and structured environment to interact with others [11].

Furthermore, collaborative learning helps prepare students for their future workplaces: it improves communication, negotiation and teamwork skills – all very important in software engineering, where projects are often team-based. This makes collaborative learning an essential tools for educators in software engineering courses, particularly those that include intensive team projects.

### B. *Game-Based Learning*

In game-based learning (GBL), the focus is on acquiring knowledge and skills through gameplay. Whether digital or non-digital, specific games are designed to achieve certain educational goals. This strategy has drawn increased attention, in various disciplines, including software engineering. In their work, Garcia et al. [12] conducted a systematic literature review of the effects of GBL in acquisition of soft skills in undergraduate software engineering courses. Their review shows that researchers have recognized the effectiveness of using GBL in teaching and learning various software engineering topics. This is largely due to the games' intrinsic features, which make them attractive to students and improve their motivation [12]. Moreover, numerous researchers recognize that games contribute to the development of "soft skills", like teamwork and communication ([13], [14], [15]). Furthermore, Marti-Parreno et al. [16] argue that GBL can enhance the delivery of learning content by allowing students greater control over their own learning process during gameplay. By adopting a GBL approach, students are able to apply classroom concepts in practice, thereby reinforcing and deepening their understanding of those concepts [16].

GBL is used not only in academic environments, with students, but also in industry, for training employees. To give an example, the work of O'Farrell et al. [17] shows how it was used for training employees on the Scale Agile Framework, [18], through a 3D game named PlaySAFe. This study showcased various benefits of GBL, like "allowing newcomers a quick and efficient means to learn and understand the practical groundwork of SAFe in advance of learning more theoretical concepts in conventional training" [17].

Overall, GBL offers an effective pedagogical approach in software engineering by enhancing motivation and enabling students to apply theoretical concepts in practical, interactive contexts.

### C. *Global Software Engineering*

Globally distributed software projects are widespread in today's world and special skills in communication across different locations and time zones need to be learned early by software engineering students. An increasing number of universities are incorporating global software engineering into their curricula, yet its adoption remains limited. In their work, Beecham et al. [19] emphasize the need for global software engineering education and summarize several global software engineering education related challenges and proposed solutions. According to Schmiedmayer et al., [20] there are two main options to make students aware of global software engineering challenges and equip them with the necessary skills to deal with them: teachers can simulate a global software project in a classroom setting, or they can arrange a genuine global software engineering project; the latter option comes with all the organizational challenges of a distributed organization and infrastructure [20].

In terms of learning and applying soft skills, a multicultural classroom offers a suitable environment to simulate some of the challenges of a globally distributed project, as exposed by some educators' work. For instance, the work in [21] shows how the online environment brought by the Covid-19 pandemic provided an opportunity to test mini-models of distributed teams in software engineering, in the context of a multicultural classroom. In such a setting, while working together on achieving various tasks in the classroom, students learn how to collaborate with colleagues from different countries, each with their own culture, language and specific expectations.

## III. Course description

This section describes the setting, composition and content of the course that constitutes the subject of this paper.

### A. Course Setting and Class Composition

This paper is based on the experience of teaching an introductory software engineering course named "Principles of Software Engineering", as it was offered in its latest edition, in the spring semester of 2024. This course has been taught annually as an elective course in the Master's Program in Computer Science at the University of Tsukuba in Japan, between 2016 and 2022. Since 2023, it has been held once every two years, alternating with a computer ethics course; it takes place in even number years, thus no class was held in 2023. The length of the course is 10 weeks, with 3 hours held every week; if completed successfully, students obtain two academic credits for it. The grading is based on the submission of a final report, which the students have approximately 3 weeks to complete.

The course aims to familiarize students with the fundamental principles of software engineering and to highlight its importance as a modern engineering discipline. Core topics include software development models and life cycles, agile methodologies, requirements engineering, user interface design, verification and validation techniques, project planning and management, software engineering tools such as IDEs and UML, as well as the business aspects of software development. The 2024 edition of the course saw 105 participants: 35 students in their first year and 70 students in their second year of master's course. They were a mixture of Japanese students (43) and international students (49 regular students and 13 exchange students, coming from a total of 15 different countries). Most students belong to the computer science department; only 4 students belong to different departments.

It is worth mentioning that this course started with 15 participants in 2016 and reached 105 enrolled participants in 2024 (it generally grew every year, apart from a steep decline in 2020, during the beginning of Covid-19 pandemic). Table I shows the total number of students, along with the number and percentage of international students enrolled in each of the 8 editions of the course. As can be observed, international students usually represent between 50% and 70% of the total number of students. Notably, this course is being held in English (and the instructor is non-Japanese, as well). This feature is responsible for attracting comparatively many international students, who have fewer course choices of courses held in English and for whom, often, taking classes held in Japanese is challenging. All communications, teaching, class materials, plenary discussions are held using English. However, students are allowed to submit their assignments (or any other feedback on the learning-management system) in Japanese. Moreover, group discussions are allowed in any language, as long as it is understood by all the participating members. Importantly, using a language other than English is only allowed in class verbally, but never in writing (since shared documents are often seen by all students, who need to understand them). Besides English, Japanese was the most spoken language in class; there were several instances in which certain small groups spoke Chinese or French while performing their class tasks. Last, but not least, reports or assignments, for which the access is restricted to the submitting student and the instructor, could be submitted either in English or in Japanese.

TABLE I. INTERNATIONAL STUDENT ENROLLMENT IN SOFTWARE ENGINEERING COURSE

| | *Total number of students* | *Number of international students* | *Percentage of international students* |
|---|---|---|---|
| 2016 | 15 | 9 | 60% |
| 2017 | 26 | 18 | 69.2% |
| 2018 | 35 | 24 | 68.5% |
| 2019 | 66 | 33 | 50% |
| 2020 | 34 | 28 | 82.3% |
| 2021 | 53 | 37 | 69.8% |
| 2022 | 66 | 33 | 50% |
| 2024 | 105 | 62 | 59% |

### B. Course Content and Class Flow

The course combined a variety of teaching methods, including discussions, brainstorming activities, games, micro-projects, and instructor-led lectures. More specifically, classes usually started with a 5-minute warm-up activity (which could be a short discussion on a recent science or technology piece of news). This was followed by a repeated combination of lecture (in which the instructor explained a new topic), discussions among students (either in groups or with the whole class), and various activities and/or games. Figure 1 illustrates how lecture, class activities and games are interconnected during each class. The lecture part is always followed by discussions, which are closely connected to class activities and often games, as well. The cool-down part is mostly made-up of summarizing/concluding discussions. Notably, discussions were part of the class since its inception (in 2016); gradually, activities and games were introduced in subsequent years, with the variety and number of items increasing every year.

Throughout the course, the instructor used every opportunity to gather feedback from the students, either through informal discussions (during the break time or after class) or by means of written feedback, submitted trough the learning management system (LMS) used in the course, i.e., *manaba* [22]. Following some of the longer activities, students were invited to provide extensive feedback on aspects such as strengths and weaknesses, language or group preferences, and general impressions.

During the last class, the link to a comprehensive survey was distributed, to which 30 students responded, out of a total enrollment of 105. The instructor attributes this relatively low response rate to timing, as the survey was administered at the end of the course, when many students felt they had already provided sufficient feedback during earlier sessions. Finally, only 31 students responded to the end-of-course evaluation questionnaire provided by our university.

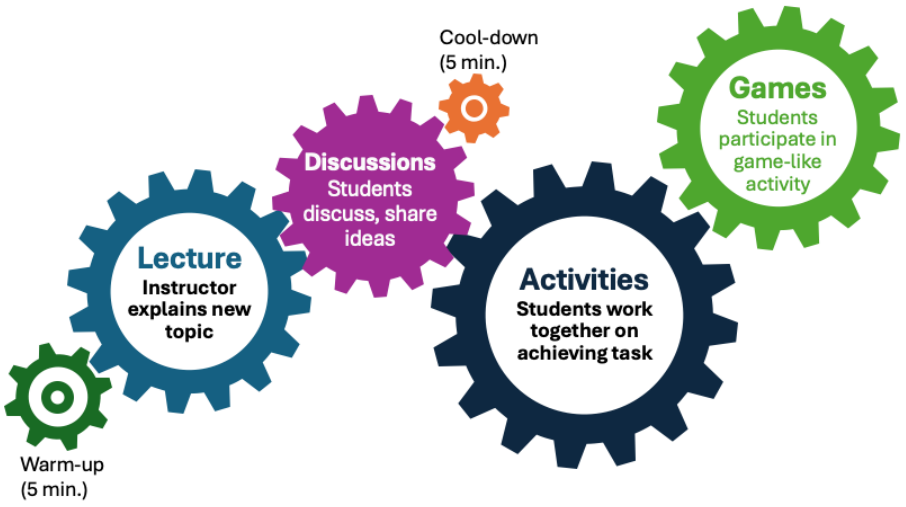

Cool-down
(5 min.)
Games
Students participate in game-like activity
Discussions
Students discuss, share ideas
Lecture
Instructor explains new topic
Activities
Students work together on achieving task
Warm-up
(5 min.)

Figure 1. Visual representation of inter-connected class components of introductory software engineering course

## IV. Implementation of Collaborative Learning and Game-Based Learning

This section illustrates two class activities: the first one shows the implementation of collaborative learning, whereas the second one provides an example of deploying game-based learning.

### A. *Class Project: Requirements Elicitation and Requirements Specification*

The ability to work and collaborate/communicate successfully in a team is a crucial skill for a future software engineer. One of the most effective ways to develop teamwork and communication skills is to include a project in the coursework where students must work in teams. This work in groups is "essential in active approaches that are based on real-world problem-solving practices" [23].

To achieve this purpose, one of the classes during this course was dedicated to a project in which teams of students were responsible with creating a (simple) requirements document for a given application. This activity covered requirements elicitation and requirements specifications for a given software application.

The "Project" function of the LMS was used to facilitate work in teams, as well as sharing the teams' work with the whole class. The activity was divided into 3 main parts, with 6 tasks in total. In the first part, requirements elicitation took place; in the second part, the requirements document was created; in the last part, feedback on this document was collected among team members. Finally, the created requirements documents and everything included in the "Project" was shared with the whole class (all the students had access to all the documents through the LMS).

Before the class began, the instructor manually divided the students into 10 teams (namely *Team A* to *Team J*), each with 10 or 11 members. Three different types of teams were created: teams with Japanese students only, teams with international students only, and teams with a mixture of Japanese and international students.

Five applications were suggested to be discussed, each of them being covered by one, two or three teams, as follows.

*Team A, Team B: language learning application*
*Team C, team D: medical records management system*
*Team E, Team F: low-budget oriented online shopping system*
*Team G, Team H, Team J: time/task management application*
*Team I: dating application for retired people*

*Purpose:* create a requirements document for a given application.

The project consisted of 6 tasks, as described in Figure 2. At the start of the activity, each team chose two members who would act as stakeholders (*users: U*); the remaining members of the team would act as requirements analysts (*developers: D*). Thus, each team included two stakeholders and a maximum of 9 developers (depending on the number of members in the team). In theory, each team was made up of either 10 or 11 members; in practice, not all students were present on the day, thus some teams had fewer members.

The instructor believed that the inclusion of stakeholders in the group highlighted their crucial role when developing an application in the real world. According to the Guide to the Software Engineering Body of Knowledge [24], real-world software projects often suffer from two primary requirements-related problems: incompleteness (when stakeholder requirements are not revealed and properly communicated) and ambiguity (when requirements are communicated in a way that is open to multiple interpretations). Students need to learn that the presence of stakeholders is essential in the requirements elicitation phase of developing a software system.

The resulting documents for each of the steps/tasks were saved in the LMS, using its "Project" function. Some teams chose to additionally create a shared google document and place the link in the project location.

<u>Task I</u>: Ds write a questionnaire for the two stakeholders NB: Ds and Us do not communicate directly)

<u>Task II</u>: Us answer the questionnaire

<u>Task III</u>: Ds discuss with Ds from the other team(s) working on the same application
- Typical questions discussed include: *"What kind of questions did you include?"*, *"How can we find out what our application should do?"* etc.

<u>Task IV</u>: Ds must think of the following question. *Which is more efficient: a pre-defined questionnaire (you had time to think about it) OR interview (you can ask "live" questions?)*

<u>Task V</u>: Ds create a requirement document in free format (suggestions: use "shall", "should", possibly, also, "unclear" statements)

<u>Task VI</u>: Us give brief feedback on the requirements document - what they disagree with, what "pleasantly" surprises them, what is missing, <u>based on their answers to the questionnaire</u> prepared by the respective Ds

Figure 2. Description of tasks to be achieved during the class project

The students were seated in a large capacity classroom, which is organized in three sections of 3-person desks. Unfortunately, the desks and chairs cannot be moved or turned around, sometimes making it difficult for students to hold discussions in groups (they would have to turn their body to be able to speak to the colleagues seated behind). Figure 3 illustrates the seating of the students and a common manner of occupying the desks.

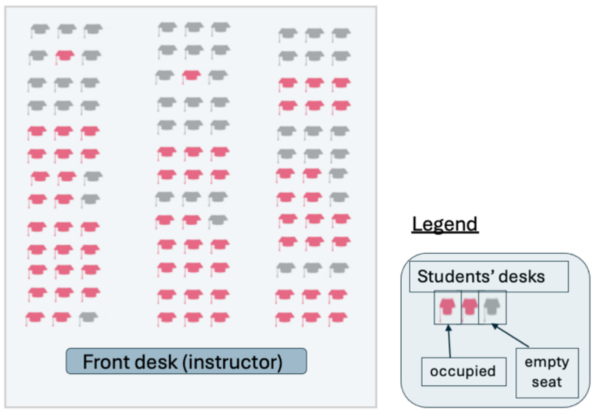


Figure 3. Visualization of clasroom seating

After the class ended, the students had to respond to a series of questions posed by the instructor, in the form of an assignment (submitted through the LMS). Importantly, the students had time until the end of the day to respond – thus allowing several hours to reflect upon the day's experience.

It should be noted that the instructor did not indicate whether the assignment was mandatory or linked to the final course grade. Her intention was to encourage students to share and reflect on their experiences only if they felt they had meaningful insights to contribute. In her view, making the task compulsory might have led some students to submit responses merely for the sake of compliance, rather than genuine engagement.

Feedback was received from 62 students (out of the ~70 students present on the day of the activity.) The requirements for the assignment are included below; the description also invited students to provide additional comments or suggestions, if they wished to do so.

*Regarding today's activity, please let me know your thoughts (anything is fine - I am truly interested in your opinions!).*

*<u>Part I. As a software developer</u>*
*- Was there a leader among developers?*
*- Did everyone participate?*
*- Which do you think is "harder": to be the U ("stakeholder") or the D ("developer")?*
*- Is it better to have "smaller" or "larger" development teams?*

*<u>Part II. As a class participant</u>*
*- Did you enjoy the activity? (Please be honest, it is important for possible future activities!)*
*- Was it difficult to communicate with your colleagues?*
*- What was "the best" part today? What was "the worst" part today?*
*- Would you prefer your group members to speak the same language as you? (Is English ok for everyone?)*

The comments submitted by the students highlighted various important and equally interesting issues. In the following, we shall focus on the answers given by students from <u>a class participant</u> point of view, i.e., Part II of the assignment.

The first question asked whether students enjoyed the activity; the majority of students declared that they did. We acknowledge that, even with the instructor's insistence on honesty, it is possible that social desirability bias was present - some students may have simply wanted to please the instructor. (Social desirability response bias is defined as a participant's tendency to over (under) report activities that are socially desirable (undesirable) [25].) Notably, since the answers were provided as assignments, they were not anonymous (they were added to each student's assignment portfolio) – an additional factor that may have influenced the students' responses.

Positive and encouraging responses included:
- *"It was fun and very different from what the other classes are doing."*
- *"YES, ideas shared by others are interesting especially if they are unexpected ones."*
- *"I enjoyed the activity. Compared to regular classes, I think this activity is a very interesting process and allows for a better grasp of the knowledge learned."*
- *"I enjoyed the activity. There were many diverse opinions, and they were stimulating."*
- *"Yes, can't wait for the next one."*
- *"I enjoyed it very much, especially interacting with others."*
- *"This activity provided me the chance to talk with other students and make new friends!"*
- *"It was great and I was able to talk to new people."*
- *"I enjoyed this activity. I am not good at speaking English, so I felt awkward to speak English and helplessness not to tell what I think well. However, I didn't felt such things this time. It is easy to communicate with my team. I was able to tell my opinion actively."*
- *"It was my first experience and I had a lot of fun." (translated from Japanese)*
- *"The class was very interesting because I never considered myself to be on the user side before."*

One notable answer was provided by a student who did not seem to enjoy the activity: "*My English is not very good. I can understand everyone's thoughts, but I have almost no practice in expressing myself. The better thing is that I can share my ideas by sharing documents. To be honest, it's not an enjoyment."*. The same student noted that it was difficult to communicate with their colleagues and that they would prefer for group members to speak the same language.

One other participant expressed their lack of satisfaction with the activity in the following manner: *"Everything felt arbitrary, and far from actual developing/consulting processes."* Unfortunately, they did not provide further clarifications, which would have helped the instructor understand the issue.

With regard to the preferred language of communication, most students seem to accept English as a common language. However, 6 students expressed their desire to use Japanese (it is worth remembering that 43 class participants are Japanese, although not all of them were present on the day of this activity). One student expressly stated that they would like to use English, even though their mother tongue is Japanese: *"I would prefer if we could speak in English, not put in a group where everyone speaks Japanese (I speak Japanese, but since it's an English class I'd prefer it that way)"*. Another participant stated: *"Yes, I prefer to speak the same language. (Because Japanese student may not be able to follow the speed of discussion in English.)"*.

When asked directly whether it was difficult to communicate with colleagues during the activity, only one student (out of the 62 who answered) specifically stated that it was (his answer was in Japanese). Although all the other participants considered that communication was not difficult, several comments highlighted the benefits of using one's own language for ease of communication and expressing own ideas; some examples are provided below.
- *"English is fine for everyone, but I think communicating in my native language would allow me to express my ideas more fluently."*
- *"[...] we have communicated in Japanese. Discussing in English should be harder than in Japanese."*
- *"I think communication is smoother and more efficient when group members speak the same language. English is fine, but Japanese is easier to understand."*
- *"I usually hesitate to speak English, but when I could speak the same language, I could be more active."*
- *"It is quite difficult to exchange opinions in English." (translated from Japanese)*
- *"As a Japanese, I think it is easier to have in-depth discussions about the content of the class if I am in a group with Japanese. (Although it does not seem to improve my English skills.)"*
- *"We want to speak the same language because it is easier to communicate."*
- *"I think communication would be smoother if we spoke the same language, but I think I'll improve more in English." (translated from Japanese)*

A particularly interesting comment was provided by one Japanese student: *"In the aspect of smoothly communication, it's better to have same language members. Inversely, in the aspect of diversity of opinions, it's better to have members with different languages."*. The student appears to be aware of the difficulties of speaking different languages (and, in the instructor's opinion, difficulties with Japanese vs. English), but also highly aware of the importance of diversity of opinions – different languages imply different cultures and thus "diverse opinions".

Various other students' opinions were elicited through this short assignment. One of these questions asked the students what the "best" and the "worst" part of the class project were. Two answers stood out among the students' responses.
- *"There were many diverse opinions, and they were stimulating. The best part of today's class was "when the opinions came together," and the worst part was "when no opinions were expressed.""*
- *"The best part is everyone working together collaboratively. The worst part is to write the report."*

As expected, cultural differences played an important part in the development of the class. For some students, as can be observed from the following comment, working with colleagues from the same cultural background is important: *"Grouping people of the same background together is the best part"*. This observation aligns with the instructor's own empirical observations – she can often observe participants (Japanese students in particular), who, when offered a choice, decide to work in a mono cultural team (i.e., they make sure to be part of a Japanese only team). As shown by Rodriguez-Perez et al. [26], some of the problems that arise in diverse working teams "can be explained by the Similarity-Attraction theory and the social categorization perspective". According to the Similarity-Attraction theory [26], individuals working in groups tend to prefer working with those who resemble them, while the social categorization perspective suggests that group members are more likely to trust, like and cooperate with similar others [27]. Classroom observations made by the instructor appeared to confirm the relevance of both perspectives.

In their responses, students mentioned other "best parts" of the activity, like leadership skills, working with the team to develop the questionnaire, collaborating with classmates and solving problems together, "communicating among us developers", "when everyone accepts each other's opinions". As for "worst" parts, one opinion stood out in particular: "*The "worst" is when I know we have to work as a team (shy)*". This statement suggests that for some students, especially those who describe themselves as shy, the requirement to work in teams can be perceived as stressful or uncomfortable.

Through this "Project" activity, as performed in class, with the use of the LMS, students were faced with issues similar to those that arise during the requirements elicitation phase in the development of a real-life software product. Moreover, they could observe firsthand how cultural differences impact working in a team, as well as how they affect the overall process of software development.

### B. *Game-Based Learning: Agile Paper Airplane Game*

One classical example of a classroom activity designed to simulate agile software development practices is the "Agile Paper Airplane Game [28]. This activity is particularly suitable for students who learn about agile methodologies (and Scrum [29]) for the first time, teaching them "the benefits of working in sprints, planning, retrospectives and teamwork" [28]. Students are divided into small teams and tasked with designing and building paper airplanes under iterative conditions. Instead of planning everything in advance, teams work in short "sprints," receiving feedback after each round on design improvements. Some important concepts are highlighted and experienced through this activity: continuous improvement, lean workflow and "Definition of Done" (defined as all the characteristics and standards a product increment needs to meet in order to be released [29]).

In our classroom, the students were divided into teams of 6-8 members. The teams were created based on the student seating. Before the class started, they had the opportunity to sit down in areas designated "Japanese language", thus belonging to groups in which everyone spoke Japanese. Only two groups of students were created in this manner. Incidentally, one team turned out to be made up of Chinese students only, thus making Chinese the language spoken in this group.

The activity started with the instructor explaining that the "goal" is to deliver paper airplanes that "meet customer expectations", i.e., fly a certain distance (the same one for all the teams). A leader was designated before any work started. Similarly to agile work, the teams worked in sprints – in our case, 3 sprints. In each iteration, the first minute was dedicated to planning; next, within three minutes, the teams had to fold and test as many paper airplanes as they could. After the three minutes of "development", they had one minute for retrospective – to discuss what could be improved towards the next iteration. One important rule stated that each team member was allowed to perform only one fold at a time, after which they had to pass the airplane under construction to the next team member. The third sprint came with a modified version of Definition of Done: the paper airplane's nose had to be blunt, and the required flying distance was increased.

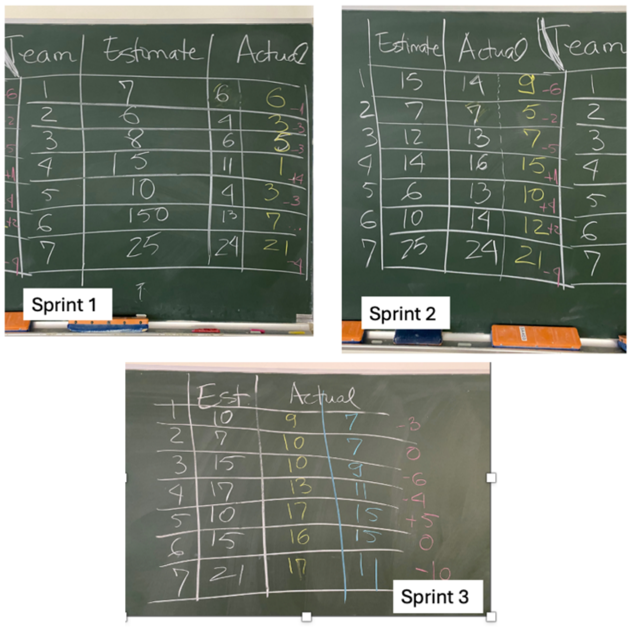
Figure 4 shows three snapshots of the class taken during this activity, showing students folding airplanes and discussing during "retrospective" (included with the permission of the students).

Figure 4. Students participating in the agile paper airplane game

Notably, before each sprint, the teams were asked to estimate how many airplanes they would be able to build. "Definition of Done" plays an important role here: the airplanes must fly the designated distance, otherwise they cannot be counted as successful. After each sprint, the teams had the opportunity to refine their design – they made new decisions, under the guidance of the leader, during the one-minute retrospective. Each sprint focused on continuous improvement and responding to requirements. Teamwork and communication played a crucial role in this activity – aspects clearly noticed by the students, as could be seen from the comments they submitted after class.

When it comes to estimating how many "correct" airplanes their team would be able to build, the 3 sprints proved the concept of continuous improvement: as they advanced to the next sprint, the teams managed not only to estimate better the number of airplanes, but also to build more such planes which fulfill the Definition of Done characteristics. Figure 5 shows examples of the teams' estimations after each of the 3 sprints, along with the actual numbers of planes created, including "correct" ones. (The figure shows screenshots of the blackboard, as they were taken in class by the instructor.)

As can be observed, in the first sprint, the teams had no idea how difficult/easy it would be to build as many planes as possible within the 3-minute timeframe. Team 6, for instance, estimated that they would be able to build 150 planes (they ended up building 13, out of which only 7 fulfilled the DoD). In subsequent sprints, Team 6 greatly improved their estimations, as well as their results. In sprint 2, they estimated 10 planes and built 14 (with 12 "correct" ones), whereas in sprint 3, they estimated 15 and built exactly 15 "correct" ones (out of the 16 planes in total).


Sprint 1

| Team | Estimate | Actual | | |
|---|---|---|---|---|
| 1 | 7 | 6 | 6 | -1 |
| 2 | 6 | 4 | 3 | -3 |
| 3 | 8 | 6 | 5 | -3 |
| 4 | 15 | 11 | 1 | -14 |
| 5 | 10 | 4 | 3 | -3 |
| 6 | 150 | 13 | 7 | |
| 7 | 25 | 24 | 21 | -4 |

Sprint 2

| | Estimate | Actual | | | Team |
|---|---|---|---|---|---|
| 1 | 15 | 14 | 9 | -6 | 1 |
| 2 | 7 | 7 | 5 | -2 | 2 |
| 3 | 12 | 13 | 7 | -5 | 3 |
| 4 | 14 | 16 | 15 | +1 | 4 |
| 5 | 6 | 13 | 10 | +4 | 5 |
| 6 | 10 | 14 | 12 | +2 | 6 |
| 7 | 25 | 24 | 21 | -4 | 7 |

Sprint 3

| | Est | Actual | | |
|---|---|---|---|---|
| 1 | 10 | 9 | 7 | -3 |
| 2 | 7 | 10 | 7 | 0 |
| 3 | 15 | 10 | 9 | -6 |
| 4 | 17 | 13 | 11 | -4 |
| 5 | 10 | 17 | 15 | +5 |
| 6 | 15 | 16 | 15 | 0 |
| 7 | 21 | 17 | 11 | -10 |

Figure 5. Estimates and actual figures for number of "correct" planes in the agile paper airplane game (7 teams, 3 sprints)

After this activity, in the usual style, the students were asked to provide their impressions. Again, they had time until the end of the day to submit their comments through the LMS, as a non-mandatory assignment. A number of 68 students provided feedback to the assignment described below.

*Please let me know your thoughts regarding how things went today - any thoughts!*
*For example:*
*- Did you enjoy the activity?*
*- How was the communication with your colleagues?*
*- What was "the best" part today? What was "the worst" part today?*
*- Would you prefer your group members to speak the same language as you?*
*(Is English ok for everyone? etc.)*

*#Please include any comment/suggestion you think is important.*

The students expressed their appreciation for this activity in various forms, as seen in the examples below.

- *"I enjoyed the activity. I got the feelings of scrum participant. It was a good activity to easily experience the scrum development."*
- *"This was the best activity so far. Probably because it was so educational."*
- *"Thank you very much for this kind of activity; it improve[s] the understanding of real world work."*
- *"Today's activity was very enjoyable. It was an innovative way to experience agile development."*
- *"Really appreciate for the wonderful activity!"*
- *"Overall, this was the best activity so far, because it was educational and I learned more than I did from books and lectures (I'm sorry)."*

During the informal discussions held by the instructor after class, she heard numerous opinions which classified this activity as the most enjoyable of the whole course. The game aspects which underpin game-based-learning proved to be very useful indeed: students had a taste of agile environments through a fun, game-like class activity.

Through this activity, the students became aware of some of the benefits of working in sprints, continuous improvement, retrospectives, as well as teamwork and the importance of communication and leadership.

For instance, one participant noted: *"I think the iterative improvement process is the most interesting"*. To go one step further, two comments regarding leadership stood out: *"These activities are good to find natural leaders"* and *"The leader decided the big folds and the order, which made me realize that the leader's presence was very important"*. Moreover, students recognized that managing people was another aspect which was illustrated through the activity: *"It was enlightening to learn about various aspects [...]. Equally valuable was the insight into people management and how motivating team members plays a pivotal role."*.

When it comes to teamwork and communication, many students emphasized the usefulness of the activity in this respect, as shown in their comments, some of which are included below.

- *"<The best> [part] is the group communication made me feel good, and I really like such kind of activity."*
- *"Communication was smooth for the most part. Everyone was eager to collaborate and share their ideas, which made the activity run smoothly."*
- *"The communication with my colleagues was effective and collaborative."*
- *"The best part today is the overall collaboration and teamwork were very enjoyable. It was great to see everyone working together and sharing ideas."*
- *"The process of folding airplanes is very interesting; everyone worked together in unity, which was great. There were no bad parts."*
- *"The communication between team members is very efficient, everyone expressed their ideas well, and everyone is very friendly."*

In terms of language, which could be viewed either as a barrier or as a bridge to communication, several students emphasized the importance of a common language (i.e., Japanese, for the local Japanese students), in order to communicate ideas clearly and effectively; the following two comments illustrate this idea.

- *"While English is fine and everyone can communicate, I think speaking in my native language would allow me to express my ideas more fluently and clearly."*
- *"Since Japanese is my first language, it would be easier for me if the group members could speak Japanese as much as possible. "*.

In one other instance, one of the (usually very active) students noted: *"In previous assignment I wrote I want to make a group with English speaker. I joined the English speaker group then, but I can't speak as fast as foreigners and they are so active, so I cannot act actively. It was more difficult than I expected. But, it doesn't mean I should do activity in same language group. I enjoyed this activity!"*. This student's reflection highlights both the challenges and benefits of participating in a multicultural group. After joining an English-speaking group, he found it difficult to keep up with the pace and high level of participation of more fluent members. Despite these difficulties, he recognized the value of the experience and, ultimately, found it enjoyable.

At the same time, students recognize that, more than the language, willingness to communicate is essential for an effective group work: *"I prefer my group members to speak same language of course because we'll get less communication mistake for language problem. However more important thing is not rather same language but rather will to communicate each other. With same language, we cannot communicate with unmotivated silent people."*

Some groups, made entirely of participants sharing a native language, performed very well and underlined the easiness of communication, as shown in the following comment: *"Since almost everyone in our group could say* [sic] *Chinese, I think our communication was effective and concise but enough for our work. After discussing our group work, we even had time to talk with each other."*. Another observation provided by a Chinese speaker (who can also communicate in Japanese) is relevant: *"As a Chinese student collaborating with Japanese peers, we used both Japanese and English to complement each other's understanding. This bilingual approach enhanced our teamwork and ensured that everyone was on the same page, which was critical for the success of the activity"*.

Similarly to the cooperative activity described in the previous sub-section, although teamwork is central to software engineering, not all students view it positively. Shy or less confident students may experience group activities as a source of anxiety rather than engagement, especially when they involve colleagues that they do not know well, as illustrated by the following comment: *"It was an excellent activity. At first I felt a little shy about talking and working together with strangers, but after doing the actual works, I think we knew each other more and became more harmonious."*.

In conclusion, as one student summarized, this activity contributed to improving the students' communication and collaboration skills, providing valuable experience for their future: "*Overall, this activity not only enhanced our teamwork and communication skills but also gave me a deep understanding of the importance of process optimization and continuous improvement in scientific research and engineering practice. As a graduate student in science, I will apply these insights to my future studies and research, striving to improve my comprehensive abilities."*

## V. Discussion and lessons learned

Teaching this introductory software engineering course in a multi-cultural classroom provided an invaluable experience, to both the students and the instructor. This section will offer further considerations regarding team formation, general course perceptions, as well as key lessons learned through the activities described in Section IV.

### *A. Team Formation*

While conducting the class activities, the students found themselves in different settings, in the form of different kinds of teams - a different one each time. The process of team formation can "significantly impact the learning process, the social behavior of team members, and the team's overall performance" [23]. This concept has long been studied in social sciences areas, like resource management, sociology and psychology. Forming a capable team is significant for all kinds of organizations, businesses, sports, etc. ([30] [31]), just like it is important during the process of education, when students work in teams. However, in software engineering, studies on team formation are still relatively limited, compared to studies focused on technology and process-related aspects [32].

As explained earlier, three different types of teams were created throughout the course: teams with Japanese students only, teams with international students only, and teams with a mixture of Japanese and international students. The instructor chose different team formation styles for the purpose of understanding the best setting, the one which allows the easiest communication between students.

As student feedback showed, there is no perfect solution; however, the different arrangements offered students different experiences. Some Japanese speakers prefer mono-cultural groups, whereas other prefer to be challenged to speak English or to be part of groups with people from different cultures. The same is true for the international students: some of them find communication with other international students easiest, whereas others are very eager to make Japanese friends. Even when students had the choice to create their own groups, without the instructor's interference, various types of groups were formed, based on the same principles as above. Further work is needed to identify the best method of team formation in collaborative projects, in order to yield the best results in a multicultural software engineering class.

## B. Student Course Evaluation

As explained in Section III, at the end of the course, as per university rules, the students are required to provide a course evaluation, based on a university-prepared questionnaire. Their answers are provided anonymously, following a Likert scale of 1 to 5, corresponding to "strongly agree", "agree", "neutral", "disagree" and "strongly disagree", respectively. As can be observed from the partially summarized results in Table II, with only a few exceptions of respondents choosing "neutral", all the participants selected either "strongly agree" or "agree" with the 5 statements. It is noteworthy that participants reported an increased interest in the subject following completion of the course. They mostly believed that the ways in which the instructor explained and planned the class contents were suitable for the course, and that there were sufficient opportunities to ask questions during class. Finally, their responses showed that the students were satisfied with the course: 19 chose "strongly agree", 11 chose "agree" and 1 chose "neutral".

TABLE II. RESULTS OF END OF COURSE QUESTIONNAIRE (1: STRONGLY AGREE; 2: AGREE; 3: NEUTRAL; 4: DISAGREE; 5: STRONGLY DISAGREE)

| | *1* | *2* | *3* | *4* | *5* |
|---|---|---|---|---|---|
| "The instructions were well prepared for the course." | 22 | 9 | 0 | 0 | 0 |
| "The ways the instructor explained and planned the class contents were suitable for the course." | 23 | 8 | 0 | 0 | 0 |
| "Attending this course, I developed a stronger interest in the field of study related to this subject than before." | 16 | 13 | 2 | 0 | 0 |
| "Overall, I am satisfied with this course." | 19 | 11 | 1 | 0 | 0 |
| "You were given sufficient opportunities for asking questions to the instructor(s)." | 24 | 5 | 2 | 0 | 0 |

The university-prepared course evaluation questionnaire allowed the provision of free comments, as well. One of them simply mentioned: *"I really like the group work and interacting with other students"*. In another comment, one student noted: *"This interactive opinion exchange class on software engineering principles provided a good opportunity for discussion of essential topics. It is well-designed. […] I was very happy to understand many different opinions!"* Notably, this student also included ideas for improving the course, by suggesting specific additional topics to be covered in the future (like SOLID design principles in C# [33]).

As one comment illustrates, at least some of the students were highly satisfied with the course: *"I am particularly grateful to [professor] for her dynamic teaching approach, which made complex concepts accessible and engaging. Her ability to convey intricate software engineering principles in a clear and practical manner significantly enhanced my understanding and interest in the subject. I have no suggestions for improvement at this time, as the course met all my educational needs. Thank you, professor […], for a truly enriching learning experience."*

Another set of data was obtained through the questionnaire prepared by the instructor at the end of the course. In this questionnaire, also anonymous, the participants were asked several questions; some of them elicited their opinions regarding multicultural classrooms, in terms of advantages/disadvantages of such an environment. Two more questions addressed the issue of whether cultural differences affect communication with their teachers, on one hand, and their student colleagues, on the other hand. The answers to these two questions are aggregated in Figure 6. As can be observed, about one third of the students believe that, in a multicultural classroom, communication is "definitely" affected or affected "most of the time" (a total of 36.67% students chose one of these two options, in both cases – with teachers and with colleagues). "Definitely not" was chosen by almost a quarter of the students (23.33%), for both situations of communication.

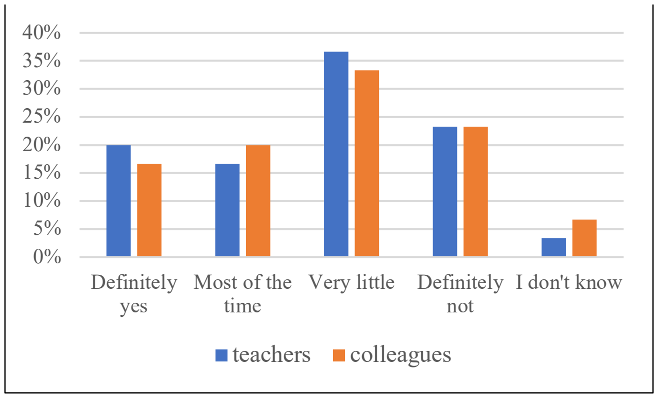


Figure 6. Answers to question "In your experience, do cultural differences affect communication/interaction with your teachers/student colleagues"?

## C. Key Lessons

The key lessons drawn from our experience, as presented in this paper, are the following.

a) The software engineering classroom can provide an environment for students to experience developing an actual software product. Instructors must provide practical experience as much as possible, not only theoretical concepts.

b) Collaborative learning is especially useful in a software engineering classroom because it mirrors real-world team-based development, enabling students to practice technical problem-solving alongside essential communication and teamwork skills.

c) Games make learning fun and practical; game-based learning is valuable in a software engineering classroom because it increases motivation and engagement, while allowing students to apply theoretical concepts through interactive practice.

d) Multicultural classrooms provide students with an experience that reflects global work settings and offers a glimpse into software engineering practices across cultures.

e) The need for effective strategies to engage students in the classroom cannot be overstated. Fulfilling course objectives and achieving student satisfaction are closely tied to engagement, as higher levels of involvement typically translate into greater effort and commitment to learning.

## VI. Conclusions and Future Work

Based on the experience of teaching a graduate software engineering course to a multicultural group of students, this paper highlighted ways to achieve student engagement through collaborative activities and games. Collaborative learning is an excellent way to prepare students for real-world, team-based software development, whereas game-based learning boosts motivation and helps students apply theory through interactive practice. Our study showed that group work in multicultural classrooms not only exposes students to linguistic and cultural differences but also helps them recognize the importance of stepping outside their comfort zone for personal and professional growth.

Future work will focus on evaluating the extent to which course objectives are met through the proposed strategies, and on identifying the most effective approaches for optimizing cooperative and game-based learning, particularly in multicultural settings, with attention to overcoming language and cultural barriers through suitable team formation. Moreover,. artificial intelligence can be considered to provide useful tools to overcome the language barriers in class.

## References

[1] S. Vasilache, "Working together: Class activities as cross-cultural bridges in software engineering teaching," in *Proc. ICSEA 2024.*

[2] The Joint Task Force on Computer Science Curricula, "Computer science curricula 2023," [Online]. Available: https://ieeecs-media.computer.org/media/education/reports/CS2023.pdf. [Accessed: Nov. 30, 2025].

[3] S. Vasilache, "Bringing interactive instruction to the software engineering classroom: A multicultural group case study," in *Proc. IEEE Global Engineering Education Conf. (EDUCON)*, 2025, pp. 1–5.

[4] E. Lopez-Fernandez, P. Tovar, P. P. Alarcon, and F. Ortega, "Motivation of computer science engineering students: Analysis and recommendations," in *Proc. 2019 Frontiers in Education Conf. (FIE 2019)*, 2019.

[5] D. López-Fernández, A. Gordillo, P. P. Alarcón, and E. Tovar, "Comparing traditional teaching and game-based learning using teacher-authored games on computer science education," unpublished.

[6] M. A. Alsubaie, "Examples of current issues in the multicultural classroom," *J. Educ. Pract.*, vol. 6, no. 10, pp. 86–89, 2015.G. G. Borges and R. C. G. de Souza, "Skills development for software engineers: Systematic literature review," *Inf. Softw. Technol.*, vol. 168, p. 107395, 2024.

[7] M. Malcon-Cervera and C. Montaudon-Tomas, "Multicultural classrooms: Advantages for foreign and local students. A comparative study," in *Proc. EDULEARN17*, IATED, 2017, pp. 6477–6485.

[8] S. Vasilache, "Bringing interactive instruction to the software engineering classroom: A multicultural group case study," in *Proc. IEEE Global Engineering Education Conf. (EDUCON)*, 2025, pp. 1–5.

[9] M. Laal and M. Laal, "Collaborative learning: What is it?," *Procedia – Social Behav. Sci.*, vol. 31, pp. 491–495, 2012.

[10] M. Laal and S. M. Ghodsi, "Benefits of collaborative learning," *Procedia – Social Behav. Sci.*, vol. 31, pp. 486–490, 2012.

[11] "Collaborative learning vs. cooperative learning in the classroom," Promethean World, [Online]. Available: https://www.prometheanworld.com/resource-center/blogs/collaborative-vs-cooperative-learning/. [Accessed: Nov. 30, 2025].

[12] I. Garcia, C. Pacheco, F. Méndez, and J. A. Calvo-Manzano, "The effects of game-based learning in the acquisition of 'soft skills' on undergraduate software engineering courses: A systematic literature review," *Comput. Appl. Eng. Educ.*, vol. 28, no. 5, pp. 1327–1354, 2020.

[13] M. J. Sousa and Á. Rocha, "Game-based learning contexts for soft skills development," in *Proc. World Conf. Inf. Syst. Technol.*, Cham: Springer, 2017, pp. 931–940.

[14] B. S. Tan and K. S. Chong, "Unlocking the potential of game-based learning for soft skills development: A comprehensive review," *J. ICT Educ.*, vol. 10, no. 2, pp. 29–54, 2023.

[15] J. A. Medina-Merodio, A. Castillo-Martinez, R. Barchino, R. Estriegana, and R. Robina-Ramírez, "Factors influencing the acquisition of soft skills in a collaborative learning environment supported by game-based application," *IEEE Access*, 2024.

[16] J. Martí-Parreño, A. Galbis-Córdova, and M. J. Miquel-Romero, "Students' attitude towards the use of educational video games to develop competencies," *Comput. Hum. Behav.*, vol. 81, pp. 366–377, 2019.

[17] E. O'Farrell, M. Yilmaz, U. Gulec, and P. Clarke, "Playsafe: Results from a virtual reality study using digital game-based learning for safe agile software development," in *Proc. Eur. Conf. Softw. Process Improvement*, Cham: Springer, 2021, pp. 695–707.

[18] "The scaled agile framework (SAFe)," [Online]. Available: https://framework.scaledagile.com/#big-picture. [Accessed: Nov. 30, 2025].

[19] S. Beecham, T. Clear, J. Barr, M. Daniels, M. Oudshoorn, and J. Noll, "Preparing tomorrow's software engineers for work in a global environment," *IEEE Softw.*, vol. 34, no. 1, pp. 9–12, 2017.

[20] P. Schmiedmayer et al., "Global software engineering in a global classroom," in *Proc. ACM/IEEE 44th Int. Conf. Softw. Eng.: Softw. Eng. Educ. Training (ICSE-SEET)*, 2022, pp. 113–121.

[21] S. Vasilache, "A taste of distributed work environments: Emergency remote teaching and global software engineering," in *HCI Int. 2021 – Posters*, C. Stephanidis, C. Antona, and S. Ntoa, Eds. Cham: Springer, 2021, pp. 624–628.

[22] "Manaba," [Online]. Available: https://manaba.jp/products/. [Accessed: Nov. 30, 2025].

[23] J. Vilela, S. C. dos Santos, and D. Maia, "Impact of team formation type on students' performance in PBL-based software engineering education," in *Proc. CSEDU (2)*, 2024, pp. 327–338.

[24] IEEE Computer Society, "Software engineering body of knowledge," [Online]. Available: https://www.computer.org/education/bodies-of-knowledge/software-engineering. [Accessed: Nov. 30, 2025].

[25] A. M. O'Donnell and C. E. Hmelo-Silver, "Introduction: What is collaborative learning? An overview," in *The Int. Handbook of Collaborative Learning*, 2013, pp. 1–15.

[26] G. Rodríguez-Pérez, R. Nadri, and M. Nagappan, "Perceived diversity in software engineering: A systematic literature review," *Empirical Softw. Eng.*, vol. 26, no. 5, p. 102, 2021.

[27] A. C. Homan, D. Van Knippenberg, G. A. Van Kleef, and C. K. De Dreu, "Bridging faultlines by valuing diversity: Diversity beliefs, information elaboration, and performance in diverse work groups," *J. Appl. Psychol.*, vol. 92, no. 5, p. 1189, 2007.

[28] B. Willmott, "The agile paper airplane game," [Online]. Available: https://www.ppm.academy/post/the-agile-paper-airplane-game. [Accessed: Nov. 30, 2025].

[29] "Scrum," [Online]. Available: https://www.scrum.org. [Accessed: Nov. 30, 2025].

[30] P. Zainal, D. Razali, and Z. Mansor, "Team formation for agile software development: a review," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 10, no. 2, pp. 555–561, 2020.

[31] D. Strnad and N. Guid, "A fuzzy-genetic decision support system for project team formation," *Appl. Soft Comput.*, vol. 10, no. 4, pp. 1178–1187, 2010.

[32] M. I. Yilmaz, R. V. O'Connor, R. Colomo-Palacios, and P. Clarke, "An examination of personality traits and how they impact on software development teams," *Inf. Softw. Technol.*, vol. 86, pp. 101–122, 2017.

[33] N. C. Ramachandrappa, "SOLID design principles in software engineering," *Int. J. Comput. Trends Technol.*, vol. 72, no. 9, pp. 18–23, 2024.

[34] R. G. Tweed and D. R. Lehman, "Learning considered within a cultural context: Confucian and Socratic approaches," *Am. Psychol.*, vol. 57, no. 2, pp. 89–99, 2002.

[35] D. Byrne, *The Attraction Paradigm*, vol. 11. Cambridge, MA, USA: Academic Press, 1971.

# Vessel Route Planning with Worker Schedule Optimization for Offshore Windmill Maintenance

E. De Kuyffer, L. Martens, W. Joseph, T. De Pessemier
Department of Information Technology
Ghent University/IMEC
Ghent, Belgium
e-mail: Erik.DeKuyffer@UGent.be, Luc1.Martens@UGent.be, Wout.Joseph@UGent.be, Toon.DePessemier@UGent.be

***Abstract*—The high fuel prices and the important costs linked to the down-time of the windmills during maintenance urge the need for minimization of the travel time and the scheduling of jobs within a minimal time span. Since landing in windmills at sea is difficult and depends on meteorological parameters, the constraint of maintenance windows is added when searching for the optimal route. To minimize the distance traveled, the Vehicle Routing Problem with Time Windows (VRPTW) is solved, using three different methods. The VRPTW is applied to two separate databases, namely various sets of windmills to be maintained and several numbers of customers to be serviced. Applications with 8 to 175 windmills, divided over 3 farms have shown that the VRPTW solved by using three different methods resulted in a comparable relative gain in travel distance, compared to a randomly chosen route. The main difference between the methods studied is the amount of calculation time needed, which varies from 1 second to 6 minutes for the different methods. To demonstrate the general applicability, the same three methods were executed on a set of service tasks performed on 8 to 40 customers of a window decoration company, distributed throughout Belgium, resulting in similar results. In a second part of the paper, the Job Shop Scheduling Problem (JSSP) is solved to minimize the total maintenance span of offshore windmills as an additional objective function. This led to a relative gain of up to 62% in maintenance time, compared to the total maximum maintenance span for an application of 40 windmills. Finally, both objectives - minimal distance and minimal maintenance time span - are combined, resulting in a set of non-dominated maintenance sequences that the planner can use [1].***

***Keywords*-VRPTW; VRPy; OR Tools; ACO; Job Shop Scheduling; Pareto.**

## I. Introduction

Due to high fuel prices and significant labor costs, it is extremely important, especially for offshore windmill management companies, to limit the distance covered and time consumed for offshore windmill maintenance. Expressed in numbers, the maintenance cost ranges from 20% to 25% of the total Levelised Cost of Electricity (LCOE) of contemporary wind power systems as shown in the Guide to an Offshore wind farm [2]. The costs of Operation and Maintenance (O&M) of offshore wind farms are significantly higher than those of land-based windmills, due to the difficulties associated with the offshore environment. They could vary between USD 0.027 and USD 0.048/kWh as calculated by the International Renewable Energy Agency in 2012 [3]. By improving the durability of the turbines and increasing the size of the windmills, the maintenance cost per windmill has dropped significantly to 1.5 to 2% per year of the original turbine investment. However, because of economic evolution, the actual maintenance to be performed is becoming more expensive again. Operation and Maintenance costs can be divided into a limited number of components, which are: Insurance, Spare Parts, Administration, Regular maintenance, and Repair [4]. The research carried out for this article focuses on the reduction of regular maintenance and repair costs, denoted in the above list of Operational Expenses (OPEX) for offshore windmill farms. The maintenance of offshore windmill parks can be divided into three specific maintenance types, namely corrective, preventive (and predictive) and inspection maintenance, although the latter is often considered an operational activity or part of preventive maintenance [5]. Maintenance inspection consists of evaluating the condition of the windmill or ground station in order to determine what tools, materials, and work are needed to keep it in optimal working state [6]. Preventive maintenance is planned maintenance - typically quarterly, half-yearly, and yearly - of the assets to make sure electricity production is stable and unexpected equipment failure, leading to costly and unplanned downtime, is prevented. Preventive maintenance is work scheduled based on calendar time, asset run-time, or other time periods, while predictive maintenance is scheduled as-needed based on real time conditions of the assets [7]. Finally, corrective maintenance consists of maintenance tasks that are performed to identify, isolate, and treat an issue in order to restore equipment, a machine, or a system to an operational condition so it can perform its intended function. Typical preventive maintenance actions on windmill farms are electrical and mechanical in nature [8]. When planning these maintenance interventions, the availability of the vessel and workers must be taken into account, as well as weather parameters, sea currents, and wave heights. Optimization of vessel routing for offshore windmill maintenance is a very complex problem. It has been the subject of recent studies [9]–[12]. As discussed previously, it is critical to obtain a good understanding of wind direction and speed, wave height, and other parameters to decide whether a vessel can leave the dock for maintenance. A stranded vessel is very expensive, and therefore planning is crucial to avoid as much as possible inoperative maintenance vessels.

To demonstrate the general applicability of the VRPTW to obtain minimal distance routes and to show that all three methods used lead to similar results for other data sets, the procedures are applied to discrete product installation planning.

With the fast-growing consumer demand for Taylormade products, the customer requires efficient distribution, installation, and maintenance planning. Optimization of vehicle routes to satisfy the customer and reduce fuel consumption and $CO_2$ emissions has become a hot topic [13]. The interventions of companies that distribute and maintain unique products per customer - the so-called Value Added Resellers or VARs - can be split in the installation of the products and ad hoc maintenance of previously installed products [14]. The planning of the delivery and installation can be considered as proactive planning, allowing optimization of the distance to be traveled, and thus the amount of fuel used. Maintenance interventions are more reactive in nature, making optimal planning more difficult. In typical operations of companies installing and maintaining discretely manufactured products, a mix of both types of interventions is to be planned, making it a complex task to determine the optimal routes.

If the actions of a distributor are limited to the instantaneous drop-off of products and there are no time limitations or capacity restrictions, the routes to be followed can be optimized by solving the Traveling Salesman Problem (TSP) [15][16]. The lack of time limitations means that the distributor can drop off or pick up products at any time of the day, as the customer does not necessarily need to be present. However, when intervention, be it installation or maintenance, requires a certain amount of time and the presence of the customer at the site, time windows come into play, and more vehicles need to be deployed simultaneously. In the latter case, the VRPTW will be solved, taking into account that there are no capacity constraints. This VRPTW problem will be solved for a data set of 8 to 40 customers with given coordinates that require an installation or service.

In a second part of this paper, we focus on another strategy to lower the maintenance costs of an offshore windmill farm, namely the reduction of the downtime by arranging the maintenance jobs in such a way that the total service time span is minimized. To obtain this objective, the Job Shop Scheduling Problem (JSSP) is solved, in which the machines are replaced by workers. For each worker, a sequence is calculated so that all maintenance jobs are executed within a limited time frame, reducing the total downtime of all the windmills that need service. Where the first part focuses on limiting the distance of all maintenance routes, the second part is thus focused on reducing maintenance time. In the last part of this paper, both objectives are applied to the same set of windmills, resulting in a Pareto front of non-dominated solutions offered to the planner to choose from. It will become clear from the list of these Pareto points that reaching both objectives at the same time is nearly infeasible, and thus the optimal sequence must be chosen from this list.

The novelties of this paper are:

- Comparison of three solution methods for VRPTW applied to windmill maintenance vessels and to discrete product installation and service.
- The optimization of the combined windmill sequence travel distance and maintenance time span by solving both VRPTW and JSSP on the same data set.
- The importance of achieving both goals in reducing maintenance costs.

The remainder of the paper is organized as follows. In Section II, references are made to related work and Section III describes the problem formulation. The three methods used to solve the Vehicle Routing Problem with Time Windows, as well as the solution method for the JSSP are listed in Section IV. Section V lists the results of all the optimization methods discussed for VRPTW and JSSP and compares both by calculating the corresponding Pareto points. Finally, Sections VI and VII contain an evaluation of the results and provide a conclusion, respectively.

## II. Related work

The Vehicle Routing Problem (VRP) was first instigated more than six decades ago (1959) by Dantzig and Ramser under the title The Truck Dispatching Problem. The study of this routing problem led to major developments in the fields of exact algorithms and heuristics [17]. The VRP comprises the design of least cost delivery routes through a set of geographically dispersed locations, subject to one or more side constraints. The VRP thus plays an important role in distribution management, and tens of thousands of carriers worldwide are faced with it daily [18]. Constraints to vehicle routing problems linked to capacity result in the Capacitated Vehicle Routing Problem (CVRP). If a time window is added to each location, asset, or customer, we talk about the vehicle routing problem with time windows [19]. In addition to the capacity constraint, a vehicle in the VRPTW has to visit a location, asset, or customer within a certain time frame. The vehicle - car, vessel, or other - is allowed to arrive before the time window opens, but the customer or asset cannot be serviced until the respective time window opens. In addition, it is not allowed to arrive after the time window has closed [20]. According to Goel and Maini [21], there are different solution methodologies for VRP, which can be divided into three categories: Exact methods, heuristics, and meta-heuristics. The exact methods generate optimal solutions and guarantee their optimality. This method class includes a variety of approaches, mainly branch and X (X being cut, bound, price, etc.), dynamic programming, and column generation methods. The heuristics aim to methodically find an acceptable solution within a limited number of iterations. Meta-heuristics can finally be defined as a class of heuristics that search beyond the local optima if they exist [21].

Research papers on VRPs, with or without time windows, are quite common, since application in daily life is widely spread, for example, in the delivery of packages and the route planning of nurses [22]. Jayarathna et al. for example described in their paper the study to implement a better route plan that optimizes the truck allocation system at the lowest possible costs of transportation, warehouse, and administration for a large-scale Fast Moving Consumer Goods (FMCG) company [23]. Arnold et al. explored how to design a local search heuristic that generates good solutions for very large-scale CVRP instances in an acceptable computational time [24]. In [25], Irawan et al.

defined a Mixed Integer Linear Programming (MILP) model to find the optimal maintenance schedule for turbines, the best routes for crew transfer vessels (CTV) to service the windmills, as well as the optimal number of technicians required for each vessel. Stalhane et al. furthermore used an arc flow and a path flow formulation model in instances of different numbers of vessels and maintenance tasks on offshore windmill farms [26]. The principle of using VRPTW to optimize offshore wind farm maintenance routes for multiple vessels has never been applied. Instead of developing a new model, this paper uses existing VRPTW solvers and compares them to define the best method, both in the distance reduction obtained and in the calculation time used. While in the studied research papers the focus is on describing the VRP and developing new solution methods (also for windmill maintenance), this paper thus focuses on comparing existing methods on usability and optimization results.

Second, there has been extensive research on production scheduling according to the flow-shop method and the job-shop method. Al-Shayea et al. designed a model to integrate production scheduling and maintenance planning for flow-shop production systems. This model is based on the optimal sequence of jobs for jobs that will be processed on several machines connected in series. The objective of this study is to find the optimal sequence of jobs, while reducing total production and maintenance costs [27]. Han et al. present an improved iterated greedy algorithm for the distributed flow shop scheduling problem [28], while in [29], Kerem Bülbül and Philip Kaminsky describe a decomposition heuristic as a solution approach for a large class of job-shop scheduling problems. Yu summarizes the research development and the current situation of job-shop scheduling problems. He divides the existing research methods into different classes and talks about the future research direction of job-shop scheduling problems [30]. However, none of these papers apply to maintenance scheduling in windmill farms, while costs for this type of maintenance are very high, and every hour of downtime (due to maintenance) result in an important loss of revenue. This paper offers insight into how production scheduling can be applied to maintenance, especially for offshore windmill parks.

## III. Problem Formulation

This section describes the experimental design for both VRPTW and JSSP, and the output data obtained after solving these problems.

### A. Experimental Design VRPTW

Figure 1 shows an example of a windmill configuration after applying the VRPTW solution method. The configuration used in this example has one dock and 16 windmills spread over three farms to be serviced. The different windmills are indicated as $WM_i$ (i = 1-16), where each windmill must be visited exactly once by one of the vessels. Solving the VRP leads to three routes that three different vessels must take. Figure 2 shows the result of the application of the VRPTW to

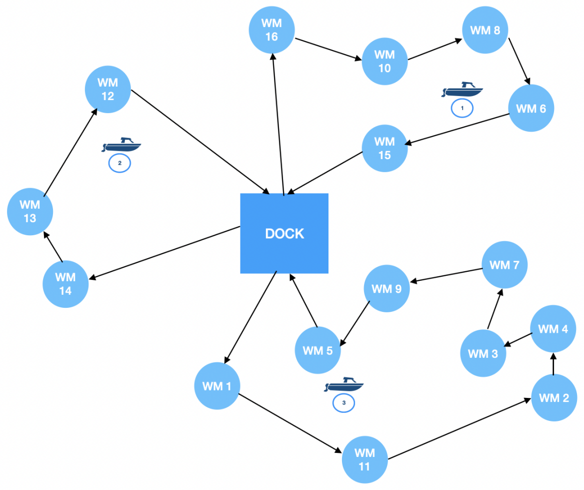


Figure 1. Experimental vessel routing for a configuration of 16 windmills after solving VRPTW

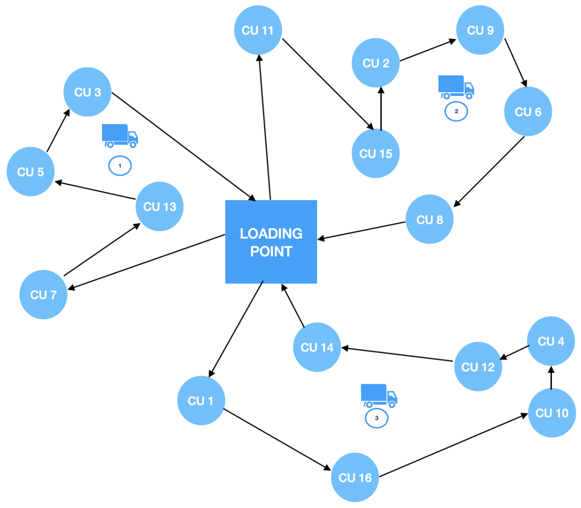


Figure 2. Experimental van routing for a configuration of 16 customers after solving VRPTW

a set of 16 customers in the discrete manufacturing environment. The different customers are indicated as $CU_i$ (i=1-16), where each of the customers has to be visited exactly once. Similar results are obtained and the same conclusions can be drawn as for the windmill use case.

### B. Experimental Design JSSP

Figure 3 shows an example of a job sequence per worker obtained by solving the job-shop scheduling problem for a group of 16 windmills spread over 3 farms. The number of workers is set to three on the vertical axis, in analogy with the number of machines in the original JSSP used in a production environment. In each windmill, one worker needs to perform a service task and each worker needs to perform several maintenance tasks in separate windmills. Applying the JSSP solver to this configuration leads to an optimal sequence in which each worker needs to perform service on the windmills

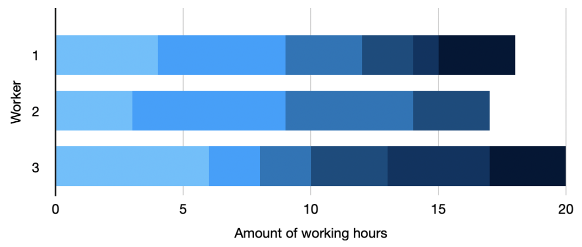


Figure 3. Experimental maintenance planning for 3 workers for a configuration of 16 windmills after solving the JSSP

he or she is responsible for, with a different service time on each windmill, shown on the horizontal axis. Thus, for each worker, a sequence of jobs is shown, each corresponding with a different color and with a size in accordance with the length of the job. In contrast to VRPTW, where the optimal number of vessels is calculated, the number of workers can be chosen. More workers will lead to a shorter total maintenance span, but the optimum in the configuration has been shown to be 3. More workers will not lead to a significant increase in downtime.

### C. Output Parameters

The output parameters for the VRPTW problem are the optimal number of ships or trucks, the optimal sequence, and the travel time for each vessel or truck in minutes. These are compared with the result obtained by using a Genetic Algorithm (GA) for the Traveling Salesman problem (TSP) where no time-frame restrictions are present and only one vessel or truck is used. Furthermore, the relative gain in travel time ($\Delta G_t$) is calculated by dividing a randomly chosen travel time ($TTt|Random$) minus the total travel time (TTt) by a randomly chosen travel time (see Equation 1). The output parameters for the JSSP problem are an optimized sequence of maintenance tasks per worker to minimize total downtime (TDt). This is then compared to the total time needed for one worker ($TTt|OneWorker$) and a relative gain is calculated ($\Delta G_w$) using Equation 2.

$$\Delta G_t = 100 \cdot \frac{TTt|Random - TTt}{TTt|Random}(\%) \qquad (1)$$

$$\Delta G_w = 100 \cdot \frac{TDt|OneWorker - TDt}{TDt|OneWorker}(\%) \qquad (2)$$

## IV. Method

In this section, the three different methods to solve VRPTW will be discussed, as well as the JSSP solution algorithm and the use of the Pareto front to obtain results with combined optimization for both problems.

### A. Solution method - VRPTW

When VRP involves scheduling visits to windmills that are only maintainable during specific time windows, the problem becomes VRPTW [31][32]. If there are no additional restrictions, such as time windows or load capacity, and all maintenance or installations can be performed consecutively, the optimal solution for one team is found by assigning just one vessel to visit all locations once and identifying the minimum distance route for that vessel. In this case, the VRP can be seen as a generalization of the Traveling Salesman problem. In this paper, there is an optimization of the minimization of the longest single route between all vessels. In fact, the goal of solving the VRPTW will therefore be to complete all maintenance tasks as soon as possible, taking into account that some maintenance tasks can only be serviced for a certain period of time. If we further incorporate the load capacity as a limiting factor, the problem to solve becomes a Capacitated Vehicle Routing Problem with Time Windows (CVRPTW). However, for both data sets, the load is never an issue, neither for the vessels that solely need to transport maintenance people, nor for the vans big enough to carry all products that need to be installed in one day. Both data sets are directly obtained from the windmill maintenance company and the window decoration value added reseller, and no prefiltering of preprocessing was done, except a random selection of a predefined batch out of the total set, ranging from 8 to 175 windmills and from 8 to 40 VAR customers.

There are several ways to solve VRPTW, such as an exact approach, a heuristic or constructive method, and a meta-heuristic solution method, such as a genetic algorithm (GA) [33][34]. In this paper, three methods are discussed, namely VRPy, a tool using a column generation approach (CGA), the OR Tools solver, developed by Google Operational Research, and an ACO algorithm, a meta-heuristic solving method. Table 1 summarizes the three methods and describes their characteristics, where the speed is measured on a MacBook Pro from 2021 with the new Apple M1 chip and 8Mb RAM.

TABLE I. Comparison of all methods used to solve the VRPTW

| Method | Advantage | Disadvantage |
|---|---|---|
| VRPy | Easy Interface | Less Powerful |
| OR Tools | Fast and Accurate | No optimal result |
| ACO | Optimal results | No Easy Interface |

To calculate the distances between the windmills or customers and between the starting point and the windmills or customers, spherical trigonometry formulas are used. In this paper, all vehicles are considered the same: they have the same velocity, the same capacity, and unit freight. Furthermore, the capacity and cargo of the vessel are not considered constraints. When defining $t_i$ as the time it takes for the vessel to arrive at location i, e as the cost of waiting and f as the cost of arriving too late, the objective of solving the VRPTW for a collection of vehicles A, can be written as:

$$min(\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{a=1}^{A} x_{ija} * d_{ij} + \sum_{i=1}^{N} max\{e*(m_{bi}-t_i);0;f*(t_i-m_{ei})\} \qquad (3)$$

Where:

$$x_{ija} = \begin{cases} 1 & \text{if the vehicle a travels from i to j,} \\ 0 & \text{in all other cases} \end{cases} \qquad (4)$$

$$t_{ija} = \sum x_{ija}(t_i + \frac{d_{ij}}{v} + s_i) \quad (t_0 = 0, s_0 = 0) \tag{5}$$

The constraints are:

$$\sum_{j=1}^{N}\sum_{a=1}^{A} x_{jia} = \sum_{j=1}^{N}\sum_{a=1}^{A} x_{ija} = A \quad (i = 0) \tag{6}$$

$$\sum_{j=0}^{N}\sum_{a=1}^{A} x_{ija} = 1 \quad (i \in N) \tag{7}$$

$$\sum_{i=0}^{N}\sum_{a=1}^{A} x_{ija} = 1 \quad (j \in N) \tag{8}$$

$$\sum_{j=1}^{N} x_{ija} = \sum_{j=1}^{N} x_{jia} = 1 \quad (i = 0 \quad a \in A) \tag{9}$$

In (3), the second part of the equation, sum of maximums, defines the time window constraint. In (5), $t_{ija}$ is the time it takes the vehicle to travel from location i to j, v is the speed and $s_i$ the service at location i. At the depot (node 0 in the equations), both t and s are equal to zero. The constraint in (6) implies that the number of vehicles that start from the loading point and go back there is A. Constraints (7) and (8) mean that each location can be visited only by one vehicle. Finally, constraint (9) represents that all vehicles that start at the loading point also go back there.

VRPy solves VRP with a column generation approach [35]. The term refers to the fact that, continuously, routes are generated with a pricing problem and fed to a restricted master problem. The latter selects the best routes among a pool so that each node (windmill or customer in this case) is serviced exactly once. The pricing problem is actually a shortest elementary path problem. Additional constraints, such as the time windows discussed in this paper, contribute to a shortest-path problem with resource constraints. VRPy does not lead to an optimal solution, even without time limits. Hence, when solving pricing problems does not result in a route with negative marginal cost, the master problem is solved as mixed integer programming (price-and-branch strategy).

Next, the above solution will be compared with the results found for the same operational VRPTW using the solver developed for Google OR Tools (Table I) [36]. The algorithm based on the Python routing library wrapper results in a new set of optimal routes, taking into consideration that all windmills or customers need to be serviced in a specific time frame. As for the first method, no other restrictions are taken into account. The algorithm used to solve VRPTW starts with the creation of input data, followed by a callback function. After adding the time constraints, the default search parameters and a heuristic method are set for the first solution. Finally, the same function is used to solve the Traveling Salesman Problem (TSP), resulting in the route for each vehicle, the total travel time of the vehicle route, and the solution windows for each location. The solution window at a location is defined as the time interval during which a vehicle must arrive, so it stays on schedule. The pseudocode for solving VRPTW with a solver can be written as:

```
procedure VRPTW Solver is
    data model creation
    declare the routing model
    add the constraints
        define distance and time callback
        set time windows at each location
        define the number of vehicles
    set a search parameter(e.g., time limit)
    call the solver
end procedure
```

The third method to solve VRPTW with the same entry data, such as the position of the windmills or customers and service windows, is based on an ant colony optimization algorithm [12][37]. Ant Colony Optimization (ACO) is one of the most recent metaheuristic approaches to combinatorial optimization problems. The pseudocode is shown below. All three solution methods, VRPy, OR Tools, and ACO are heuristic methods. When it is impossible or impractical to find an optimal solution, heuristic methods can be used to accelerate the process of discovering a satisfactory solution.

```
procedure ACO Meta-Heuristic is
    while not terminated do
        ConstructAntsSolutions()
        UpdatePheromones()
        daemonActions()
    repeat
end procedure
```

The ACO is based on the foraging behavior of real ants. They arbitrarily explore the environment, using pheromone deposits to find the shortest routes. Therefore, ACO algorithms are probabilistic techniques suitable for solving optimization problems that aim at minimizing the distance traveled (e.g., TSP and VRP). In the first step, *ConstructAntsSolutions* - of the algorithm, each artificial ant generates a solution: thereby it randomly chooses the next city to visit, based on a heuristic combination of the distance to that city and the amount of virtual pheromone left behind on the arc to that city. The ants explore and dump the pheromone on each arc they traverse until they have all completed a tour (see Equation (10)). At this point, the ant that has completed the shortest tour deposits virtual pheromone along its entire route (*UpdatePheromones*). Equation (14) shows that the amount of pheromone deposited is inversely proportional to the length of the tour. Thus, the shorter the route, the more pheromone the ant deposits on the arcs of the corresponding tour. The *deamonActions* procedure is used to carry out centralized actions that cannot be carried out by individual ants, as they do not possess global knowledge.

A typical example of these *deamonActions* is the collection of global information that can be used to decide whether it might be useful to deposit additional pheromone to bias the search process from a nonlocal perspective. As long as the termination condition is not met, these three steps are repeated [38]. The pheromone $\tau_{ij}$, associated with the edge joining locations i and j, is updated as follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \qquad (10)$$

where:

$$\rho = \text{evaporation rate,} \qquad (11)$$

$$m = \text{number of ants,} \qquad (12)$$

$$\Delta\tau_{ij}^{k} = \text{the quantity of pheromone laid on edge (i, j) by ant k,} \qquad (13)$$

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{L_k} & \text{if ant k used edge (i,j) in its tour,} \\ 0 & \text{in all other cases} \end{cases} \qquad (14)$$

$$Q = \text{a constant,} \qquad (15)$$

$$L_k = \text{is the length of the tour built by ant k,} \qquad (16)$$

To determine and confirm that the solutions obtained by the three methods to solve VRPTW are applicable in other areas, the same solution procedures were applied to two data sets. In addition to windmill farms, a data set is used consisting of the customer coordinates at which discrete Taylormade products (curtains) are to be installed and maintained. Although this data set differs quite extensively from that of the windmills, the optimization goals are the same, namely travel time, and thus fuel consumption reduction. Transport for Taylormade products goes over land and cannot follow a straight line, the distances between customers are smaller than for the windmill farms (typically a few tens of kilometers versus a few hundreds for the windmills) and not clustered around different farms, making the data sets for windmills and customers quite different.

## B. Solution Method - Job shop

To apply the solver to maintenance planning, we have made the following assumptions: the machines in the JSSP are replaced by the workers performing maintenance jobs (the job is a sequence of windmills to be serviced), and the tasks are linked to the windmills. The processing time is chosen randomly, as are the workers for each maintenance job. The final result of the algorithm created to solve the JSSP will be a schedule optimized for each worker to minimize the total maintenance span. The pseudocode of the algorithm used to solve JSSP with the OR solver is [39]:

```
procedure JSSP Solver is
    data creation do
        define set of machines (M) and jobs (J)
        define the processing time for each task i in job j
        define the worker sequence for each job
    declare the optimization CP model
    set decision variables for each task i in job j
    add the constraints
        add the duration constraint
        add the precedence constraint
        add the disjunctive constraint
    define the objective function
    call the solver
end procedure
```

Each of the steps in the pseudocode is further defined as:

- Data Creation: For each maintenance job, several tasks are defined, that is, the windmills or customers to be serviced. For every windmill or customer (task), the worker (in analogy with the machine) who needs to perform the task and the service time needed are given.
- Declaration of the model, a Constraint Programming (CP) model that includes variables and constraints that will be solved via the CP solver.
- Definition of the decision variables, which are the start and end time of each task, the duration (end minus start) and the interval of the task.
- Defining the constraints: A worker cannot work at two windmills at the same time (disjunctive constraint), for any two consecutive tasks in the same maintenance job, the first must be completed before the second can be started (precedence constraint), and the processing time of each job must be equal to the difference between end time and start time of the job (duration constraint).
- Determining the objective as the minimization of the make or maintenance span.
- Call the solver and show the results.

The objective function (17) and the constraints of the job shop scheduling problem are written as:

$$min\ C_S \qquad (17)$$

where:

$$C_{1j} - w_{1j} = r_j + p_{1j} \qquad \forall j \qquad (18)$$

$$C_{i-1j} - C_{ij} + w_{ij} = -p_{ij} \qquad i = 2, ..., m_j, \quad \forall j \qquad (19)$$

$$C_{ik} - C_{ij} \geq p_{ik} \quad \text{or} \quad C_{ij} - C_{ik} \geq p_{ij} \quad \forall i, \quad \forall j, k \in J_i \qquad (20)$$

$$C_{ij}, w_{ij} \geq 0 \quad i = 1, ..., m_j \quad \forall j \qquad (21)$$

Constraint (18) implies that a maintenance job can only start after its respective ready time. Constraint (19) specifies that a job j follows its processing sequence. The machine capacity constraint (20) finally ensures that a worker can process only one operation at a time, and an operation will be finished once it starts.

## C. Pareto front

Solving VRPTW and JSSP leads to sequences in which windmills or customers need to be visited, each with a different objective function. Hence, for the VRPTW, the objective is to minimize the total distance traveled, while the JSSP attempts to reduce the total maintenance time. Since both objectives are possibly contradictory, the solution methods are compared by calculating Pareto points and a corresponding Pareto front. Therefore, the maintenance sequences resulting from the VRPTW, with a minimal route distance, are offered to the second objective function to calculate the corresponding total maintenance time. Additionally, maintenance sequences with minimal maintenance are calculated by solving the JSSP and the corresponding total route distance is determined. Both lead to a set of two-dimensional coordinates of which the Pareto points are calculated. Pareto optimal points are non-dominated, meaning that there does not exist another solution that rigorously dominates the Pareto optimal solution in terms of any objective. The Pareto front is the multi-objective and multi-dimensional alternative for the individual optimal solution resulting from single objective optimization problems (VRP and JSSP).

## V. Results

The result section gives an overview of all results obtained by solving VRPTW and JSSP separately. In addition, a Pareto front is calculated with non-dominated solutions, optimizing both problems simultaneously.

### *A. Sequence comparison VRPy - OR Tools - ACO for windmill maintenance*

Table II lists the best results obtained by applying all three solution methods, and this for different configurations, ranging from 8 to 175 windmills. The relative gain shows how much better the optimized solution is than the randomly chosen one. The optimal number of vessels proposed by the VRPTW solvers is shown in the Vessel column, and finally the Runtime column lists the time needed to solve the VRPTW problem. For VRPy and OR Tools, the same sequence was obtained when running 10 tests for each. With ACO, the best result represents the shortest routes obtained after 20 tests, with 50 ants and 1000 iterations. The randomly selected route is considered identical and is expressed in minutes of travel time for the maintenance vessels. The values correspond to the total travel time of all vessels used in the maintenance schedule. Applying VRPy on a selection of 16 windmills to be maintained, to solve the VRPTW, leads to a relative gain compared to the randomly chosen route of about 44%. To obtain this minimal total travel time, three vessels need to be deployed simultaneously, each following a separate route.

Table II shows that the three solution methods, VRPy, OR Tools, and ACO, lead to an almost equal relative gain compared to a random route time of all vessels involved. This accounts for all configurations, varying from 8 to 40 windmills, and increases gradually as the number of windmills to be maintained grows. Except for the configuration of 8 Windmills, the number of vessels proposed by each method are the same, making comparison easier. The only significant difference between the VRPTW solvers is the calculation time required to obtain an optimized solution. Although the average calculation time for the smallest configuration is almost zero and comparable for all options, it rises very fast - almost exponentially - for the VRPy solution, up to more than 350 seconds for the 40 windmills. The calculation time of the ACO algorithm also increases, but is linear and thus not as distinct as for the VRPy solution method. OR Tools finally results in a set of optimized routes instantaneously, even for the set of 40 windmills.

TABLE II. Optimization results for all VRPTW solution methods for different WM Configurations

| Method | Rel gain (%) | Vessels | Run-time (sec) |
|---|---|---|---|
| 8 Windmills | | | |
| VRPy | 11.8% | 3 | 0.33 |
| OR Tools | 16.6% | 2 | 0.03 |
| ACO | 16.6% | 2 | 1.23 |
| 16 Windmills | | | |
| VRPy | 44.1% | 3 | 1.59 |
| OR Tools | 44.1% | 3 | 0.04 |
| ACO | 44.0% | 3 | 2.87 |
| 24 Windmills | | | |
| VRPy | 68.3% | 3 | 10.02 |
| OR Tools | 68.4% | 3 | 0.05 |
| ACO | 68.3% | 3 | 4.52 |
| 32 Windmills | | | |
| VRPy | 70.2% | 3 | 70.61 |
| OR Tools | 70.3% | 3 | 0.12 |
| ACO | 70.2% | 3 | 7.67 |
| 40 Windmills | | | |
| VRPy | 77.3% | 3 | 351.39 |
| OR Tools | 77.3% | 3 | 0.09 |
| ACO | 76.3% | 3 | 19.71 |
| 175 Windmills | | | |
| VRPy | 91.4% | 3 | >24h |
| **OR Tools** | **91.7%** | **3** | **3.51** |
| ACO | 81.2% | 9 | 507.50 |

Table II further contains the results obtained using the three VRPTW solution procedures for a large set of 175 windmills. For this sample, there is a (very) high relative gain for all three solvers, but also a significant difference between the yields obtained by VRPy and OR Tools and that by ACO. Although not negligible, the calculation time for OR Tools is only 3.5 seconds, while ACO now requires more than 8 minutes to obtain a much worse result for a larger number of vessels. VRPy takes an extremely long time to get to a set of optimized routes.

### *B. Sequence comparison VRPy - OR Tools - ACO for customer interventions*

The same solution methods were applied to another data set. This set contains the coordinates of customers of a company that performs interventions on site. These clients are distributed throughout Belgium and are chosen at random from the company's database. The main difference with the windmill configuration is the way the locations are spread: while the windmills are grouped in three so-called parks, the customers are scattered throughout the Belgian territory.

TABLE III. OPTIMIZATION RESULTS FOR ALL VRPTW SOLUTION METHODS FOR DIFFERENT CUSTOMER CONFIGURATIONS

| Method | Rel gain (%) | Vessels | Runtime (sec) |
|---|---|---|---|
| | 8 Customers | | |
| VRPy | 29.1% | 3 | 0.45 |
| OR Tools | 28.7% | 3 | 0.03 |
| ACO | 28.8% | 3 | 2.04 |
| | 16 Customers | | |
| VRPy | 52.5% | 4 | 1.70 |
| OR Tools | 52.9% | 3 | 0.03 |
| ACO | 52.1% | 4 | 2.98 |
| | 24 Customers | | |
| VRPy | 65.0% | 3 | 11.53 |
| OR Tools | 59.9% | 3 | 0.04 |
| ACO | 63.4% | 4 | 6.64 |
| | 32 Customers | | |
| VRPy | 64.8% | 3 | 47.93 |
| OR Tools | 64.8% | 3 | 0.09 |
| ACO | 62.8% | 4 | 7.89 |
| | 40 Customers | | |
| VRPy | 70.5% | 4 | 70.09 |
| **OR Tools** | **70.3%** | **4** | **0.12** |
| ACO | 64.1% | 4 | 11.08 |

Table III shows that the relative gain obtained by the OR solver, VRPy and ACO is again increasing as the number of customers to be served grows. Also, the conclusions about the calculation times are similar to those made for the windmill case: very limited for OR Tools, being almost instantaneously; slightly increasing for the ACO algorithm, ranging from 2 seconds for 8 customers up to 11 seconds for 40 customers and evolving in a more or less linear way; and finally more largely increasing for VRPy, from less than 1 second for 8 to over 70 seconds for 40 customers, following a more exponential curve. However, there are some important differences. First, there are slightly larger gaps between the relative gain obtained for every solution method, while for the windmill case, the results are nearly equal. This is probably due to the fact that there is a clustering around the different farms, making it easier for each method to get stuck in local minima much faster in the previously discussed windmill case. In the taylormade data set, there is no clustering, and thus this phenomenon does not arise. Second, the optimal number of vehicles is not always equal for each solution, making the comparison more difficult.

### C. Job Shop - Workers and windmill maintenance combined

Table IV shows the results of the tests with a different number of windmills, divided over 3 separate farms, ranging from 8 to 40 assets. If all maintenance jobs would be carried out consecutively by one worker without waiting time - being the worst-case scenario for the total maintenance time span - the total time span for all jobs would be 18h for 8 windmills and 120h for 40 windmills. However, if we optimize the schedule for more workers, the total time span would be much lower, being 11 hours for 2 workers in the 8 windmills configuration and 45h for 6 workers in the 40 windmills configuration. This corresponds to a relative gain in maintenance time of respectively around 39% and 62% in the total maintenance time span with respect to the single worker case. By employing more workers simultaneously, the total maintenance time lost is (more than) halved, and therefore downtime is reduced by (more than) 50%. Although the total number of working hours is higher when using three workers instead of one, the amount of money gained by halving the downtime is significantly higher, hence the huge advantage of the JSSP solver. According to the average price per kWh in December 2022, the loss per windmill for 1h downtime is at least 203€ per hour if we presume that a windmill operates 24h per day, 365 days per year. A reduction of the downtime by 67h (with 3 workers) thus leads to a cost reduction of more than 13.6K Euro per windmill. If we further estimate the average labor cost per worker at 60 euros per hour and compare the total amount of hours worked by three workers (154 hours) with the 120 hours needed for one worker, then the extra costs would be 34 times 60 euros, or 2K euros. The net gain would then be 13.6K minus 2K, and thus 11.6K.

TABLE IV. JSSP OPTIMIZATION BY USING OR TOOLS

| Use Case | Relative gain |
|---|---|
| 08 Windmills - 2 workers | 38,9% |
| 16 Windmills - 3 workers | 48,7% |
| 24 Windmills - 4 workers | 53,8% |
| 32 Windmills - 5 workers | 59,6% |
| 40 Windmills - 6 workers | 62,5% |

Table V shows that in the 40 windmill configuration, the largest downtime gain is obtained when switching from one to two workers (44%) and a much lower but significant gain when switching to three workers. From 4 workers onward, the total maintenance time span does not decrease very much when adding additional workers. The trade-off can thus be put at 4 workers or, when labor is expensive, at 3 workers. Remark that when using as many workers as there are tasks to perform, the relative gain is obtained by dividing the longest task by the total time for all tasks, and thus results in a very high optimization (95% in our case).

TABLE V. JSSP OPTIMIZATION IN FUNCTION OF THE NUMBER OF WORKERS FOR 40 WM AND 80 WM

| Number of workers | Rel gain 40WM | Rel gain 80WM |
|---|---|---|
| 2 | 44.2% | 45.7% |
| 3 | 55.8% | 56.0% |
| 4 | 60.0% | 59.0% |
| 5 | 60.8% | 61.2% |
| 6 | 62.5% | 62.5% |

Table V also shows similar results for a configuration of 80 windmills. A large reduction in total maintenance time when a second worker is added, with a relative trade-off at 4 workers. The same results can be extrapolated to the use case of Taylormade products, since results are based on randomly chosen maintenance times, and the location of windmills and customers does not influence the final results of the JSSP.

### *D. Combined results and Comparison*

In order to determine the link between the optimal route resulting from solving VRPTW and the routes determined by solving the JSSP to minimize the time span of all maintenance jobs, a Pareto front is calculated. To compute this Pareto front with non-dominated solutions, tests were run on the two separate problems, and each result was then offered to the other problem. To clarify this, the following example is described: the ACO algorithm, VRPy and OR Tools solution methods ran to solve the VRPTW resulted in maintenance sequences with minimal total traveling distance. For this windmill sequence, the corresponding total time span for all maintenance tasks is calculated by adding the maintenance time for all jobs in this sequence. On the other hand, the total distances are computed for the sequences resulting from solving the JSSP (with a minimal time span). This is done for sequences of 40 windmills and 3 vessels. Figure 4 shows the Pareto front.

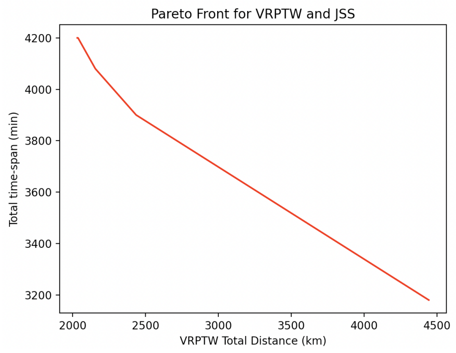


Figure 4. Pareto front for windmills maintenance planning, optimizing both routing (VRPTW) and sequence (JSSP) simultaneously.

Our research has led to a group of Pareto optimal maintenance sequences as a result of the multi-objective optimization model (see coordinates in Table VI). However, it has proven to be very difficult to find a maintenance sequence that is optimal for both objectives. For example, the first and second jobs to be carried out initially according to minimize the total time span can be far away from each other, resulting in a total distance higher than the one obtained by solving the VRPTW. The tests resulted in maintenance paths that either have a low total time span and a high distance, or have a low distance but a high maintenance time span. In determining the optimal sequence, the planner has to decide which parameter is most important when making the choice. From all studied maintenance sequences, a list of 4 non-dominated solutions is obtained. Three of the solutions offer a path with a lower distance and a higher maintenance time span, one is showing a large distance and a lower time span. None of the tests resulted in a path with low values for both objectives.

To compare the financial gain obtained by applying VRPTW and JSSP, we consider the case of 40 windmills and 3 workers.

TABLE VI. PARETO POINTS FOR WINDMILLS MAINTENANCE PLANNING, OPTIMIZING BOTH ROUTING (VRPTW) AND SEQUENCE (JSSP) SIMULTANEOUSLY

| Coordinates | Distance VRPTW (min) | Distance JSSP (min) |
|---|---|---|
| 1 | 2156 | 4080 |
| 2 | 2436 | 3900 |
| 3 | 2037 | 4200 |
| 4 | 4444 | 3180 |

- When reducing the distance from 8380km to 2064km with VRPTW, the financial gain is around 6180€. Fuel consumption is calculated as the distance traveled, multiplied by the weight of the vessel (30 tons), divided by 1000. The speed of the vessel is set at 5.5 km per hour, the fuel price is 2€ per liter, and no wind or current is taken into account.
- When setting the cost per hour downtime at 203€ and, from Table V, the reduction of the downtime at 67h, the total financial gain is 13600€.

The JSSP method thus leads to a much greater benefit than the distance reduction of the VRPTW solution.

## VI. Discussion | Evaluation

Of all methods tested to solve VRPTW, the OR Tools solver offers the quickest solutions, while VRPy and ACO generate similar results but much slower. Both use cases - windmill maintenance and product installation - show similar results with respect to the outcome of the solution method used and the calculation time needed. Also, for the JSSP, the OR Tools solver has proven to be fast and accurate. Comparing the solutions for both objective functions, being distance minimization and maintenance time span optimization, led to sequences that are only optimal for one of the two objectives. Therefore, Pareto points are calculated to obtain solutions that are as optimal as possible for both objectives. The planner can then use these resulting sequences to schedule maintenance tasks for a windmill park to minimize the distance traveled, downtime, or both. In all cases, this leads to a significant reduction in maintenance costs by reducing the fuel used or the loss of energy production. However, several constraints were not taken into account when solving VRPTW, such as sea currents, wind, and vessel capacity. These can be integrated in future work to determine the impact they could have on the final results.

## VII. Conclusion and Future Work

The relative gain obtained is 77% for a set of 40 windmills and 17% for a group of 8 windmills spread over 3 farms, and this for all VRPTW solution methods used. Similar results were found and the same conclusions can be drawn for the second use case, the installation and maintenance of discrete products, showing the general applicability of all methods used to solve the VRPTW. A relative gain of the total maintenance span of almost 62.5% was obtained compared to the situation where all maintenance was done by one worker for a configuration with 40 windmills and 39% for 8 windmills when solving the JSSP. The total time needed for every added worker resulted in a higher total number of working hours to be paid. However, the total maintenance time span was more than halved, resulting in a significant gain in up-time.

In future research, other methods for solving VRPTW and JSSP can be studied and benchmarked. Possible other modi operandi to solve the VRPTW are (nonexhaustive): Harmony Search Algorithms (HAS), Memetic algorithms (MA), Genetic Algorithms (GA), the Hexaly solver, etc. For calculating the JSSP, heuristics or metaheuristics - such as Simulated Annealing, Tabu Search, ACO and Genetic Algorithms - can be compared. In addition, extended and different data sets can be investigated to further determine the applicability of the methods discussed.

## Acknowledgement

This work was carried out within the OPAL project, Offshore Predictable Accessibility by Learning from data and experts with reference HBC.2020.3227, and funded by the Flemish government.

## References

[1] E. De Kuyffer, W. Joseph, L. Martens, and T. De Pessemier, “Vessel route planning optimization combined with time windows versus worker scheduling for offshore windmill maintenancehe vehicle routing problem”, *INTELLI2025: The Fourteenth International Conference on Intelligent Systems and Applications*, 2025.

[2] The Crown Estate and the Offshore Renewable Energy Catapult, “Guide to an offshore wind farm - updated and extended”, *https://guidetoanoffshorewindfarm.com/wind-farm-costs*, 2025.

[3] International Renewable Energy Agency, “Renewable energy technologies: Cost analysis series”, [Volume 1 - Power Sector], 2012.

[4] X. T. Castellà, “Operations and maintenance costs for offshore wind farm”, National Taiwan University of Science and Technology, 2020.

[5] J. Cai, Y. Liu, and T. Zhang, “Preventive maintenance routing and scheduling for offshore wind farms based on multi-objective optimization”, 2023, pp. 1–6.

[6] Z. Ren, A. S. Verma, Y. Li, J. J. E. Teuwen, and Z. Jiang, “Offshore wind turbine operations and maintenance: A state-of-the-art review”, *Renewable and Sustainable Energy Reviews*, vol. 144, 2021.

[7] M. Khan, A. Ahmad, F. Sobieczky, M. Pichler, B. A. Moser, and I. Bukovsky, “A systematic mapping study of predictive maintenance in smes”, *IEEE Access*, vol. 10, pp. 88 738–88 749, 2022.

[8] Y. Merizalde, L. Hernández-Callejo, O. Duque-Perez, and V. Alonso-Gómez, “Maintenance models applied to wind turbines. a comprehensive overview”, *Energies*, vol. 12, 2019.

[9] D. Fan, Y. Ren, Q. Feng, B. Zhu, Y. Liu, and Z. Wang, “A hybrid heuristic optimization of maintenance routing and scheduling for offshore wind farms”, *Journal of Loss Prevention in the Process Industries*, vol. 62, 2019.

[10] R. Dawid, D. McMillan, and M. Revie, “Decision support tool for offshore wind farm vessel routing under uncertainty”, *Energies*, vol. 11, 2018.

[11] D. Juliandri, H. Mawengkang, and F. Bu'ulolo, “Discrete optimization model for vehicle routing problem with scheduling side constraints”, *IOP Conference Series: Materials Science and Engineering*, vol. 300, 2018.

[12] Z. Y. Zhang, “Multi-aco application in routing and scheduling optimization of maintenance fleet (rsomf) based on conditions for offshore wind farms”, *Journal of Power and Energy Engineering*, vol. 6, pp. 20–40, 2018.

[13] F. Meng, Y. Ding, W. Li, and R. Guo, “Customer-oriented vehicle routing problem with environment consideration: Two-phase optimization approach and heuristic solution”, *Mathematical Problems in Engineering*, 2019.

[14] CFI Team, “Value-added resellers (var)”, *https://corporatefinanceinstitute.com/resources/management/value-added-resellers-var/*, 2025.

[15] P. C. Pop, O. Cosma, C. Sabo, and C. P. Sitar, “A comprehensive survey on the generalized traveling salesman problem”, *European Journal of Operational Research*, vol. 314, pp. 819–835, 2024.

[16] P. Sahai, R. Kumar, S. Kumar, S. Shastri, and S. Sharma, “Analysing ai driven bat algorithm to solve the traveling salesman problem”, *Communications in Computer and Information Science*, vol. 2699, 2025.

[17] X. Liu, Y.-L. Chen, L. Y. Por, C. S. Ku, “A systematic literature review of vehicle routing problems with time windows”, *Sustainability*, vol. 15, 2023.

[18] F. Liu, C. Lu, L. Gui, Q. Zhang, X. Tong, and M. Yuan, “Heuristics for vehicle routing problem: A survey and recent advances”, *arXiv*, 2023.

[19] P. T. Anh, C. T. Cuong, and P. N. K. Phuc, “The vehicle routing problem with time windows: A case study of fresh food distribution center”, *11th International Conference on Knowledge and Systems Engineering (KSE)*, pp. 1–5, 2019.

[20] C. M. M. Frey, A. Jungwirth, M. Frey, and R. Kolisch, “The vehicle routing problem with time windows and flexible delivery locations”, *European Journal of Operational Research*, vol. 308, pp. 1142–1159, 2023.

[21] R. Goel and R. Maini, “Vehicle routing problem and its solution methodologies: A survey”, *Int. J. Logistics Systems and Management*, vol. 28, pp. 419–435, 2017.

[22] Z. Injac and D. Draskovic, “Classification of vehicle routing problem”, *JTTTP - Journal of Traffic and Transport Theory and Practice*, vol. 9, pp. 36–41, 2024.

[23] D. G. N. D. Jayarathna, G. H. J. Lanel, and Z. A. M. S. Juman, “Industrial vehicle routing problem: A case study”, *Journal of Shipping and Trade*, vol. 7, 2022.

[24] F. Arnold, M. Gendreau, and K. Sörensen, “Efficiently solving very large-scale routing problems”, *Computers and Operations Research*, vol. 107, pp. 32–42, 2019.

[25] C. A. Irawan, D. Ouelhadj, D. Jones, M. Stålhane, and I. B. Sperstad, “Optimisation of maintenance routing and scheduling for offshore wind farms”, *European Journal of Operational Research*, vol. 256, pp. 76–89, 2017.

[26] M. Stalhane, L. M. Hvattum, and V. Skaar, “Optimization of routing and scheduling of vessels to perform maintenance at offshore wind farms”, *Energy Procedia*, vol. 80, pp. 92–99, 2015.

[27] A. Al-Shayea, E. Fararah, E. A. Nasr, and H. A. Mahmoud, “Model for integrating production scheduling and maintenance planning of flow shop production system”, *Procedia Engineering*, vol. 8, pp. 208 826–208 835, 2020.

[28] X. Han, Y. Han, Y. Liu, Q. Pan, H. Qin, and J. Li, “An improved iterated greedy algorithm for the distributed flow shop scheduling problem with sequence-dependent setup times”, *11th International Conference on Information Science and Technology (ICIST)*, pp. 323–340, 2021.

[29] K. Bülbül and P. Kaminsky, “A linear programming-based method for job shop scheduling”, *Journal of Scheduling*, vol. 16, pp. 161–183, 2013.

[30] Y. Yu, “A research review on job shop scheduling problem”, *E3S Web of Conferences*, vol. 253, 2021.

[31] C. Truden, K. Maier, and P. Armbrust, “Decomposition of the vehicle routing problem with time windows on the time dimension”, *Transportation Research Procedia*, vol. 62, pp. 131–138, 2022.
[32] C. Truden, K. Maier, and P. Armbrust, “Decomposition of the vehicle routing problem with time windows on the time dimension”, *Transportation Research Procedia*, vol. 62, pp. 131–138, 2022.
[33] Q. A. Pham, M. H. Hà, D. M. Vu, and H. H. Nguyen, “A hybrid genetic algorithm for the vehicle routing problem with roaming delivery locations”, *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 32, pp. 297–306, 2022.
[34] V. Sivaram Kumar, M. R. Thansekhar, R. Saravanan, and S. Miruna Joe Amali, “Solving multi-objective vehicle routing problem with time windows by faga”, *Procedia Engineering*, vol. 97, pp. 2176–2185, 2014.
[35] R. Montagné and D. Torres Sanchez, “Vrpy documentation”, *https://vrpy.readthedocs.io/en/latest/*, 2025.
[36] Google OR-Tools, “Vehicle routing problem with time windows”, *https://developers.google.com/optimization/routing/vrptw*, 2025.
[37] W. Othman, A. Wahab, S. Alhady, and H. Wong, “Solving vehicle routing problem using ant colony optimisation (aco) algorithm”, *International Journal of Research and Engineering*, 2018.
[38] S. Li, Y. Wei, X. Liu, H. Zhu, and Z. Yu, “A new fast ant colony optimization algorithm: The saltatory evolution ant colony optimization algorithm”, *Mathematics*, vol. 10, 2022.
[39] Google OR-Tools, “The job shop problem”, *https://developers.google.com/optimization/scheduling/job_shop*, 2025.

# Relevance-Aware Semantic Communication for Intelligent Collective Perception System

Romain Tessier[1,2], Oyunchimeg Shagdar[2], Bruno Monsuez[1]

[1]*U2IS, ENSTA, Institut Polytechnique de Paris*, 828 Bd. des Maréchaux, Palaiseau, France
[2]*Ampere Software Technology*, 1 Av. du Golf, Guyancourt, France
Email: {romain.tessier, bruno.monsuez}@ensta.fr, oyunchimeg.shagdar@ampere.cars

*Abstract*—**Autonomous vehicles face critical limitations when navigating dynamic environments where occlusions or sensor range constraints prevent full situational awareness. Cooperative Intelligent Transport Systems (C-ITS) offer a solution by enabling vehicles to share perception data. However, the uncontrolled volume of exchanged messages leads to congestion and interpretation challenges. This paper proposes a context-aware approach to collaborative perception that transmits only semantically relevant information. By leveraging ontologies to build a knowledge graph of the driving scene, vehicles can reason over their environment, identify safety-critical events, and generate Semantic Collective Perception Messages (S-CPMs). These messages encode not just raw data, but meaningful, situationally prioritized insights, improving decision-making and communication efficiency. A hidden pedestrian use case demonstrates the framework's ability to anticipate and communicate high-risk interactions even in the absence of direct visibility. This semantic approach lays the groundwork for intelligent V2X systems that communicate with precision, relevance, and safety in mind.**

*Keywords-Collaborative Perception; V2X; Ontology; Context-aware; Semantic-Communication.*

## I. Introduction

This work advances our prior research on semantic and context-aware collaborative perception by integrating dynamic relevance estimation mechanisms and a refined ontology-based message generation process [1].

As the global number of vehicles on the road continues to rise, ensuring road safety remains a critical concern. According to the World Health Organization [2], approximately 1.2 million people died in 2023 due to road traffic crashes, with countless more suffering non-fatal injuries. In response to these alarming statistics, the automotive industry faces mounting pressure to improve vehicle safety systems aimed at preventing accidents and reducing fatalities. Automated driving technologies play a key role in this effort by enabling real-time perception, analysis, and response to complex driving environments. Despite these advancements, automated vehicles still face limitations when making decisions based on their own perception of the environment, particularly in scenarios where obstacles obstruct a vehicle's line of sight or where objects are out of sensor range [3][4]. To address these limitations, C-ITS have emerged as a promising solution [5]. By facilitating real-time information exchange among vehicles, infrastructure, and other road users, C-ITS enhances situational awareness beyond the capabilities of onboard sensors alone. Leveraging Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication, C-ITS enables vehicles to access a broader array of information from nearby vehicles or RoadSide Units (RSUs), allowing them to make more informed decisions in critical situations. By sharing data on traffic conditions, potential hazards, and road infrastructure, C-ITS offers a proactive approach to accident prevention that goes beyond the limitations of non connected autonomous systems.

Integrating Collective Perception Services (CPS) within the C-ITS framework represents a crucial step toward achieving safer and more efficient roadways [6][7]. CPS allows vehicles to collaboratively perceive and interpret road users, significantly improving their global perception. The Collective Perception Message (CPM) is the standardized message format used to transmit aggregated data which contain information relative to the locally-detected elements. Particularly valuable is the ability to share data about occluded or out of sensor range objects in real time, which enhances a vehicle's capacity to anticipate and respond to hidden dangers. However, as the number of connected nodes—such as vehicles and infrastructure—continues to grow, so does the volume of data transmitted over communication channels. Given that each CPM usually includes data on the perceived elements, this exponential increase in data can lead to communication congestion, resulting in latency, energy over-consumption, and complexities in merging data across heterogeneous sources.

In the context of vehicular networks, effective communication relies not only on the volume of transmitted data but also on its contextual relevance to the receiver. As conceptualized by Shannon's Information Theory, information corresponds to the reduction of uncertainty, or entropy [8][9]. Accordingly, relevant information is that which significantly decreases the receiver's uncertainty about the driving environment. In collaborative perception, this means that transmitted data should be selected based on its potential to support timely and accurate decisions by downstream systems. However, the relevance of a given piece of information is not absolute—it depends on the receiver's context and decision-making process. For example, an Automatic Emergency Braking (AEB) system requires highly precise, short-range predictions to initiate immediate safety actions, whereas an Autonomous Driving (AD) module benefits from broader, long-term situational awareness, such as anticipating pedestrian intent. In both cases, not all sensed or shared data contributes equally to system performance. To address this, we adopt an ontology-based approach that enables vehicles to formally represent and reason over their observed environment. This structured representation supports the identification of safety-critical situations and the prioritization of messages accordingly. By doing so, the system can dynamically adapt the frequency and content of CPMs, ensuring that only semantically relevant and high-impact information is communicated.

This paper is organized as follows. Section II reviews the current state of the art in congestion control and semantic communication for vehicular networks. Section III introduces a formal problem formulation that defines the collaborative perception setting, including the structure of observations, transmissions, and information relevance. Section IV presents the ontological framework used to represent and reason about the driving scene. Section V describes how contextual relevance is estimated through semantic reasoning over a knowledge graph. Section VI details the construction and transmission of semantically enriched CPMs within the C-ITS communication stack. Section VII discusses the limitations and implementation challenges of the proposed framework, including issues related to ontology standardization, real-time reasoning, and integration into ADAS pipelines. Finally, Section VIII concludes the paper and outlines potential directions for future work.

## II. Related Work

Mitigating channel congestion has been the main concern in a large number of research activities. For example, in [10], vehicles reduce the CPM generation frequency in high-density areas. Decentralized Congestion Control (DCC) techniques have been proposed to allow individual nodes to autonomously adjust their transmission rates based on channel congestion level observed locally [11–14]. While these congestion control systems effectively alleviate network congestion, they often lack explicit consideration of context. In critical scenarios, this can lead to potentially harmful information gaps. To address this, some solutions incorporate context-awareness. For example, [15] proposes limiting collaborative communication to the most relevant nodes by creating a matching score between nodes. However, in C-ITS, where actors change rapidly, this approach is incompatible with the handshake mechanism explained in Who2Com [15]. Consequently, other studies propose limiting communication within geographical zones to ensure a level of relevance. In Direct-CP [16], collaborative communication is monitored by infrastructure based on each vehicle's maneuver intent. In contrast, Where2Com [17] does not rely on infrastructure to manage communication; instead, it uses a spatial confidence map at each agent to facilitate pragmatic compression, guiding agents on what to communicate, with whom, and whose information to aggregate. Additionally, [18] introduces a protocol that takes context into account for CPM generation frequency by aggregating information about the communication channel and environmental context (e.g., other vehicles and road layout). However, these solutions do not ensure that transmitted messages remain semantically relevant to the receiver; in other words, they do not consider what information will be efficiently consumed. Consequently, the receiver must infer semantic information about the sender's context, which may lead to interpretation issues.

To tackle these challenges, recent studies advocate for semantic communication between vehicles, which aims to convey meaningful content with inherent contextual value. For instance in [19], the authors implemented collaborative perception by extracting semantic features that are gathered and computed by an edge server. This concept of communicating high semantic-value information is also explored in [20–23] where a semantic encoder/decoder achieves higher transmission efficiency. This approach is demonstrated in [24] for image segmentation: rather than sending a full image (6 MB), it can be advantageous to transmit only the semantic interpretation of the image (30.5 KB). However, in semantic communication, the data is not merely compressed; it is reduced to the essential meaning. Thus, both the sender and receiver must have some form of shared knowledge to encode and decode the information effectively. This notion of a knowledge base can be linked to situational context, as the context forms part of the vehicle's knowledge. Finally, [25] provides initial steps for implementing semantic communication in V2X, introducing a new layer between the application layer and the transport/network layer. The authors illustrate the benefits of semantic communication through use cases such as adaptive traffic light management and collaborative driving. In this work, we aim to advance these efforts by (i) enhancing context-awareness in collaborative perception to generate situationally relevant messages, and (ii) adding semantic precision to collaborative messages, thereby minimizing interpretation issues and improving decision-making capabilities.

## III. Problem Formulation

In a collaborative perception setting, vehicles both *observe* and *receive* overlapping targets information, fusing local sensor readings with messages from peers to enhance situational awareness. The system under study comprises $\mathcal{M} = \{1, \dots, M\}$ road users, of which $N \leq M$ are connected vehicles, forming the set $\mathcal{C} = \{1, \dots, N\}$. Each connected vehicle $i \in \mathcal{C}$ observes each target $k$ (from the set $\mathcal{M}$) through a state vector

$$\hat{\mathbf{x}}_{i,k}(t) = \begin{bmatrix} p_k(t) \\ v_k(t) \\ \theta_k(t) \\ a_k(t) \end{bmatrix},$$

where $p$, $v$, $\theta$ and $a$ denote position, velocity, heading and acceleration, respectively. Visibility is captured by

$$b_{i,k}(t) = \begin{cases} 1, & \text{if } i \text{ senses target } k \text{ at } t, \\ 0, & \text{otherwise,} \end{cases}$$

and each potential transmission from $i$ to $j$ about $k$ is governed by

$$u_{i \to j,k}(t) \in \{0, 1\},$$

We define the *information set* available to vehicle $j \in \mathcal{C}$ about target $k \in \mathcal{M}$ at time $t$ as the union of its own local observation and all received messages from peers. Formally:

$$\mathcal{I}_{j,k}(t) = \underbrace{\{\, b_{j,k}(t)\, \hat{\mathbf{x}}_{j,k}(t) \}}_{\substack{\text{own observation} \\ \text{(if visible)}}} \cup \underbrace{\bigcup_{i=1}^{N} \{\, b_{i,k}(t-\tau)\, \hat{\mathbf{x}}_{i,k}(t-\tau) \mid u_{i \to j,k}(t-\tau) = 1 \}}_{\text{received observations}}. \quad (1)$$

Here:

- $b_{i,k}(t) \in \{0,1\}$ is the visibility indicator of target $k$ to vehicle $i$.
- $\hat{\mathbf{x}}_{i,k}(t)$ is the state vector $[p_k, v_k, \theta_k, a_k]$ observed by $i$ at time $t$.
- $u_{i \to j,k}(t) \in \{0,1\}$ indicates whether $i$ transmits its observation of $k$ to $j$ at time $t$.
- $\tau$ is the delay between the generation of the data by the observer $i$ and its usage by the receiver $j$ at time $t$.

This precisely captures, for each $j, k$, the mixture of locally generated and peer-received information available at any given instant.

This formulation assumes that the core objective of CPS is to provide each connected vehicle with timely and accurate information about all other road users. Under this assumption, the goal is to reduce the ego vehicle's perception uncertainty regarding its surrounding environment, thereby enabling more informed and safer decision-making. Redundant or imprecise transmissions are undesirable, as they consume communication resources without meaningfully enhancing the receiver's awareness. To address this, each transmitter $i \in \mathcal{C}$ optimizes its message generation decision $u_{i \to j,k}(t)$ with respect to a shared communication objective. Rather than broadcasting frequently and independently, transmitters coordinate their transmissions in a distributed and complementary manner to ensure that all targets $k \in \mathcal{M}$ are covered. This strategy promotes an even distribution of the transmission load, with connected vehicles collectively sharing the responsibility of informing their peers. As a result, each receiver $j \in \mathcal{C}$ can maintain a high-frequency, low-uncertainty perception of surrounding targets using only a limited number of CPMs. This maximizes perception accuracy while minimizing communication overhead.

However, not all information has the same value. The impact of shared data varies depending on the external situation and the internal context of the decision-making process that consumes it. In hierarchical decision-making architectures, the relevance of information is evaluated differently at each layer, ranging from high-level route planning to low-level motion control [26–28].

The internal context of the receiving vehicle, such as its current goal, position, maneuver stage, or driving intent, directly influences which pieces of information are considered useful. For example, at an unsignalized intersection, a vehicle approaching the crossing must closely monitor lateral traffic with unclear right-of-way. In such a case, timely and accurate information about cross-traffic is critical, while data about distant vehicles or non-threatening agents may be irrelevant [29]. Consequently, not all shared information contributes equally to decision quality. Transmitting irrelevant or low-impact data not only wastes bandwidth but may also introduce unnecessary computational load. An effective communication strategy must therefore go beyond reducing global uncertainty. It must be context-aware and prioritize the transmission of semantically relevant data tailored to the receiver's situational needs. In practice, this means that the set of useful information is often a small subset of all accessible data. This distinction becomes especially important in safety-critical tasks such as collision avoidance, which demand high levels of precision and low latency. In a collaborative perception framework, the reacting agent is not the observer, but the receiver, whose decisions are subject to communication and processing delays. This delay-sensitive structure amplifies the need to transmit only relevant and actionable information. Sharing data that does not contribute to the receiver's immediate awareness not only wastes resources but may also lead to late or suboptimal decisions. Therefore, early recognition of potentially dangerous situations, before they escalate, is essential. Prioritizing contextually relevant information allows the system to allocate communication resources more effectively, sustaining high safety standards despite delay constraints.

Nevertheless, estimating the potential value of information for other vehicles is inherently challenging. Transmitters typically lack full knowledge of the receiver's internal state, including its goals, plans, or decision criteria. Instead, they must infer relevance from observable contextual cues. The challenge, then, is to design communication policies that prioritize information likely to be beneficial, while filtering out data known to be irrelevant.

One promising direction is to ground communication decisions in established accidentology research, which identifies scenarios where information sharing has demonstrable safety benefits. For instance, the European SECURE project has evaluated the benefits of V2X communication in 15 high-risk driving scenarios [30]. These findings provide a valuable foundation for defining high-impact situations in which transmitting specific information is strongly justified. If a transmitting vehicle can recognize such scenarios in real time, it can dynamically adapt its communication behavior to match the inferred safety requirements of the environment. This enables a context-aware, safety-driven communication policy that prioritizes messages when and where they are most likely to reduce risk. Importantly, this approach does not require modeling the receiver's decision-making process directly. Instead, it justifies information sharing based on the expected safety benefit of transmission, as inferred from the scenario.

By grounding message generation in accidentology results, this approach provides a pragmatic and risk-informed framework for collaborative perception in vehicular networks.

To illustrate this, let us consider the scenario illustrated in

Figure 1. A vehicle (V1) is positioned on the left side of a straight road, while a pedestrian (P1) is crossing, and another vehicle (V2), approaching from the right, is obscured by a bus (O1). This hidden pedestrian situation is particularly critical for accident prevention [30], highlighting the importance of collaborative perception between vehicles.

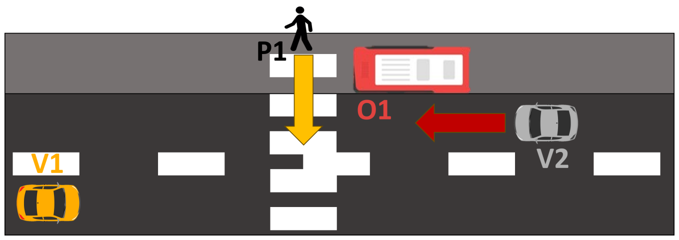


Figure 1. Use Case: Hidden Pedestrian Intending To Cross.

This use case illustrates a fundamental limitation of conventional CPS when operating in safety-critical contexts. Traditional systems, which lack the ability to interpret the situational context, must manage a trade-off between communication frequency and channel load. In the absence of contextual understanding, these systems are unable to determine which road user information should be prioritized. As a result, they often resort to broadcasting all detected objects at a high frequency to ensure that actionable information is delivered promptly.

However, such an approach can be counterproductive. In situations where not all information is relevant, as illustrated by the unnecessary transmission of bus data in this scenario (see Figure 1), the additional communication load increases channel congestion and latency. This, in turn, delays the delivery of critical information, diminishing its impact. Consequently, the receiving vehicle (V2) may be unable to make timely and appropriate decisions, thereby increasing the risk of collision or unsafe behavior.

These limitations emphasize the need for a context-aware and adaptive communication strategy, one that can recognize high-risk situations and dynamically modulate the information-sharing rate based on inferred safety requirements. By tailoring the communication to the situational context, the system can achieve both high efficiency and improved safety performance, avoiding the pitfalls of channel overload or under-communication during critical moments.

A promising solution involves the integration of a formalized and shared knowledge base within communicating vehicles. In this approach, the sender vehicle (V1) can selectively transmit only the most semantically relevant and safety-critical information, such as the detection of a hidden pedestrian, while the receiver (V2), equipped with an aligned knowledge representation, is capable of interpreting the data with greater accuracy and urgency. This knowledge-driven communication paradigm facilitates a more intelligent and context-sensitive collaborative perception framework, ultimately enhancing decision-making capabilities and improving safety in complex traffic environments.

## IV. Understanding the Driving Scene

### A. *Formalizing concepts*

Ontologies—structured models in knowledge representation—enable this level of contextual relevance by defining sets of concepts, their attributes, and relationships within a specific domain [31–34]. Leveraging ontologies enables machines to process and share information with enhanced semantic precision. In autonomous vehicle systems, ontologies provide a standardized framework for consistently interpreting and integrating data across diverse systems—an essential capability for effective inter-vehicular communication and decision-making. Given the variety of data sources in autonomous driving, from real-time sensors to camera feeds, ontological mapping transforms raw data into semantically enriched formats.

To capture the complexity of the driving environment, we used two interlinked ontologies [35]. The first, the *Road Topology Ontology*, formalizes the physical and regulatory structure of the road network. The second, the *Agent Ontology*, models road users, their behaviors, interactions, and visibility conditions. At runtime, a knowledge graph is constructed by instantiating these ontologies using real-time perception data, enabling semantic reasoning for context-aware decision-making.

The complete class and property definitions of both ontologies are provided in the Annex (Table I–Table IV).

### B. *Building the Knowledge Graph*

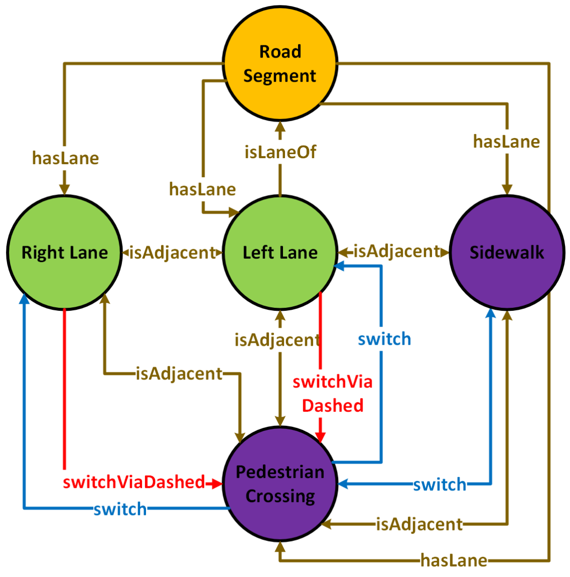


Figure 2. Representation of the road topology for the use case.

The construction of the knowledge graph begins with modeling the *road topology*, which captures the structural layout and regulatory logic of the driving environment. This includes the relationships between `Lanes`, `Intersections`, `PedestrianCrossings`, and turn directions, as well as control elements such as `TrafficLights` and `LaneRestrictions`.

These concepts are formalized through a dedicated *Road Topology Ontology*, which defines not only the static geometric entities but also their topological and regulatory interconnections using semantically meaningful properties such as `hasLane`, `isConnected`, `hasTurnDirection`, and `switchViaIntersection`.

As shown in Figure 2, the resulting topology graph provides a static, machine-interpretable representation of the road network, where each `RoadSegment`, `Lane`, and `PedestrianCrossing` is instantiated and connected according to the real-world configuration. This component may be precomputed and retrieved from a high-definition map or a dedicated infrastructure knowledge base. The ontology also supports rule-based reasoning, such as inferring priority relationships at intersections or determining occlusion risks based on spatial adjacency and lane layout.

The second step involves populating the graph in *real time* with dynamic information about road users and their observed behaviors. These agents—such as vehicles, pedestrians, and cyclists—are integrated into the knowledge graph using a separate *Agent Ontology*, which models classes of road users, their actions (e.g., `Accelerating`, `Stopping`, `Walking`), their positions, and their interactions. This dynamic content is derived from onboard sensors as well as received CPMs and is expressed as semantic triples, enabling structured querying and logical inference.

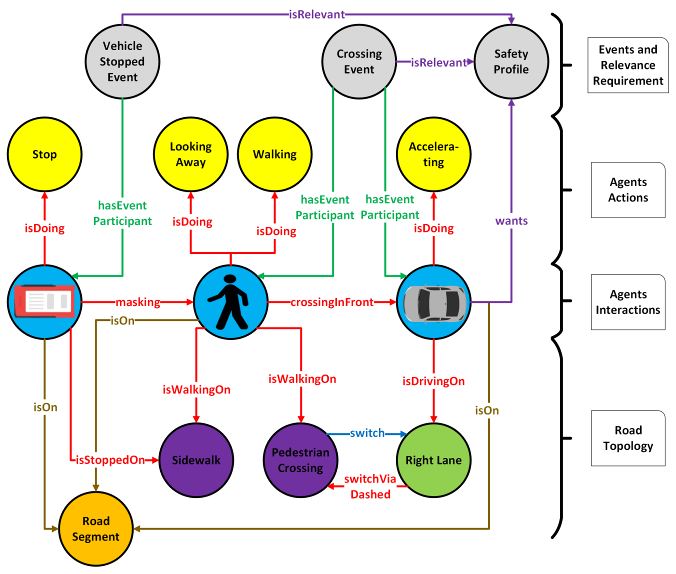


Figure 3. Example of a full Knowledge Graph for the use case, including topology and agent interactions.

Figure 3 illustrates the resulting integrated knowledge graph. Static elements (e.g., lanes, sidewalks, crossings) are semantically linked to dynamic agents (e.g., vehicles driving on specific lanes, pedestrians crossing a road). These links enable high-level reasoning about the scene, including interaction detection, right-of-way analysis, and occlusion inference.

Once the knowledge graph is populated with both the *road topology* and the *agent state and behavior*, the system can assess the relevance of specific observations by detecting contextually significant or safety-critical situations. This process, described in Section V, forms the basis for generating semantically filtered, high-utility CPMs tailored to the needs of receiving agents.

## V. Contextual Relevance Estimation

Relevance identification is achieved by recognizing high-risk interactions between road users. The first step for the transmitting vehicle is to reason over its local knowledge in order to extract safety-critical events.

### A. Extracting High-Risk Interactions

High-risk interactions are detected through a reasoning layer that applies a set of logical rules written in the Semantic Web Rule Language (SWRL). These rules operate over an ontological representation of the driving scene, enabling the system to infer new knowledge from the existing structure of spatial, temporal, and behavioral data. SWRL rules follow a declarative logic-based format, composed of antecedents (conditions) and consequents (inferred facts), all expressed using the vocabulary of the domain ontology — including classes (e.g., `Vehicle`, `Pedestrian`), properties (e.g., `isOn`, `crossingInFront`), and relationships between entities.

This formalism provides a powerful mechanism for modeling and identifying semantically meaningful interactions within a scene. For instance, a rule can infer that a `CrossingEvent` is occurring when a vehicle is maintaining speed while a pedestrian is crossing its path under specific structural and behavioral conditions. Such inferences form the basis for determining whether a given situation constitutes a safety-critical event that should be communicated to interested agents. Scenario-specific rules focus on clearly defined use cases where the benefits of collaborative perception have been observed or demonstrated. These rules capture high-risk interactions such as unprotected left turns across oncoming traffic, merging at blind intersections, or pedestrian crossings obscured by static obstacles — scenarios in which an individual vehicle's perception is likely to be limited and where shared context can meaningfully improve situational awareness [30]. Although highly effective within their intended scope, these rules tend to be less robust when applied to unforeseen scenarios or rare combinations of factors not considered during their formulation. Despite the limitations of scenario-specific rules, they still provide valuable semantic structure for decision-making. Importantly, the presence of a matching rule does not imply that all other cases are irrelevant. In real-world driving, unexpected interactions often emerge from uncommon combinations of seemingly benign factors. Therefore, the system should not rely solely on exact rule matching but instead assess the semantic similarity of a current situation to known risk patterns. This can be achieved by reasoning over the ontological structure or via similarity-based approaches in the embedding space of scene descriptors.

To extend the expressiveness and adaptability of the system, an alternative is to leverage accidentology databases. By analyzing large-scale, annotated records of traffic accidents, machine learning techniques can be used to discover implicit relevance patterns — correlations between agent behavior, environmental context, and collision likelihood [31][32]. These insights can be distilled into either: Parameterized conditions that inform new SWRL rules, or direct rule generation pipelines, where learned decision trees or classifiers are translated into rule sets. Such rules are not only more specific but also adaptive, allowing the system to evolve over time as more accident data becomes available. The training process serves as a bridge between raw statistical correlations and structured, interpretable knowledge. Once trained, the vehicle can assess the relevance of a situation in real time by applying the generated SWRL rules to its local representation of the scene — continually updated through its onboard perception stack and shared knowledge modules.

### *B. Example Rule: Crossing Event Detection*

In the pedestrian crossing scenario, we can define a SWRL rule to detect and infer the relevance of such an event, as shown below:

Crossing Event Detection Rule

```
IF:
  Vehicle(?lowPriority) ^
  RoadUser(?highPriority) ^
  RoadSegment(?road) ^
  isOn(?lowPriority, ?road) ^
  isOn(?highPriority, ?road) ^
  Lane(?lowPriorityLane) ^
  isDrivingOn(?lowPriority, ?lowPriorityLane) ^
  Lane(?highPriorityLane) ^
  (isDrivingOn(?highPriority, ?highPriorityLane)
   OR isWalkingOn(?highPriority, ?
       highPriorityLane)) ^
  crossingInFront(?lowPriority, ?highPriority) ^
  (switchViaDashed(?lowPriorityLane, ?
      highPriorityLane)
   OR (switchViaTrafficLight(?lowPriorityLane, ?
       highPriorityLane) ^
      hasTrafficLight(?lowPriorityLane, ?
          trafficLight) ^
      hasTrafficSignalPhase(?trafficLight, ?
          phase) ^
      sameAs(?phase, Red))
   OR switchViaIntersection(?lowPriorityLane, ?
       highPriorityLane)) ^
  isDoing(?lowPriority, ?action) ^
  (sameAs(?action, Accelerating)
   OR sameAs(?action, MaintainingSpeed))

THEN:
  CrossingEvent(?crossing) ^
  hasEventParticipant(?crossing, ?lowPriority) ^
  hasEventParticipant(?crossing, ?highPriority)
```

This SWRL rule identifies a crossing event involving a low-priority vehicle and a higher-priority road user (such as a pedestrian or another vehicle) when specific spatial and behavioral conditions are met within a driving scene. The rule applies when both agents are present on the same road segment, each traveling on a distinct lane. The high-priority user is either driving or walking on their lane, while the low-priority vehicle is in a situation where the two lanes are linked—meaning a lane change or crossing is structurally possible—either via a dashed line, a traffic light (which is currently red for the low-priority vehicle), or an intersection. Furthermore, the higher-priority user is observed to be crossing in front of the vehicle, indicating a potential interaction. Despite this, the vehicle is accelerating or maintaining its speed, which contrasts with the expected behavior in such a scenario, where the vehicle should slow down due to its lack of priority. Given these conditions, the rule infers the existence of a `CrossingEvent`, linking both agents as participants, and potentially signaling a conflict or risk that needs to be addressed in downstream reasoning or decision-making processes.

### *C. Determining Receiver Relevance*

Once a safety-critical event has been inferred, the next step is to assess whether any connected agents in the vicinity should be informed. Relevance is not solely determined by the severity of the event itself, but also by the contextual usefulness of the information to a potential receiver. This dual perspective — sender-side significance and receiver-side utility — enables efficient and targeted communication in collaborative perception systems.

Receiver relevance is evaluated through reasoning and querying mechanisms over the shared semantic knowledge base. A SPARQL query can be issued to identify nearby connected agents that satisfy two key conditions:

1) **Capability:** The agent must be technically able to receive and interpret the message, i.e., it is a connected agent with the adequate support.
2) **Contextual Awareness:** The agent must be in a situation where the received information could affect its decision-making process or enhance its situational awareness.

In practice, this assessment involves evaluating spatial, temporal, and behavioral factors for each agent in the vicinity. For example, a *pedestrian crossing* event is highly relevant to a vehicle approaching the crossing from the same or an intersecting road segment, as it may need to slow down or stop. However, it is largely irrelevant to a vehicle moving away from the area or traveling on a disconnected or parallel segment.

To support this reasoning, all participants involved in an event are explicitly linked to it using the `hasEventParticipant` property. This semantic relationship ensures that the presence of at least one connected participant in a critical interaction can trigger message generation. In the pedestrian crossing scenario, for instance, if a connected vehicle is involved (as the lower-priority participant), the system infers that information about the other participant (e.g., a pedestrian) should be included and prioritized in the transmitted message.

Conversely, in a the use case, the bus (O1) parked on the sidewalk may trigger a `StoppedVehicle` event. However, this event is only considered relevant to vehicles driving on the same lane, who might need to change lanes, slow down, or adapt their trajectory. Even if a connected vehicle (e.g., V2) is present in the scene, it is located on a different lane that making the information about the bus under-prioritized. This approach avoids unnecessary communication overhead and ensures that bandwidth is preserved for information with immediate operational value.

## VI. Knowledge Sharing

Knowledge sharing between vehicles complements local sensor data by providing additional context, which is essential for autonomous decision-making. Studies have shown that ontologies and formalized knowledge representations significantly enhance the decision-making capabilities of automated systems [33][34][36]. Semantic-aware messages allow vehicles to exchange not only raw data but also high-level, structured information about their environment [19–21][24][25].

To enable the sharing of semantically relevant content, the CPM format can be extended to incorporate semantic properties. Unlike conventional CPMs that transmit raw object-level data (e.g., positions and velocities), Semantic CPMs (S-CPMs) include structured annotations grounded in a shared ontology. This enables vehicles to encode both the "what" (e.g., a pedestrian at position $X$) and the "why it matters" (e.g., "pedestrian hidden from eastbound traffic, located on sidewalk, and likely to cross").

In the presented use case, vehicle V1 constructs a local knowledge graph and identifies a `CrossingEvent` involving a pedestrian obscured by a bus (O1). The onboard reasoning system determines that vehicle V2, approaching from the opposite direction with no line of sight, would benefit from this information. Consequently, V1 transmits an S-CPM enriched with semantic annotations, such as the masking relationship between the bus and the pedestrian or the pedestrian's location on the sidewalk.

This concise yet semantically rich message allows V2 to interpret the situation even without direct visual contact, enabling faster and more informed reactions.

Figure 4 illustrates how the semantic layer is integrated into the traditional CPM pipeline. On the left, sensor inputs from local perception modules and received CPMs are used to populate a knowledge graph where entities, actions, and relationships are semantically defined (cf. Section IV). A reasoning engine then operates over this graph to infer high-risk events, such as a `CrossingEvent` involving a hidden pedestrian (cf. Section V).

When a safety-relevant situation is inferred, the system evaluates whether any nearby connected agents would benefit from the information. If so, it constructs an S-CPM that includes only the contextually relevant semantic content, as shown on the right side of the diagram. This message is transmitted over the V2X channel, allowing the receiving vehicle to understand the event without reconstructing the entire scene from raw data.

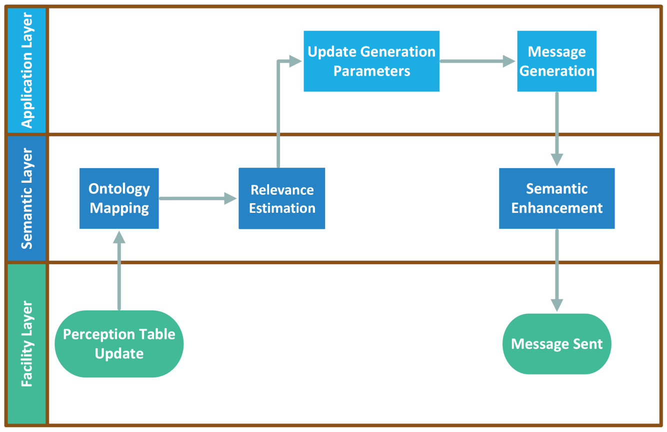


Figure 4. Integration of Semantic Layer for CPM.

This architecture tightly integrates perception, reasoning, and communication by embedding semantic understanding into the message-generation pipeline. It shifts the paradigm from periodic, raw data transmission to the sharing of selectively filtered, semantically prioritized, and meaning-rich content.

## VII. Discussion

The proposed framework introduces a semantically enriched, context-aware communication mechanism for vehicular networks, aiming to transmit only the most relevant information to downstream agents. While the architecture demonstrates promising potential for reducing channel load and enhancing safety, several key issues must be considered for real-world deployment.

### A. Generalization Beyond Rule-Based Reasoning

Our approach currently relies on manually crafted SWRL rules to detect high-risk interactions. While this ensures transparency and interpretability, it limits the system's adaptability to unexpected or complex scenarios not captured by predefined logic. Human-authored rules are also labor-intensive to maintain and susceptible to obsolescence as traffic environments evolve.

To address this, future developments should incorporate data-driven approaches such as decision tree induction or statistical relational learning to automatically derive semantic rules from annotated driving datasets. These methods have been successfully applied in other fields to generate interpretable knowledge bases [14], and could help enhance generalization in dynamic environments.

### B. Real-Time Reasoning and Scalability Constraints

Ontologies provide rich structure but incur computational overhead during reasoning and querying. Inference over large-scale knowledge graphs in real time remains challenging, especially under V2X latency constraints (typically <100 ms per CPM cycle).

To mitigate this, techniques such as incremental reasoning, reasoning over lightweight ontology subsets, or offloading to roadside infrastructure may be required [34][36]. The feasibility of these methods in realistic urban traffic conditions must be quantitatively evaluated through simulation and profiling experiments.

### *C. Knowledge Graph Requirements and Semantic Interoperability*

The effectiveness of the proposed semantic communication framework relies on each agent's ability to access and reason over a structured knowledge graph representing its local environment. However, this requirement introduces significant challenges for real-world implementation. In a collaborative setting, the utility of semantically enriched messages depends on the receiver's capacity to correctly interpret the transmitted content—something that is only feasible if both sender and receiver share not only common ontological definitions but also compatible graph structures.

This dependency presents a barrier to semantic interoperability in practical deployments. Discrepancies in class hierarchies, naming conventions, or modeling assumptions can result in semantic mismatches or the misinterpretation of critical safety messages. For instance, the semantics of a `CrossingEvent` or `Occlusion` may differ in granularity or causal meaning across implementations, even when referring to the same real-world observation.

To address this, recent initiatives have introduced ontology standards to support semantic alignment. ETSI's SAREF4Automotive ontology [37] extends the SAREF (Smart Applications REFerence) framework to describe key automotive concepts such as vehicle status, driving modes, and environmental features in a machine-interpretable way. Likewise, the W3C Semantic Sensor Network (SSN) and its lightweight core, SOSA (Sensor, Observation, Sample, and Actuator) [38], offer an ontology stack for modeling sensors, observations, and actuators.

While these ontologies provide a valuable foundation, they currently lack explicit support for behavior- and event-centric modeling required in collaborative perception tasks, such as interaction between agents, priority inference, or semantic occlusion reasoning. Bridging this gap will require extending or aligning these standards with richer ontologies that capture the spatio-temporal and causal dynamics of road scenes. A long-term solution may involve developing a modular and extensible ontology framework where standardized core concepts are combined with domain-specific modules tailored to V2X semantic communication.

### *D. Integration with Planning and ADAS Systems*

Currently, the semantic layer focuses on perception-level reasoning. However, its benefits extend further downstream. Integrating semantic information into behavior planning or trajectory generation modules can enable more proactive and interpretable decision-making [26][27].

For example, a `CrossingEvent` involving a pedestrian could trigger adaptive speed control or early braking in an ADAS module, even before visual confirmation is available. Future work should explore pipeline integration and quantify decision quality improvements.

### *E. Communication Policy and Network Efficiency*

By prioritizing relevant messages, the framework also acts as a semantic-aware congestion control mechanism. This aligns with recent works in context-aware message generation such as Direct-CP [16] and Where2Com [17], but introduces a more explicit semantic reasoning layer. However, network-level performance metrics such as bandwidth usage, packet delivery ratio, and channel congestion under high vehicle density must be evaluated to confirm expected efficiency gains. Additionally, fallback mechanisms—such as periodic CPMs or redundant messages in safety-critical scenarios—should be designed to ensure robustness under communication loss or partial knowledge graph failures.

### *F. Ethical and Privacy Considerations*

Semantic communication inherently transmits high-level interpretations of behavior and intent. While beneficial for decision-making, this raises privacy and ethical concerns. For instance, sharing a message stating that a pedestrian is likely to cross constitutes behavioral profiling. If transmitted over unsecured channels, this information could be exploited or misused. Ensuring compliance with privacy regulations such as GDPR requires anonymization, and minimization. Moreover, inference confidence scores or uncertainty annotations may help mitigate the impact of incorrect or spurious reasoning.

## VIII. Conclusion

This work introduces a semantic communication framework for collective perception that prioritizes the transmission of contextually relevant information through ontological reasoning. By leveraging structured knowledge graphs and logical inference, the system identifies high-risk interactions and generates semantically enriched CPMs that improve the precision and utility of shared information. Beyond reducing communication overhead, this approach enhances safety in occluded or complex environments by enabling proactive and informed decisions. In future work, relevance estimation will be implemented within a simulation environment, leveraging ontologies to support various consumers, such as Perception, Advanced Driver Assistance Systems (ADAS), and Automated Driving. This effort will involve the development of an ontology-based framework and a comparative analysis of two distinct approaches to defining relevance. The first approach will utilize machine learning algorithms for pattern extraction, employing data-driven techniques to derive relevance rules. The second approach will adopt a scenario-specific exploration, where relevance is defined based on predefined scenarios and expert-driven criteria tailored to specific use cases. By comparing these methods, this study aims to uncover their respective strengths, limitations, and areas of applicability, paving the way for more adaptive and effective relevance estimation strategies across diverse applications. Additionally, comparisons will be made with methodologies presented in recent literature [15][16][17] to benchmark and validate the proposed approaches. It is also crucial to address the challenges posed by ontology computation in real-time scenarios, ensuring its feasibility and robustness in practical implementations.

## References

[1] R. Tessier, O. Shagdar, and B. Monsuez, "Context-Aware Collaborative Perception: Estimating Relevance through Knowledge Representation," *VEHICULAR 2025, The Fourteenth International Conference on Advances in Vehicular Systems, Technologies and Applications*, 2025, ISSN: 2327-2058.

[2] *Global Status Report on Road Safety 2023*, 1st ed. Geneva: World Health Organization, 2023, ISBN: 978-92-4-008651-7.

[3] J. Ruan et al., "A review of occluded objects detection in real complex scenarios for autonomous driving," *Green Energy and Intelligent Transportation*, vol. 2, no. 3, p. 100 092, Jun. 2023, ISSN: 27731537. DOI: 10.1016/j.geits.2023.100092

[4] E. Marti, M. A. De Miguel, F. Garcia, and J. Perez, "A Review of Sensor Technologies for Perception in Automated Driving," *IEEE Intelligent Transportation Systems Magazine*, vol. 11, no. 4, pp. 94–108, 2019, ISSN: 1939-1390, 1941-1197. DOI: 10.1109/MITS.2019.2907630

[5] J. Ko, J. Jang, and C. Oh, "Assessing the Safety Benefits of In-Vehicle Warning Information by Vehicle Interaction Analysis in C-ITS Environments," *Journal of Korean Society of Transportation*, vol. 39, no. 1, pp. 1–13, Feb. 2021, ISSN: 1229-1366, 2234-4217. DOI: 10.7470/jkst.2021.39.1.001

[6] G. Yu, H. Li, Y. Wang, P. Chen, and B. Zhou, "A review on cooperative perception and control supported infrastructure-vehicle system," *Green Energy and Intelligent Transportation*, vol. 1, no. 3, p. 100 023, Dec. 2022, ISSN: 27731537. DOI: 10.1016/j.geits.2022.100023

[7] S. Hu, Z. Fang, Y. Deng, X. Chen, and Y. Fang, *Collaborative Perception for Connected and Autonomous Driving: Challenges, Possible Solutions and Opportunities*, arXiv:2401.01544 [cs, eess], Jan. 2024.

[8] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948. DOI: 10.1002/j.1538-7305.1948.tb01338.x

[9] J. V. Stone, *Information Theory: A Tutorial Introduction*, arXiv:1802.05968 [cs], Jun. 2019.

[10] S. Ishihara, K. Furukawa, and H. Kikuchi, "Congestion Control Algorithms for Collective Perception in Vehicular Networks," *Journal of Information Processing*, vol. 30, pp. 22–29, 2022, ISSN: 1882-6652. DOI: 10.2197/ipsjjip.30.22

[11] A. Mansouri, V. Martinez, and J. Harri, "A First Investigation of Congestion Control for LTE-V2X Mode 4," in *2019 15th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, Wengen, Switzerland: IEEE, Jan. 2019, pp. 56–63, ISBN: 978-3-903176-13-3. DOI: 10.23919/WONS.2019.8795500

[12] A. Bazzi, "Congestion control mechanisms in ieee 802.11p and sidelink c-v2x," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, Nov. 2019, pp. 1125–1130. DOI: 10.1109/IEEECONF44664.2019.9048738

[13] H.-J. Gunther, R. Riebl, L. Wolf, and C. Facchi, "Collective perception and decentralized congestion control in vehicular ad-hoc networks," in *2016 IEEE Vehicular Networking Conference (VNC)*, Columbus, OH, USA: IEEE, Dec. 2016, pp. 1–8, ISBN: 978-1-5090-5197-7. DOI: 10.1109/VNC.2016.7835931

[14] M. H. Jabardi and A. S. Hadi, "Using Machine Learning to Inductively Learn Semantic Rules," *Journal of Physics: Conference Series*, vol. 1804, no. 1, p. 012 099, Feb. 2021, ISSN: 1742-6588, 1742-6596. DOI: 10.1088/1742-6596/1804/1/012099

[15] Y.-C. Liu et al., *Who2com: Collaborative Perception via Learnable Handshake Communication*, arXiv:2003.09575 [cs], Mar. 2020.

[16] Y. Tao, S. Hu, Z. Fang, and Y. Fang, *Direct-CP: Directed Collaborative Perception for Connected and Autonomous Vehicles via Proactive Attention*, arXiv:2409.08840 [cs], Sep. 2024.

[17] Y. Hu, S. Fang, Z. Lei, Y. Zhong, and S. Chen, *Where2comm: Communication-Efficient Collaborative Perception via Spatial Confidence Maps*, arXiv:2209.12836 [cs], Sep. 2022.

[18] A. Chtourou, P. Merdrignac, and O. Shagdar, "Context-Aware Content Selection and Message Generation for Collective Perception Services," *Electronics*, vol. 10, no. 20, p. 2509, Oct. 2021, ISSN: 2079-9292. DOI: 10.3390/electronics10202509

[19] Y. Feng, H. Shen, Z. Shan, Q. Yang, and X. Shi, *Semantic Communication for Edge Intelligence Enabled Autonomous Driving System*, arXiv:2406.10606 [eess], Sep. 2024.

[20] W. Xu et al., *Semantic Communication for Internet of Vehicles: A Multi-User Cooperative Approach*, arXiv:2212.03037 [eess], Dec. 2022.

[21] Z. Qin, X. Tao, J. Lu, W. Tong, and G. Y. Li, *Semantic Communications: Principles and Challenges*, arXiv:2201.01389 [cs, eess, math], Jun. 2022.

[22] S. Iyer et al., "A Survey on Semantic Communications for Intelligent Wireless Networks," *Wireless Personal Communications*, vol. 129, no. 1, pp. 569–611, Mar. 2023, ISSN: 0929-6212, 1572-834X. DOI: 10.1007/s11277-022-10111-7

[23] J. Lu et al., *Generative AI-Enhanced Multi-Modal Semantic Communication in Internet of Vehicles: System Design and Methodologies*, arXiv:2409.15642 [cs], Sep. 2024.

[24] J. M. Gimenez-Guzman, I. Leyva-Mayorga, and P. Popovski, "Semantic V2X Communications for Image Transmission in 6G Systems," *IEEE Network*, pp. 1–1, 2024, ISSN: 0890-8044, 1558-156X. DOI: 10.1109/MNET.2024.3420214

[25] T. Lyu, M. Noor-A-Rahim, A. O'Driscoll, and D. Pesch, *Semantic Vehicle-to-Everything (V2X) Communications Towards 6G*, arXiv:2407.17186 [cs], Jul. 2024.

[26] S. Chauvin, "Hierarchical Decision-Making for Autonomous Driving," 2018, Publisher: Unpublished. DOI: 10.13140/RG.2.2.24352.43526

[27] X. Wang, X. Qi, P. Wang, and J. Yang, "Decision making framework for autonomous vehicles driving behavior in complex scenarios via hierarchical state machine," *Autonomous Intelligent Systems*, vol. 1, no. 1, p. 10, Dec. 2021, ISSN: 2730-616X. DOI: 10.1007/s43684-021-00015-x

[28] O. S. Tas and C. Stiller, "Limited Visibility and Uncertainty Aware Motion Planning for Automated Driving," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, Changshu: IEEE, Jun. 2018, pp. 1171–1178, ISBN: 978-1-5386-4452-2. DOI: 10.1109/IVS.2018.8500369

[29] C. Chen, M. Rickert, and A. Knoll, "Combining task and motion planning for intersection assistance systems," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, Gotenburg, Sweden: IEEE, Jun. 2016, pp. 1242–1247, ISBN: 978-1-5090-1821-5. DOI: 10.1109/IVS.2016.7535549

[30] L. Dulewicz et al., "Deliverable 1.2 : Accident parameters description for the chosen scenarios," SECUR Project Consortium (VUFO, UTAC, Volvo Cars, Continental, Renault), Europe, Project Deliverable D1.2, version 1.3, Apr. 2022, SECUR Project, WP1 – Accident Data Study.

[31] N. Shoaip et al., "Alzheimer's Disease Diagnosis Based on a Semantic Rule-Based Modeling and Reasoning Approach," *Computers, Materials & Continua*, vol. 69, no. 3, pp. 3531–

3548, 2021, ISSN: 1546-2226. DOI: 10.32604/cmc.2021.019069

[32] H. E. Massari et al., "Effectiveness of applying Machine Learning techniques and Ontologies in Breast Cancer detection," *Procedia Computer Science*, vol. 218, pp. 2392–2400, 2023, ISSN: 18770509. DOI: 10.1016/j.procs.2023.01.214

[33] C. Schlenoff, S. Balakirsky, M. Uschold, R. Provine, and S. Smith, "Using ontologies to aid navigation planning in autonomous vehicles," *The Knowledge Engineering Review*, vol. 18, no. 3, pp. 243–255, Sep. 2003, ISSN: 0269-8889, 1469-8005. DOI: 10.1017/S0269888904000050

[34] R. Regele, "Using Ontology-Based Traffic Models for More Efficient Decision Making of Autonomous Vehicles," in *Fourth International Conference on Autonomic and Autonomous Systems (ICAS'08)*, Gosier, Guadeloupe: IEEE, Mar. 2008, pp. 94–99, ISBN: 978-0-7695-3093-2. DOI: 10.1109/ICAS.2008.10

[35] L. Mlodzian et al., "nuScenes Knowledge Graph – A comprehensive semantic representation of traffic scenes for trajectory prediction," in *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, arXiv:2312.09676 [cs], Oct. 2023, pp. 42–52. DOI: 10.1109/ICCVW60793.2023.00011

[36] H. Bhuiyan, G. Governatori, A. Rakotonirainy, M. W. Wong, and A. Mahajan, "Driving Decision Making of Autonomous Vehicle According to Queensland Overtaking Traffic Rules," *The Review of Socionetwork Strategies*, vol. 17, no. 2, pp. 233–254, Oct. 2023, ISSN: 2523-3173, 1867-3236. DOI: 10.1007/s12626-023-00147-x

[37] ETSI SmartM2M Technical Committee, *Smartm2m; extension to saref; part 7: Automotive domain*, ETSI TS 103 410-7 V2.1.1 (2024-08), 2024.

[38] M. Compton et al., "The SSN ontology of the W3C semantic sensor network incubator group," *Journal of Web Semantics*, vol. 17, pp. 25–32, Dec. 2012, Publisher: Elsevier BV, ISSN: 1570-8268. DOI: 10.1016/j.websem.2012.05.003

## Appendix

### Annex: Ontology Specifications

TABLE I. ONTOLOGY CLASSES – ROAD TOPOLOGY

| **Class** | **Description** |
|---|---|
| *RoadElement* | |
| `RoadSegment` | A segment of the road, describing topographical proximity. |
| `Intersection` | A specific `RoadSegment` which describes the intersection between two or more `RoadSegments`. |
| `Lane` | A designated path describing topological proximity. |
| `MergingLane` | A subclass of `Lane`. |
| `Sidewalk` | A subclass of `Lane` which is a pedestrian path along the side of a road. |
| `PedestrianCrossing` | A subclass of `Lane` which designates areas for pedestrians to cross the road safely. |
| `ParkArea` | An area intended for parking vehicles. |
| *Traffic Management* | |
| `TrafficLight` | A signaling device used to control vehicle and pedestrian traffic. |
| `LaneRestriction` | A constraint or rule that limits how a lane can be used. |
| `Carpooling` | A subclass of `LaneRestriction`. Restriction allowing only high-occupancy vehicles in a lane. |
| `Closed` | A subclass of `LaneRestriction`. Indicates a lane that is temporarily or permanently inaccessible. |
| `VehicleType` | A subclass of `LaneRestriction`. Indicates the authorized vehicles (e.g., bus, bike). |
| `TurnDirection` | Represents a direction that a vehicle is allowed to take when leaving this lane. |
| `Left`, `Right`, `Front`, `Back` | Specific subclasses of `TurnDirection`. |
| *Environmental Context* | |
| `NonRoadElement` | An urban element not forming part of the road (e.g., buildings, trees). |

TABLE II. ONTOLOGY PROPERTIES – ROAD TOPOLOGY

| **Property** | **Description** |
|---|---|
| *RoadElement Properties* | |
| `hasLength` | Describes the length of a `RoadSegment`. |
| `hasLane` | Relates a `RoadSegment` to a `Lane`. |
| `hasIntersection` | Describes the relationship between a `RoadSegment` and an `Intersection`. |
| `hasSidewalk` | Links a `RoadSegment` to a `Sidewalk`. |
| *Traffic Management Properties* | |
| `hasLaneRestriction` | Relates a `Lane` to a `LaneRestriction`. |
| `hasTrafficLight` | Links a `RoadSegment` to a `TrafficLight`. |
| `hasTurnDirection` | Relates a `Lane` to a `TurnDirection`. |
| `isLaneOf` | Relates a `Lane` to a `RoadSegment`. |
| `isAdjacent` | Describes the adjacency of two `Lanes`. |
| `isConnected` | Indicates whether two `RoadSegments` are connected. |
| *Environmental Context Properties* | |
| `hasNonRoadElement` | Links a `RoadSegment` to `NonRoadElements` like trees or street furniture. |
| *Intersection Properties* | |
| `incomingLane` | Links an `Intersection` to an incoming `Lane`. |
| `outcomingLane` | Links an `Intersection` to an outcoming `Lane`. |
| *Switching Properties* | |
| `switchVia` | Relates a `Lane` to another `Lane` via a switching route. |
| `switchViaDashed` | A specific case of `switchVia`, where the route is dashed. |
| `switchViaIntersection` | A specific case of `switchVia`, involving a switch at an `Intersection`. |
| `switchViaStop` | A specific case of `switchVia`, involving a stop. |
| `switchVia TrafficLight` | A specific case of `switchVia`, involving a traffic light. |

TABLE III. ONTOLOGY CLASSES – AGENTS

| Class | Description |
|---|---|
| *Road Users* | |
| `RoadUser` | Any participant in road traffic. |
| `Vehicle` | A subclass of `RoadUser` representing motorized vehicles. |
| `ConnectedCar, EmergencyVehicle, Bus, Car, Truck` | Specific types of `Vehicle`. |
| `NonVehicle` | A subclass of `RoadUser` representing non-motorized or non-mechanical entities. |
| `Animal, Cyclist, Pedestrian` | Specific types of `NonVehicle`. |
| *Actions* | |
| `Action` | A generic action performed by a `RoadUser`. |
| `PedestrianAction` | Actions specific to `Pedestrian` and `Cyclist`. |
| `LookingAway, LookingRoad, Lying, Walking, Standing` | Specific types of `PedestrianAction`. |
| `VehicleAction` | Actions specific to `Vehicle`. |
| `ToLeftChange, ToRightChange, Accelerating, Decelerating, MaintainingSpeed, Stopping, TurningLeft, TurningRight, UTurn` | Specific types of `VehicleAction`. |
| *Profiles* | |
| `Profile` | Specifies the information required from the CPM for a `ConnectedCar`. |

TABLE IV. ONTOLOGY PROPERTIES – AGENT INTERACTIONS

| Property | Description |
|---|---|
| *RoadUser Actions* | |
| `isDrivingOn` | Associates a `Vehicle` with the `Lane` it is driving on. |
| `isStoppedOn` | Associates a `Vehicle` with the `Lane` it is stopped on. |
| `isWalkingOn` | Associates a `NonVehicle` with the `Lane` it is walking on. |
| `crossingInFront` | Indicates that a `NonVehicle` is crossing in front of a `Vehicle`. |
| `isDoing` | Indicates the `Action` being performed by a `RoadUser`. |
| *Visibility Properties* | |
| `masking` | Indicates that a `Vehicle` or `NonRoadElement` obstructs the view of a `RoadUser`. |
| `hasVisibility` | Specifies that a `Vehicle` has visibility of a given `RoadUser`. |
| `hasNoVisibility` | Specifies that a `Vehicle` does not have visibility of a given `RoadUser`. |
| *Event Participation* | |
| `hasEventParticipant` | Associates an `Event` with the `RoadUser(s)` involved. |
| `isParticipantOf` | Links a `RoadUser` to the `Event` in which they participate. |
| *Profile Properties* | |
| `wants` | Associates a `ConnectedCar` with a `Profile` specifying CPM-related requirements. |
| `isRelevant` | Links an `Event` to a `Profile` if the event is relevant to that profile. |

# SHAP_SVD: Integrating SHAP Values with Singular Value Decomposition for Conceptual Analysis in Regression

Yukari Shirota
Faculty of Economics
Gakushuin University
Tokyo, Japan
e-mail: yukari.shirota@gakushuin.ac.jp

Tamaki Sakura
Nikkei Economics Centre

Tokyo, Japan
e-mail: sakura@jcer.or.jp

*Abstract*— **In this paper, we propose a novel explainable AI (XAI) method for regression analysis, named SHAP_SVD, which integrates SHAP (SHapley Additive exPlanations) values with Singular Value Decomposition (SVD) to uncover latent structures in model interpretations. The Shapley value, a concept initially introduced by Lloyd Shapley in the field of cooperative game theory, has recently gained substantial attention in the AI community, particularly through its adaptation into the SHAP framework by Scott Lundberg. SHAP values enable us to interpret the contribution of each explanatory variable to a specific prediction by treating the prediction process as a game in which variables are players, and their marginal contributions are evaluated across all possible coalitions. In regression analysis, SHAP values can be seen as a matrix of attributions: for each observation, the contribution of each feature is calculated relative to a baseline. Our SHAP_SVD method applies SVD to this SHAP value matrix, thereby reducing dimensionality while preserving key information. The eigenvalues and corresponding eigenvectors obtained from SVD allow us to identify "concepts" or "latent semantic structures" that govern the interaction between features and the target variable. These concepts are encoded in both the left singular vectors. As a case study, we conducted a regression analysis of stock price growth rates for leading Indian and Japanese automobile manufacturers. The SHAP values were computed using a tree-based ensemble regression model, and our SHAP_SVD method was applied to reveal underlying structures. Two principal components emerged from the decomposition. In the extended analysis, we present a more detailed examination of the SHAP value distribution and structure. Specifically, we analyze how the SHAP captures nonlinear interactions that are invisible in traditional raw data correlation matrices. By comparing the performance of models evaluated using raw input features with those interpreted through SHAP, we demonstrate that SHAP-based interpretation yields greater stability, clarity, and interpretability, particularly in cases where multicollinearity or redundant variables obscure the true contribution of each feature. Our results demonstrate that SHAP analysis not only enhances the transparency of complex models but also, when combined with SVD, offers a powerful tool for discovering and visualizing conceptual dimensions underlying the data. The SHAP_SVD approach thus serves both as a diagnostic tool for regression models and a framework for semantic exploration in high-dimensional datasets.**

***Keywords- Explainable AI (XAI); SHAP values; Shapley theory; Singular Value Decomposition (SVD); Dimensionality reduction; Regression analysis; Latent semantic structure; Stock price analysis; Feature attributions.***

## I. INTRODUCTION

In this paper, we introduce a new Explainable AI (XAI) method named SHAP_SVD, which was first introduced in [1]. XAI has emerged as a critical field, bridging the gap between complex machine learning models and human interpretability. Among the numerous XAI techniques developed, Shapley values, introduced by Lloyd Shapley, have gained prominence for their ability to allocate the contribution of each feature in a model's predictions [2-4]. Adapted into the SHAP (SHapley Additive exPlanations) framework by Lundberg [5-7], this method has become a widely used tool for interpreting machine learning models, particularly in regression analysis [8-11]. In regression analysis, SHAP values quantify the contribution of each explanatory variable to the target value by utilizing characteristic functions of the data. These values offer deep insights into feature importance and interaction. However, as the complexity and dimensionality of the data increase, interpreting SHAP values becomes increasingly challenging. Traditional SHAP methods are limited in their ability to reveal underlying structures within the data, especially when dealing with high-dimensional or multi-faceted variables.

To address this limitation, we propose a novel method, SHAP_SVD, which applies Singular Value Decomposition (SVD) to the SHAP value matrix [1]. SVD, a well-known dimensionality reduction technique, captures the core structure of a matrix by decomposing it into eigenvalues and eigenvectors [12-14]. This allows us to extract latent semantic concepts or "principal components" from the SHAP matrix. By leveraging SVD, three matrices can be obtained, which are U, Σ, and $V^T$ (see Figure 1). This SHAP_SVD uncovers these underlying concepts, represented by two sets of eigenvectors—one from the UΣ matrix and the other from the $\Sigma V^T$ matrix—thus providing a richer understanding of the relationships between explanatory variables and the target variable.

As a concrete example, we apply SHAP_SVD to the regression analysis of stock price growth rates for Indian and

Japanese automakers, using the market capitalization growth rates as the target variable. Through this analysis, we identified two key latent concepts, which we refer to as (1) Balanced (Well-balanced) type, and (2) Sales Growth Rate (SGR)-driven type," extracted from the SHAP_SVD decomposition. By plotting company data on a two-dimensional plane defined by these two principal component axes, we can conduct a detailed analysis of the characteristics driving market capitalization growth for each company. This approach enables us to visualize and understand the underlying factors that influence the stock price performance of companies in both markets.

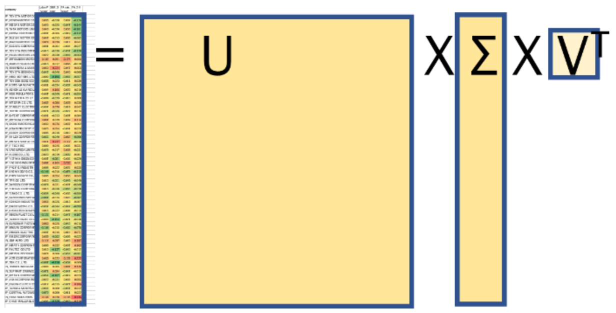


Figure 1. SVD of the given matrix.

The remainder of this paper is organized as follows. In Section II, we describe the data used for the analysis, including the sources of the data. Section III explains the methods applied, introducing both the SHAP analysis and our proposed SHAP_SVD method. Section IV presents the SHAP results, analyzing the contributions of explanatory variables to the target values. Section V details the SHAP_SVD method, illustrating how Singular Value Decomposition is applied to the SHAP matrix and how latent concepts are extracted. In Section VI, we discuss existing work related to explainable AI and dimensionality reduction, comparing these approaches with our proposed method. Section VII provides a discussion of the results and their implications. Finally, Section VIII concludes the paper with a summary of contributions and suggestions for future research.

## II. DATA

In this section, we shall explain the regression data. In the regression, we use Market Capitalization (MC) data. MC amount is a stock price times the number of issued stocks. The target variable is the Indian and Japanese automakers' "**annual MC growth rates**" in 2022. The MC growth rate in year XXX is defined as ***(MC_XXX – MC_(XXX-1)) / (MC_(XXX-1))***, namely, the ratio based on the previous year. We would like to identify the dominant factors contributing to the rapid growth rates of MC. The MC data used were retrieved from the ORBIS company database by Bureau van Dijk, with the last data update date being June 22, 2024. Measure the Indian automakers' MC changes. The damages caused by COVID-19 have exposed vulnerable supply chains in the automotive industry. This regression framework assumes that the competence of supply chains and new market development are prerequisites for the long-term sustainability of companies' high business performance, leading to high stock price evaluation [15]. Therefore, we select four managerial factors as the explanatory variables. Sales Growth Ratio (**SGR) represents the new market development competence**, and **FArate** represents the supply chain competence [15-17]. The Tangible Fixed Asset amount (FA) is the third explanatory variable used to identify the impact of the firm's scalability. These factors allow companies to earn satisfactory levels of profitability, such as their stock prices, ROE, and ROA. In addition, we focus on labor productivity. Labor productivity in the manufacturing sector refers to the goods or value one worker produces within a specific period. It is a crucial metric for assessing the efficiency and competitiveness of a manufacturing operation. We aim to evaluate which factor is more significant for the target tangible assets. Labor productivity was calculated here using the following formula:

$$Labor\ Productivity = \frac{Total\ Value\ Added}{Number\ of\ Workers}$$
$$= \frac{Net\ Sales - Cost\ of\ Goods\ Sold}{Number\ of\ Workers}$$

The managerial index data of the automobile companies were also retrieved from the ORBIS company database by Bureau van Dijk, with the last data update date being 2024/06/22. After removing companies with missing annual data, the number reached 67, including 11 Indian and 56 Japanese automakers. We conducted the regressions with the data.

## III. METHODS

In this section, the methods we used are described. The flow chart of the analysis is as follows:

1. **XGBoost Regression**: The given data is input into the XGBoost Regressor [18], and then the regression function f(X) is generated as output.
2. **SHAP Evaluation**: Based on the regression function f(X), SHAP values for each data are calculated. In this case study, we use four explanatory variables and 67 companies, resulting in a SHAP matrix of size 67 x 4.
3. **SVD of SHAP Matrix**: Applying Singular Value Decomposition (SVD) to the SHAP matrix M, the decomposition outputs three matrices such that $M=U\Sigma V^T$
4. **SHAP_SVD Interpretation**: The eigenvectors and eigenvalues extracted from SVD are interpreted to uncover underlying concepts. The two sets of eigenvectors are referred to as CompanyEigenVectors and SHAP_Eigenvectors, representing two different viewpoints of the underlying concepts.

SHAP (SHapley Additive exPlanations) is a method based on Shapley values from cooperative game theory, designed to explain machine learning model predictions, including those in regression tasks. A key strength of SHAP is its ability to create a characteristic function for the data, allowing it to calculate the contribution of each feature based on the characteristics of individual data points. This ensures that the contribution of each feature to the model's output is computed fairly and additively. SHAP enhances the interpretability of complex models, offering insights into how specific data characteristics influence predictions. We used XGBoost as the regression algorithm.

## IV. SHAP RESULTS

In this section, the results of the SHAP evaluation are presented.

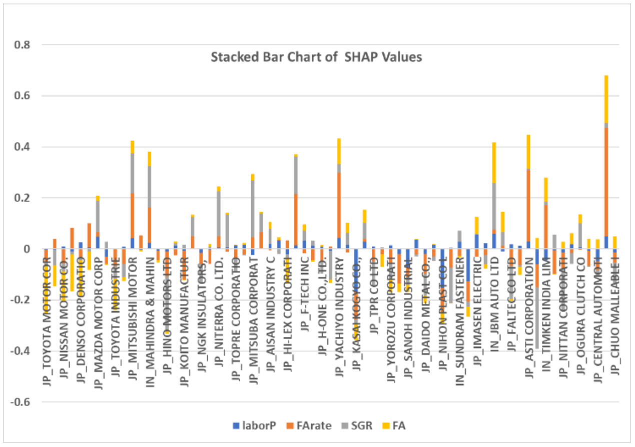


Figure 2. Stacked bar chart of the SHAP values.

The regression model, developed using XGBOOST, achieved an R-squared value exceeding 0.99, indicating a highly accurate fit to the data. Using the regression model f(X), the characteristic function is approximately evaluated. SHAP values are found based on the characteristic function. Figure 2 shows the SHAP values. The horizontal line shows the company IDs. Figure 2 illustrates a stacked bar chart of SHAP values of the individual companies. Each company has four SHAP values corresponding to the four explanatory variables.

The horizontal zero line in Figure 2 shows the average target value of the companies. Companies with positive values tend to be above-average performers. On the other hand, companies with negative values are evaluated as lower-performing than average. In corporate analysis, a fundamental question is whether a company performs above or below the industry standard. This is often the first benchmark used in evaluating a firm's standing. Therefore, the SHAP framework, which sets the industry average contribution to zero, aligns well with this evaluation logic. From a management science perspective, this interprets SHAP values as highly intuitive and meaningful.

The sum of four SHAP values in each company becomes its deviation from the target value. Table 1 shows the correlation coefficients between the target and the individual SHAP values. The highest correlation is between the FA rate_SHAP and the target, with a correlation coefficient of 0.84. The second highest is between SGR_SHAP and the target, with a value of 0.73. The third highest is between FA_SHAP and the target, with a value of 0.70. The result says that the most dominant factor for the target is FArate. Then, the second one is SGR.

Table 1. Correlation coefficients between the target and SHAP values.

| | target | laborP_SHAP | FArate_SHAP | SGR_SHAP | FA_SHAP |
|---|---|---|---|---|---|
| target | 1.00 | | | | |
| laborP_SHAP | 0.55 | 1.00 | | | |
| FArate_SHAP | 0.84 | 0.36 | 1.00 | | |
| SGR_SHAP | 0.73 | 0.24 | 0.41 | 1.00 | |
| FA_SHAP | 0.70 | 0.30 | 0.48 | 0.27 | 1.00 |

Figure 3 shows the relationship between FA_SHAP and the target as a scattering plot, and Figure 4 shows that between SGR_SHAP and the target. The SHAP correlation becomes higher than that between a raw feature value and the target.

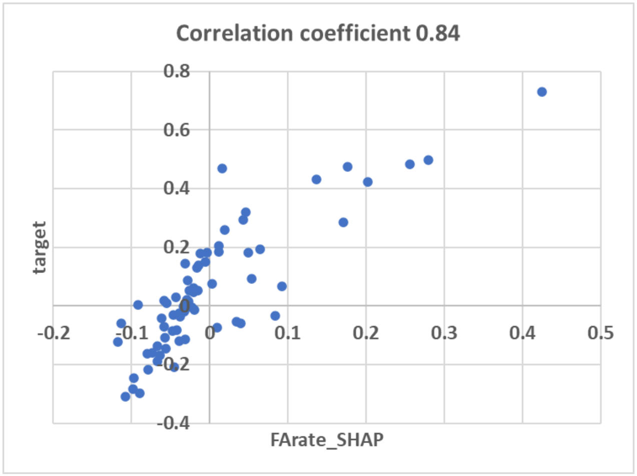


Figure 3. Scattering plot between FArate_SHAP and the target.

One key advantage of SHAP values over raw feature values is their ability to capture the actual influence of features on the target, even when simple correlations are weak. While a raw feature may show low correlation with the target variable, its corresponding SHAP value—reflecting its marginal contribution within the model—can exhibit a much stronger association. This occurs because SHAP values incorporate interactions and conditional dependencies that the model has learned. As a result, SHAP-based correlation analysis provides a more accurate representation of how features influence model predictions, particularly in complex, nonlinear, or high-dimensional settings.

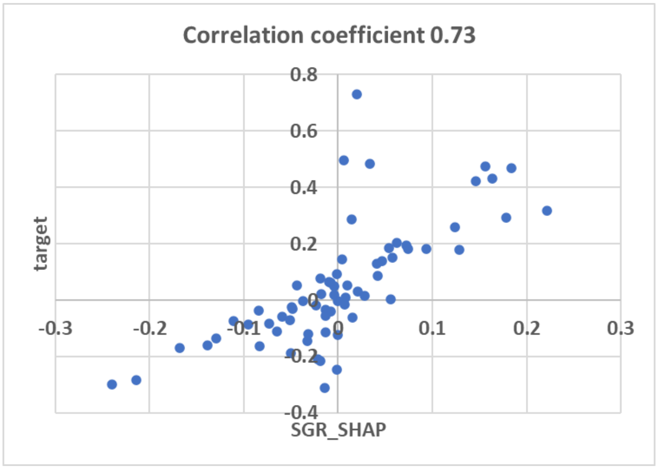


Figure 4. Scattering plot between SGR_SHAP and the target.

Another representation of the same set of SHAP values as in Figure 2 is the SHAP matrix M shown in Table 2. The matrix size becomes 67 times 4. The line represents a company, and each row displays an SHAP value. The matrix is divided by SVD.

Table 2. SHAP value matrix.

| company | laborP_SHAP | FArate_SHAP | SGR_SHAP | FA_SHAP |
|---|---|---|---|---|
| JP_TOYOTA MOTO | -0.003 | -0.045 | -0.021 | -0.188 |
| JP_HONDA MOTOF | -0.004 | 0.039 | -0.060 | -0.083 |
| JP_NISSAN MOTOF | 0.010 | -0.056 | -0.033 | -0.116 |
| IN_TATA MOTORS | -0.009 | 0.084 | -0.013 | -0.144 |
| JP_DENSO CORPOI | 0.026 | -0.031 | -0.013 | -0.146 |
| JP_SUZUKI MOTOF | 0.007 | 0.092 | -0.010 | -0.073 |
| JP_MAZDA MOTOF | 0.047 | 0.019 | 0.124 | 0.019 |
| JP_SUBARU CORP( | -0.031 | -0.027 | 0.028 | -0.005 |
| JP_TOYOTA INDUS | 0.000 | -0.067 | -0.050 | -0.122 |
| JP_ISUZU MOTORS | 0.008 | -0.059 | -0.051 | -0.016 |
| JP_MITSUBISHI MC | 0.041 | 0.176 | 0.156 | 0.050 |
| IN_MARUTI SUZUK | -0.004 | 0.054 | -0.001 | -0.006 |
| IN_MAHINDRA & M | 0.024 | 0.137 | 0.163 | 0.057 |
| JP_TOYOTA BOSH( | -0.008 | -0.034 | 0.000 | -0.009 |
| JP_HINO MOTORS | -0.009 | -0.098 | -0.214 | -0.010 |
| JP_TOYODA GOSEI | 0.012 | -0.026 | 0.010 | 0.007 |
| JP_KOITO MANUF/ | -0.007 | -0.113 | 0.015 | -0.004 |
| IN_ASHOK LEYLAN | -0.001 | 0.049 | 0.074 | 0.011 |
| JP_NGK INSULATO | -0.016 | -0.043 | -0.073 | 0.001 |
| JP_TOKAI RIKA CO | -0.004 | -0.055 | 0.007 | 0.013 |
| JP_NITERRA CO. L | 0.005 | 0.043 | 0.179 | 0.016 |

## V. SHAP_SVD METHOD

In this section, SHAP_SVD method is explained. After the SVD of the SHAP matrix, the singular value (SV) lists are obtained as the diagonal elements in matrix $\Sigma$ (see Figure 1).

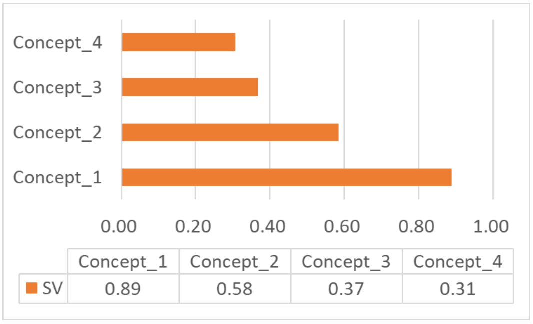


Figure 5. Singular values of the SHAP matrix M.

The SVs express, as shown in Figure 5, the strength of the latent concepts. The ratio is approximately 9:6:4:3. The eigenvalue of each concept becomes the square of the SV mathematically. Then the eigenvalues are those shown in Figure 6. The percentages of eigenvalues are shown in Table 3. Approximately 58% of the concepts can be explained by concept 1. The second concept has about 25%. Therefore, an overview of the structure can be captured simply by examining the first and second concepts.

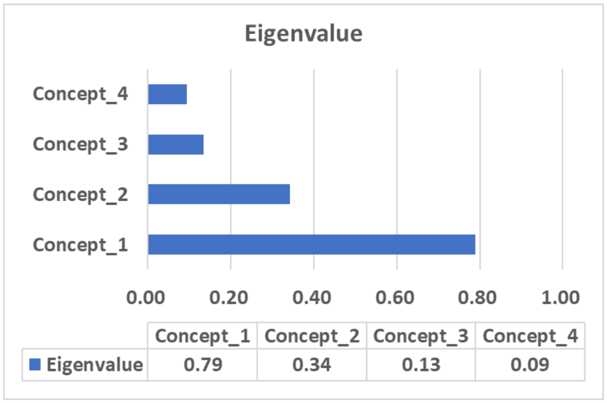


Figure 6. Eigenvalues of the matrix $U\Sigma$.

Table 3. Eigenvalue percentage of the concepts.

| Concept_1 | Concept_2 | Concept_3 | Concept_4 |
|---|---|---|---|
| 0.79 | 0.34 | 0.13 | 0.09 |
| 58% | 25% | 10% | 7% |

Then, we will interpret the meaning of the concepts, focusing on the two largest SVs. Two matrices are pre-constructed through matrix multiplication in advance: $U\Sigma$ and $\Sigma V^T$ (see Figure 7).

The individual concept can be represented by two expressions: CompanyEigenVectors and SHAP_EigenVectors as shown in Figure 7. The terminology used here was defined specifically for this case study and does not represent a general or standardized name. In other contexts, it could be referred to as *DateEigenVector* or other case-appropriate labels.

For example, concept 1 is defined by the first row of $U\Sigma$, and is represented by the first line of $\Sigma V^T$ (see Figure 7).

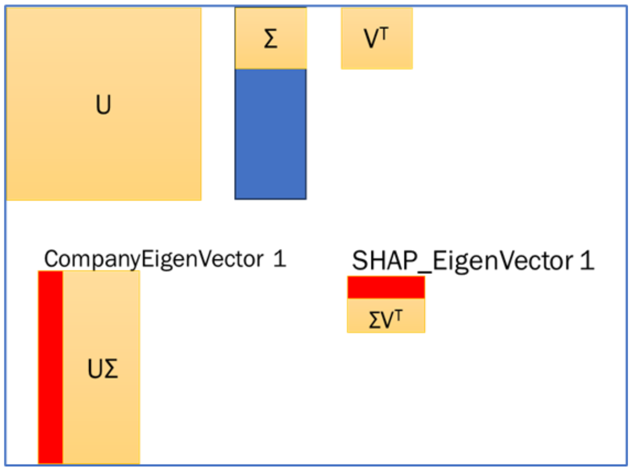


Figure 7. CompanyEigenVectors and SHAP_EigenVectors in SVD.

First, using SHAP_EigenVectors, the concepts will be expressed in Figure 8. The bottom bar graph presents SHAP_EigenVector_1, which has four elements. The upper graph shows SHAP_EigenVector_2, which has four elements as well.

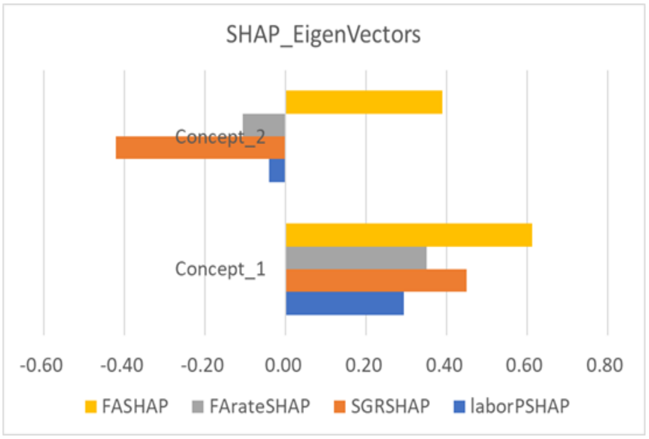
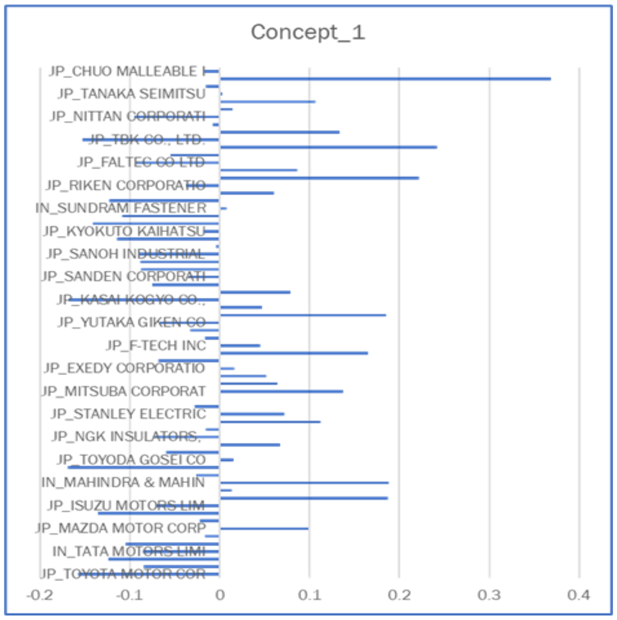


Figure 8. SHAP elements of Eigenvalues 1 and 2.

Our interpretation of the two concepts is as follows:

- 1st concept: All elements cooperate and have high values, with particularly high SHAP for FA.
- 2nd concept: SGR_SHAP is high, and FA's SHAP is low (expressing it this way reverses the sign of the vector elements).

As a result, we name the two concepts as
(1) Balanced (Well-balanced) type and
(2) Sales Growth Rate (SGR)-driven type.



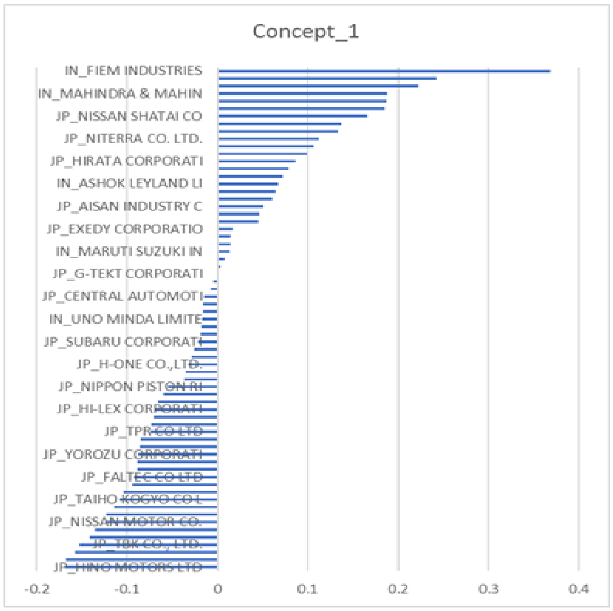


Figure 9. CompanyEigenVector_1 (the bottom is the sorted one).

Then, using CompanyEigenVectors, the concepts will be interpreted. Figure 9 shows CompanyEigenVector_1. The bottom graph is the sorted version. The largest element company was FIEM Industries. FIEM is a well-established company in India, primarily known for its expertise in automotive lighting. With over 50 years of experience, FIEM has grown into a leading supplier for original equipment manufacturers (OEMs) in India and abroad. In the representation using CompanyEigenVectors, the first concept can be interpreted as companies with SHAP distributions similar to FIEM.

Figure 10 shows CompanyEigenVector_2. The lowest value company is JP_NISSAN SHATAI, and the third lowest company is TOYOTA. The highest value company is JP_CHUO MALLEABLE.

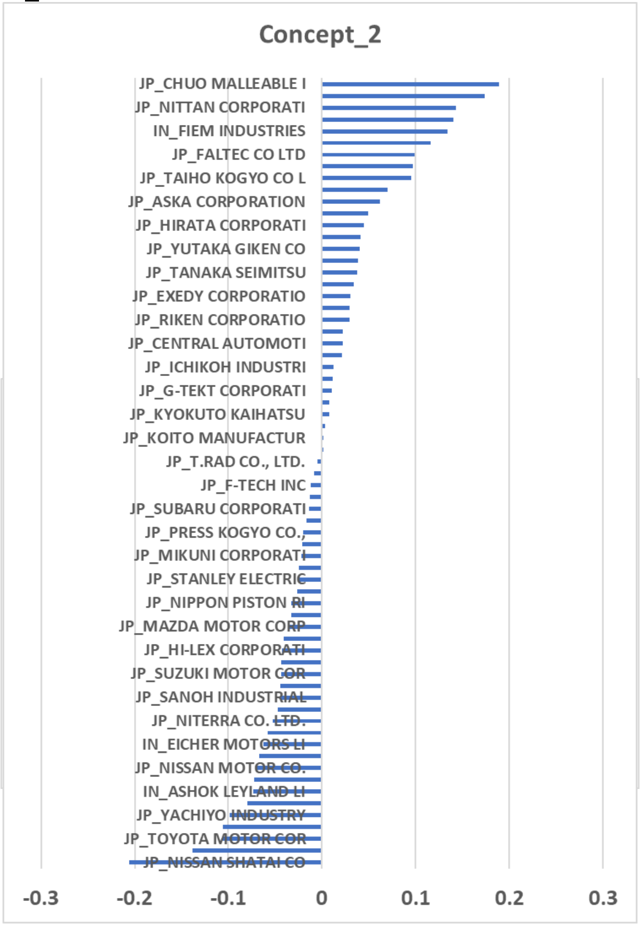


Figure 10. CompanyEigenVector_2.

Figure 11 shows a scattering plot of CompanyEigenVector_1 elements and CompanyEigenVector_2 element values of 67 companies. The x-axis represents the concept 1 level, and the y-axis represents the concept 2 level.

First, the five companies with the highest CompanyEigenVector_1 values will be explained in Figure 12, along with their corresponding SHAP values. The FIEM and ASTI are included there. The five companies are well-balanced, and the target deviation values are positive, with higher values. Seeing these five companies' SHAP distributions, the ratio of FA_SHAP and SGR_SHAP are higher than others, which are the same as the concept 1 representation in Figure 8; SHAP_EigenVector 1.

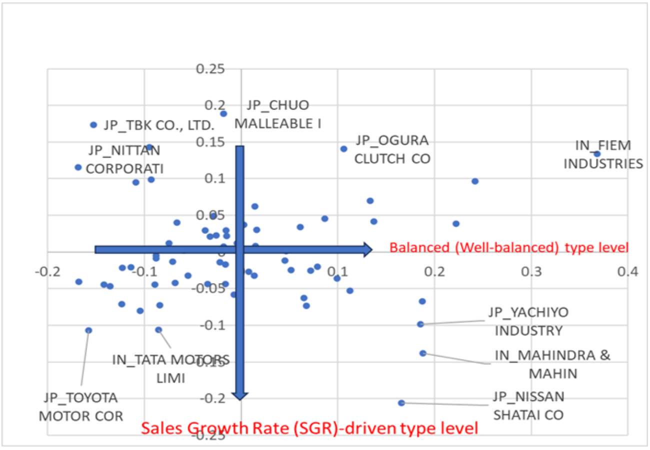


Figure 11. A scattering plot of CompanyEigenVector_1 elements and CompanyEigenVector_2 element values of 67 companies.

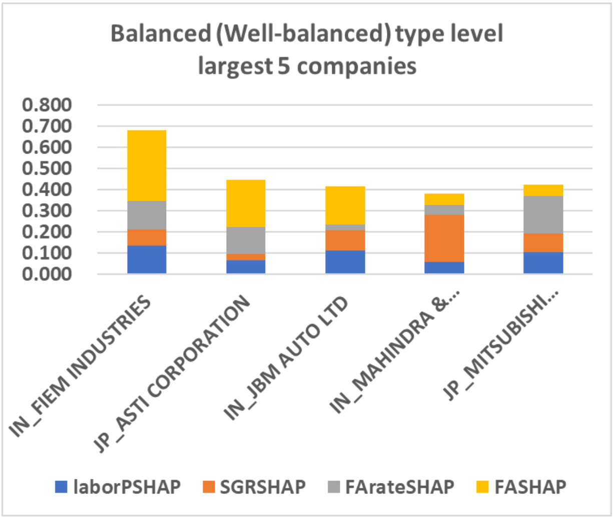


Figure 12. SHAP values of the Balanced type level of the five largest companies.

On the other hand, the five companies with the lowest CompanyEigenVector_1 in Figure 13 have negative target deviation values. The distribution of SHAP values on each company represent the company characteristics. From the comparison between Figures 12 and 13, a company with higher concept 1 value tends to be higher in the target value.

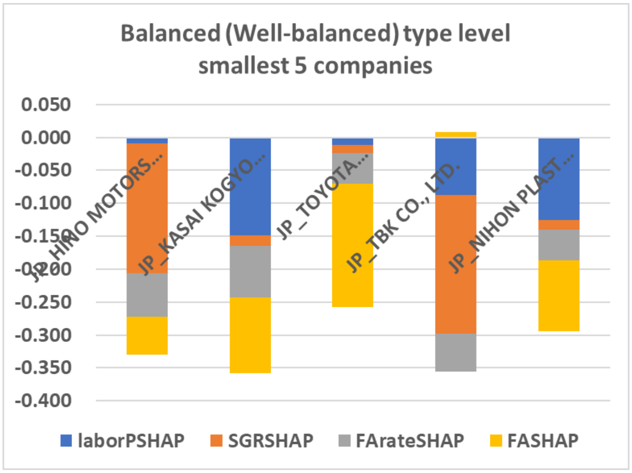


Figure 13. SHAP values of the Balanced type level of the five lowest companies.

Next, we will see SGR-driven type concept 2. Figure 14 illustrates the SHAP values of the top five companies in the SGR-driven type level. As shown in Figure 11, the second concept (y-axis) is oriented downwards, and the company with the highest y-value is Nissan Shatai, followed by Mahindra, Toyota, Tata, and Yachiyo. Figure 14 shows these five companies' stacked bar chart of SHAP values.

The target deviation positive three companies are SGR-driven ones. On the other hand, the other two companies are anti-SGR driven and have lower FA_SHAP values. As shown in Figure 8, concept 2 exhibits a higher SGR_SHAP and lower FA_SHAP. There is a reverse relationship between SGR and FA. Therefore, these anti-SGR-driven companies show the feature of concept 2, and these are evaluated as higher value companies concerning concept 2.

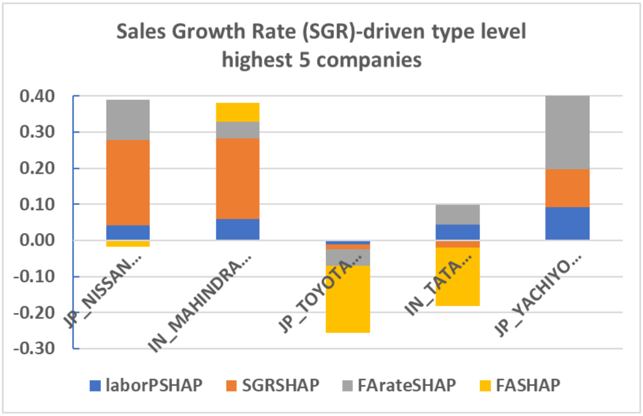


Figure 14. SHAP values of the SDG-driven type level of the five largest companies.

Next, we will evaluate the SHAP values of the five smallest companies of the SDG-driven type level (see Figure 15). In the three companies with positive SHAP values, FA_SHAP (Fixed Assets SHAP) is large, while SGR_SHAP is small, indicating that these companies are driven more by the size of their tangible assets rather than sales growth. For the two companies with negative target deviation values, SGR_SHAP is dragging performance less than FA_SHAP, with FA_SHAP values being close to zero. This suggests that FA has minimal impact on these two companies.

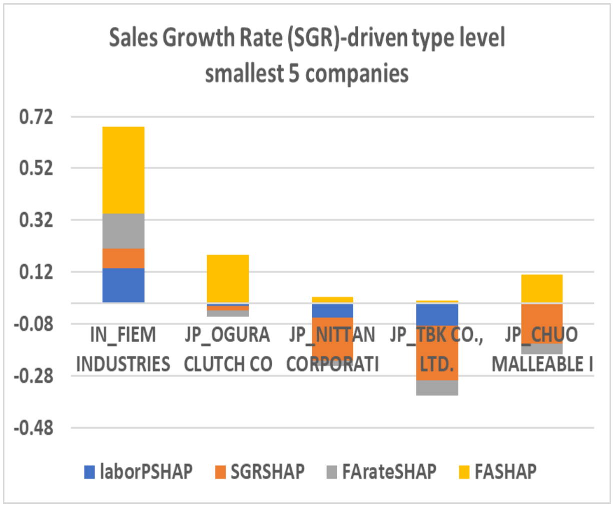


Figure 15. SHAP values of the SDG-driven type level of the smallest five companies.

To summarize the findings thus far, **concept 1 shows a positive relationship with the target deviation**. Companies with higher values of concept 1 also tend to exhibit higher target deviations. In contrast, **concept 2 exhibits mixed signs in its component values**, depending on the relative dominance of different SHAP contributions. This variation arises from the structure of concept 2 itself, which is characterized by a **positive loading on SGR_SHAP** and a **negative loading on FA_SHAP**. As a result, the overall direction of concept 2 depends on which factor exerts a stronger influence in a given company's SHAP profile.

## VI. Existing Work

In this section, the related existing works are presented. The first allocation field involves stock price evaluation, and the second allocation field utilizes text mining.

(1) Random Matrix Theory (RMT) and portfolio

RMT has been applied to stock market analysis to reduce noise in financial data. RMT helps distinguish real market signals from random fluctuations in stock price correlations [19-24]. The flow charts of the method are as follows:

1. Correlation Matrix: Begin by calculating the correlation matrix of stock returns. This matrix sizes the number of companies times the number of sales dates.
2. RMT Filtering: RMT is used to separate meaningful signals from random noise. Eigenvalues of the correlation matrix are compared with theoretical RMT

predictions. Larger eigenvalues represent accurate market information, while smaller ones reflect noise.

3. SVD (Singular Value Decomposition): SVD is applied to clean the correlation matrix further, focusing on the significant components. This improves the matrix's accuracy, filtering out noise.
4. Portfolio Optimization: Using the noise-reduced correlation matrix, more accurate risk and return estimates can be made, improving portfolio construction.

(2) Latent Semantic Analysis (LSA)

LSA is a widely used technique in natural language processing (NLP), primarily for analyzing semantic relationships between documents. It is often applied in tasks such as topic modeling, semantic analysis, and information retrieval [25-31].

Overview of LSA:

1. Purpose: LSA aims to convert the semantic relationships between words and documents into a lower-dimensional latent semantic space, allowing for the identification of similarities and relationships between documents. This helps uncover hidden patterns or topics within the text.
2. Method: LSA begins by creating a co-occurrence matrix that captures how often words appear together in a document. This matrix models the relationships between words and documents. Then, SVD is applied to reduce the dimensionality of the matrix. By using SVD, LSA compresses the high-dimensional data while preserving the important semantic relationships and filtering out noise.

LSA is a powerful mathematical approach for interpreting the semantic structure of text and is utilized in search engines, automatic summarization systems, document clustering, and more. SVD techniques are mathematically explained in [12, 14]. The two kinds of eigen vectors and the relationship among the three decomposed matrices are clearly explained using visualization in [32].

## VII. Discussion

In this section, we will discuss the result. The objective of the analysis is to group companies. The proposed SHAP_SVD method can extract the essence of the given SHAP value matrix.

In the case of India and Japan automakers, the two extracted concepts were obtained: (1) Balanced (Well-balanced) type, and (2) Sales Growth Rate (SGR)-driven type. The concept can be represented as two aspects in this case that are CompanyEigenVectors and SHAP_EigenVectors. There are found as the results of SVD.

Using each CompanyEigenVectors' element values, we can measure each company's (1) Balanced (Well-balanced) type level and (2) Sales Growth Rate (SGR)-driven type level. As shown in Figure 11, the scattering plot of the companies, generated by CompanyEigenVectors, can reveal the individual characteristics of each company. The horizontal axis represents the Balanced (Well-balanced) type level. These higher-level companies can be divided along the vertical axis into two groups: an "SGR_SHAP higher and FA_SHAP lower" group and a "FA_SHAP higher and SGR_SHAP lower" group. This means that these companies exhibit a similar pattern of feature contributions, reflecting a particular type of balance or focus in their business models.

SHAP values can more accurately reflect each company's characteristics than using the raw input data. Therefore, analyzing SHAP values through SVD (Singular Value Decomposition) allows for more accurate dimensionality reduction based on the characteristics of each company. This method enhances the ability to capture distinct business drivers by compressing the data in a way that aligns with each company's unique attributes, offering deeper insights compared to standard SHAP analysis.

In corporate management, creating appropriate KPIs (Key Performance Indicators) is crucial. The EigenVectors (principal component axes) derived from SHAP_SVD analysis can serve as the first step in developing these KPIs. By identifying the most critical factors influencing business performance through dimensionality reduction, SHAP_SVD helps to highlight key metrics that align with a company's unique characteristics, providing a strong foundation for effective KPI creation.

Although the present case study focuses on four managerial indicators, the SHAP_SVD framework is model-agnostic and can be directly applied to high-dimensional settings. Because SHAP_SVD operates on the SHAP attribution matrix rather than the raw input space, it can handle models that generate large attribution vectors, including those used for image, audio, or text data. In contrast, traditional neural-fuzzy approaches rely on human-interpretable rule bases, and the number of rules increases exponentially as the number of input variables grows, making high-dimensional applications structurally difficult. Therefore, SHAP_SVD provides a more flexible and scalable approach for extracting latent concepts from modern machine-learning models.

## VIII. Conclusions

In this paper, we propose a novel explainable AI (XAI) methodology named SHAP_SVD, which integrates SHAP values with Singular Value Decomposition (SVD) to extract latent semantic structures from regression models. This approach addresses a key challenge in modern machine learning applications—the difficulty of interpreting complex models in high-dimensional settings.

While SHAP values already offer a theoretically sound and practically beneficial way to understand feature contributions, they often lack the structural abstraction needed for global interpretation and cross-sample comparison. Our SHAP_SVD method fills this gap by applying dimensionality reduction to the SHAP value matrix, revealing underlying concepts that govern the feature-target relationships.

In our case study on stock price growth rates of Indian and Japanese automakers, the SHAP_SVD analysis revealed two principal latent concepts:

(1) a Balanced (Well-balanced) type, characterized by uniform positive contributions across features, especially Fixed Assets (FA), and
(2) an SGR-driven type, emphasizing Sales Growth Rate (SGR) over other explanatory variables.

These concepts were visualized using the eigenvectors derived from the SHAP matrix decomposition, providing two complementary perspectives CompanyEigenVectors, which characterize companies, and SHAP_EigenVectors, which describe feature patterns. This framework offers several significant contributions: First, SHAP_SVD enhances interpretability by translating local explanations (SHAP values per sample) into global structures (principal components), allowing researchers and analysts to detect patterns and clusters among entities. Second, it provides a foundation for data-driven KPI (Key Performance Indicator) design, where companies can be evaluated along meaningful conceptual axes extracted from model behavior, not just raw data.

Moreover, SHAP_SVD's utility is not limited to regression problems in the automotive sector and company sector. Its generality suggests broad applicability in other domains such as finance, healthcare, education, and policy evaluation—any field in which explainable AI meets high-dimensional structured data. For instance, medical diagnostics models could benefit from latent factor discovery among symptoms and test results, or macroeconomic models could use SHAP_SVD to explain and classify country-level performance indicators.

Future work will explore the following directions:

- Extending SHAP_SVD to classification tasks, where class probability contributions rather than regression outputs are analyzed.
- Combining SHAP_SVD with clustering algorithms to automatically group entities based on shared conceptual characteristics.
- Developing interactive visualization tools, enabling stakeholders to navigate the concept space in a user-friendly manner.

In conclusion, SHAP_SVD is a promising tool for conceptual interpretation of machine learning models. By bridging SHAP's local attribution with SVD's global structure extraction, it provides a robust and interpretable framework for analyzing the inner workings of predictive models. We believe this methodology offers not only academic value but also practical utility for decision-makers aiming to derive actionable insights from complex datasets.

## Acknowledgment

The research was partly supported by the fund of Gakushuin University Computer Centre project 2024 and Gakushuin University Economics and Management project 2025.

## References

[1] Y. Shirota and T. Sakura, "Exploring Latent Concepts in SHAP Values -A New Approach Using Singular Value Decomposition -," in *DBKDA 2025*, Lisbon, Portugal, 2023/03/9-13 2025: IARIA XPS Press, pp. 1-6, doi: 978-1-68558-244-9. [Online]. Available: https://www.thinkmind.org/library/DBKDA/DBKDA_2025/dbkda_2025_1_10_58001.html (accessed 2025/11/14).
[2] A. E. Roth, "Introduction to the Shapley Value," *The Shapley value,* pp. 1-27, 1988.
[3] A. E. Roth, *The Shapley Value: essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988.
[4] L. S. Shapley, "A value for n-Person Games, Contributions to the Theory of Games, 2, 307–317," ed: Princeton University Press, Princeton, NJ, USA, 1953.
[5] S. M. Lundberg, G. G. Erion, and S.-I. Lee, "Consistent Individualized Feature Attribution for Tree Ensembles," *arXiv preprint arXiv:1802.03888,* 2018.
[6] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model predictions," *Advances in neural information processing systems,* vol. 30, 2017.
[7] S. M. Lundberg and S.-I. Lee, "Consistent Feature Attribution for Tree Ensembles," *arXiv preprint arXiv:1706.06060,* 2017.
[8] A. R. Javed, W. Ahmed, S. Pandya, P. K. R. Maddikunta, M. Alazab, and T. R. Gadekallu, "A Survey of Explainable Artificial Intelligence for Smart Cities," *Electronics,* vol. 12, no. 4, p. 1020, 2023.
[9] R. Dwivedi *et al.*, "Explainable AI (XAI): Core Ideas, Techniques, and Solutions," *ACM Computing Surveys,* vol. 55, no. 9, pp. 1-33, 2023.
[10] A. Chaddad, J. Peng, J. Xu, and A. Bouridane, "Survey of Explainable AI Techniques in Healthcare," *Sensors,* vol. 23, no. 2, p. 634, 2023.
[11] Y. Shirota, K. Kuno, and H. Yoshiura, "Time Series Analysis of Shap Values by Automobile Manufacturers Recovery Rates," in *Proceedings of the 2022 6th International Conference on Deep Learning Technologies*, 2022, pp. 135-141.
[12] C. M. Bishop and N. M. Nasrabadi, *Pattern Recognition and Machine Learning* (no. 4). Springer, 2006.
[13] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.
[14] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. Elsevier, 2006.
[15] Y. Shirota, M. Fujimaki, E. Tsujiura, M. Morita, and J. A. D. Machuca, "A SHAP Value-Based Approach to Stock Price Evaluation of Manufacturing Companies," in *2021 4th International Conference on Artificial Intelligence for Industries (AI4I)*, 2021: IEEE, pp. 75-78.
[16] M. Fujimaki, E. Tsujiura, and Y. Shirota, "Automobile Manufacturers Stock Price Recovery Analysis at COVID-19 Outbreak," in *6th World Conference on Production and Operations Management – P&OM Nara 2022*, Nara, Japan, 2022: EurOMA (European Operations Management Association), pp.1-6, Decision Science Institute Best Paper Award.
[17] K. Yamaguchi, "Relationship Analysis Between Stock Prices and Financial Statements in the Automobile Industry," in *2023 14th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)*, 2023: IEEE, pp. 442-445.
[18] XGBoostDevelopers. "XGBoost Documentation (Revision 534c940a.)." Available: https://xgboost.readthedocs.io/en/stable/ (accessed 2025/11/14).
[19] V. Plerou, P. Gopikrishnan, B. Rosenow, L. A. N. Amaral, T. Guhr, and H. E. Stanley, "Random Matrix Approach to Cross Correlations in Financial Data," *Physical Review E,* vol. 65, no. 6, p. 066126, 2002.

[20] M. Potters, J.-P. Bouchaud, and L. Laloux, "Financial Applications of Random Matrix Theory: Old laces and new pieces," *arXiv preprint physics/0507111,* 2005.

[21] A. Utsugi, K. Ino, and M. Oshikawa, "Random Matrix Theory Analysis of Cross Correlations in Financial Markets," *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics,* vol. 70, no. 2, p. 026110, 2004.

[22] Z. Bai, H. Liu, and W. K. Wong, "Enhancement of the Applicability of Markowitz's Portfolio Optimization by Utilizing Random Matrix Theory," *Mathematical Finance: An International Journal of Mathematics, Statistics and Financial Economics,* vol. 19, no. 4, pp. 639-667, 2009.

[23] H. Aoyama, Y. Fujiwara, Y. Ikeda, H. Iyetomi, and W. Souma, *Econophysics and Companies: Statistical Life and Death in Complex Business Networks*. Cambridge University Press, 2010.

[24] J. Bun, J.-P. Bouchaud, and M. Potters, "Cleaning Large Correlation Matrices: Tools from Random Matrix Theory," *Physics Reports,* vol. 666, pp. 1-109, 2017.

[25] T. K. Landauer, P. W. Foltz, and D. Laham, "An Introduction to Latent Semantic Analysis," *Discourse processes,* vol. 25, no. 2-3, pp. 259-284, 1998.

[26] N. Evangelopoulos, X. Zhang, and V. R. Prybutok, "Latent Semantic Analysis: Five Methodological Recommendations," *European Journal of Information Systems,* vol. 21, no. 1, pp. 70-86, 2012.

[27] S. T. Dumais, "Latent Semantic Analysis," *Annual Review of Information Science and Technology (ARIST),* vol. 38, pp. 189-230, 2004.

[28] T. K. Landauer, D. S. McNamara, S. Dennis, and W. Kintsch, *Handbook of Latent Semantic Analysis*. Psychology Press, 2007.

[29] S. T. Dumais, G. W. Furnas, T. K. Landauer, S. Deerwester, and R. Harshman, "Using Latent Semantic Analysis to Improve Access to Textual Information," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1988, pp. 281-285.

[30] Y. Shirota and B. Chakraborty, "Visual Explanation of Eigenvalues and Math Process in Latent Semantic Analysis," *Information Engineering Express,* vol. 2, no. 1, pp. 87-96, 2016.

[31] Y. Shirota and B. Chakraborty, "Visual Explanation of Mathematics in Latent Semantic Analysis," in *2015 IIAI 4th International Congress on Advanced Applied Informatics*, 2015: IEEE, pp. 423-428.

[32] Y. Shirota and B. Chakraborty, "Visual Explanation of Eigenvalues and Math Process in Latent Semantic Analysis," *Information Engineering Express, Information Engineering Express,* vol. 2, no. 1, pp. 87-96, 2016. [Online]. Available: https://www.iaiai.org/journals/index.php/IEE/article/view/70 (accessed 2025/11/14).

# On the Integration of Formal Model Transformations with Manual Document Manipulations in Software Engineering Processes

Hans-Werner Sehring
Department of Computer Science
Nordakademie
Elmshorn, Germany
e-mail: sehring@nordakademie.de

*Abstract*—**Model-driven software engineering processes are based on formal models that can be automatically transformed into one another after specifications are added. However, the creation of many software products involves creative activities that result in manually generated, informal documents, which prevent automated model transformations. To enable transformation steps, the content of these documents must be accessed in a structured way when integrating them into model-driven processes. Manually maintained documents are subject to frequent changes, including modifications to their structure. To enable model-driven processes in the presence of creative activities and their documents, we are currently experimenting with parsing techniques that combine the structure of documents with domain knowledge about their content. First experiments are based on the Minimalistic Metamodeling Language and its ability to integrate semantic descriptions with syntactic representations.**

***Keywords-software development; software engineering; computer aided software engineering; top-down programming; document handling.***

## I. Introduction

Software engineering processes involve creating and consuming a series of documents. Such documents link the various phases of activity in software creation processes, whether experts perform sequential work in phase-oriented projects or cross-functional teams collaborate simultaneously using agile approaches.

In general, documents may contain various types of content, such as problem statements, requirements, constraints, domain models, solution models, abstract and concrete descriptions of (software) implementations, tests cases, data, configurations, design decisions, specifications of development and quality assurance processes, and user and maintenance manuals.

The documents' formats are diverse. They include text documents, figures, (states of) collaborative digital whiteboards, interactive presentations, prototypes, and workshops protocols (including photos of whiteboards, for instance).

As such, managing the series of documents created during a (software) development process resembles content management tasks.

The way documents are handled depends on the software development approach taken.

- Documents may be manually created, human-perceivable representations of content. This is how documents are managed in human-centered processes, particularly creative ones.
- Documents may also represent formal models. In this case, formality allows for automated transformation steps in software development and for explicit traceability. This notion of documents is found in model-driven approaches. Documents may also be generated from models as they are in content management processes.
- Dialogs between a user and a *Generative Artifical Intelligence* (*GenAI*) system can also be considered documents. Each dialogue consists of a series of prompts used as input for the GenAI, as well as the AI's responses to these prompts. The utilization of GenAI in software engineering processes is currently being researched, for instance, under the term *vibe modeling*.

Combinations of these software development methods may be of interest. This article presents a step toward integrating of creative manual work on documents into model-driven processes. It provides an extension of the work presented in [1].

The class of software engineering processes that are based on documents that contain formal models are called *Model-Driven Software Engineering* (*MDSE*) or *Model-Driven Software Development* (*MDSD*) processes [2].

Software engineering processes that include creative activities, such as conceptual modeling or interaction design [3], are often applied for interactive software. Creative activities are supported by documents that have neither a common format [4] nor formal semantics. Instead, they reflect subjective impressions, case-based presentations, alternatives, and similar content directed at a human audience.

Documents that lack formal structure cannot participate in MDSE processes per se. However, they can be annotated by their creators with, for example, with references to relevant content that are sufficiently fine-grained to address well-formed content. Such annotations allow creative documents to participate in MDSE processes.

However, such annotations refer to specific document *instances*. Documents used in creative activities are, in particular, working documents that are subject to constant change. This includes changes in the structure of the documents. Therefore, any fixed reference to content in such a document will potentially become invalid and metadata may become inconsistent as work progresses.

In this article, we investigate means of integrating informal documents, in particular ones that are subject to change, into (model-driven) software engineering processes. We are currently experimenting with linguistic means of recognizing the content of documents with changing structures. First experiments with document recognition are based on a modeling language and its special ability to integrate semantic descriptions with syntactic representations.

Preliminary results show that at least some content can be extracted from documents that lack formal representations. In this way, model-driven approaches can potentially be applied to software projects with creative aspects.

GenAI allows another take on the problem. It is specifically suited to generate formal representations from natural language descriptions. MDSE approaches based on GenAI are beyond the scope of this article.

The remainder of this article is organized as follows: In Section II, we revisit model-driven software engineering and discuss the need for incorporating informal documents. Section III presents typical ways of referencing content in single documents, and it addresses means of managing volatile references to content of mutable documents. Section IV briefly introduces a modeling language that is used for initial experiments in this article. An experimental implementation of these concepts is presented in Section V. The article concludes in Section VI with a summary and an outlook on future work.

## II. Visual Software Engineering Artifacts

The discourse in this article does not require a comprehensive introduction to model-driven approaches. However, this section introduces some basic terms and highlights the challenges of integrating creative work.

### A. Model-Driven Software Engineering

In software development processes, a series of documents is created. The kinds of documents may differ depending on the kind of software being created and on the methodology used for the process. But all documents serve common purposes, such as linking activities by the results represented in them, allowing traceability of activities [5], and others.

MDSE formalizes the flow of documents and thus the connection of development steps. Documents are *models* with a formal semantics. Models are derived by means of *model-to-model transformations* and finally to code in *model-to-text transformations* on a (semi-) automatic basis. This way, development steps can be performed (semi-) automatically and changes to models can be propagated down the model chain.

One of the first prominent examples of MDSE is the Object Management Group's *Model-Driven Architecture* (*MDA*). Various other approaches have emerged that differ in the way in which they implement transformations, for example, by means of metaprogramming [7], code templates [8], or GenAI [9].

### B. Creative Software Development Activities

Certain kinds of software solutions, such as those with a focus on the human-machine interface, include creative steps. Examples of creative activities are the definition of interaction patterns, of user experience in general, and user interface design in particular [10].

In [11], we use the term *Model-Supported Software Creation* (*MSSC*) to distinguish this kind of software development from general MDSE that relies purely on formal representations.

Figure 1 shows typical groups of artifacts created to model aspects of a software solution. Each group of artifacts belongs to a *modeling stage*, depending on the chosen development approach. Note that the sequence of artifacts is not meant to prescribe a temporal order. Depending on the development process, the artifact groups relate to consecutive activities, or they are just a classification of the outcomes of activities that are performed in an interleaved and, eventually, iteratively manner.

A first set of artifacts, in Figure 1 called *Preparation*, reflects the decisions to be made before starting a software development project. This includes identifying the problem, deciding to start a project to solve it, concluding that software is part of the overall solution, setting concrete project goals, and allocating resources.

The artifacts in the *Concept* stage take the perspective of the application domain. A *Domain model* will finally capture that application domain. Creative projects often start with *Personas*, stereotypical users for whom the software will be designed. *Customer Journeys* A *Solution hypothesis* gives a first idea of the software to be developed as part of an overall problem solution.

Typical creative processes, such as User-Centered Design or Design Sprint, rely on all stakeholders taking part in the design process. Since a discussion on the basis of models is too abstract for many domain experts, a series of prototypes that visualize the ideas is employed. Prototypes of different development stages are often called *Lo-Fi Prototypes* and *Hi-Fi Prototypes*. Low fidelity prototypes emphasize the information structure (*Information architecture*) and the rough layout of the software's use interface (*Wireframe*, *Module catalog*, *Navigation*). High fidelity prototypes provide a preview of the software to be developed based on the current insights. For this, the look of the software's user interface is designed in detail (*Style guide*) and some functionality is provided in usable or emulated form (*Click dummy*). The goal is to validate the solution hypothesis in user experiences tests (*UX Tests*).

*Solution Architecture* provides the transition from the domain perspective to a technical perspective. In includes a first idea of the software's structure, its interface, the required infrastructure and other first implementation decisions. The artifacts created to define the solution architecture depend on the kind of software to be developed. In ecommerce applications, for example, the customer journeys and the touchpoints along each journey provide valuable input. From these, the demand for functionality and data is derived by a *Touchpoint-data mapping* and a *touchpoint-function mapping*. These lead to a *High-level architecture* of main components and the required processes and data flows.

To complete the switch to a technical perspective, the design activities for a *Software Architecture* and the implementation

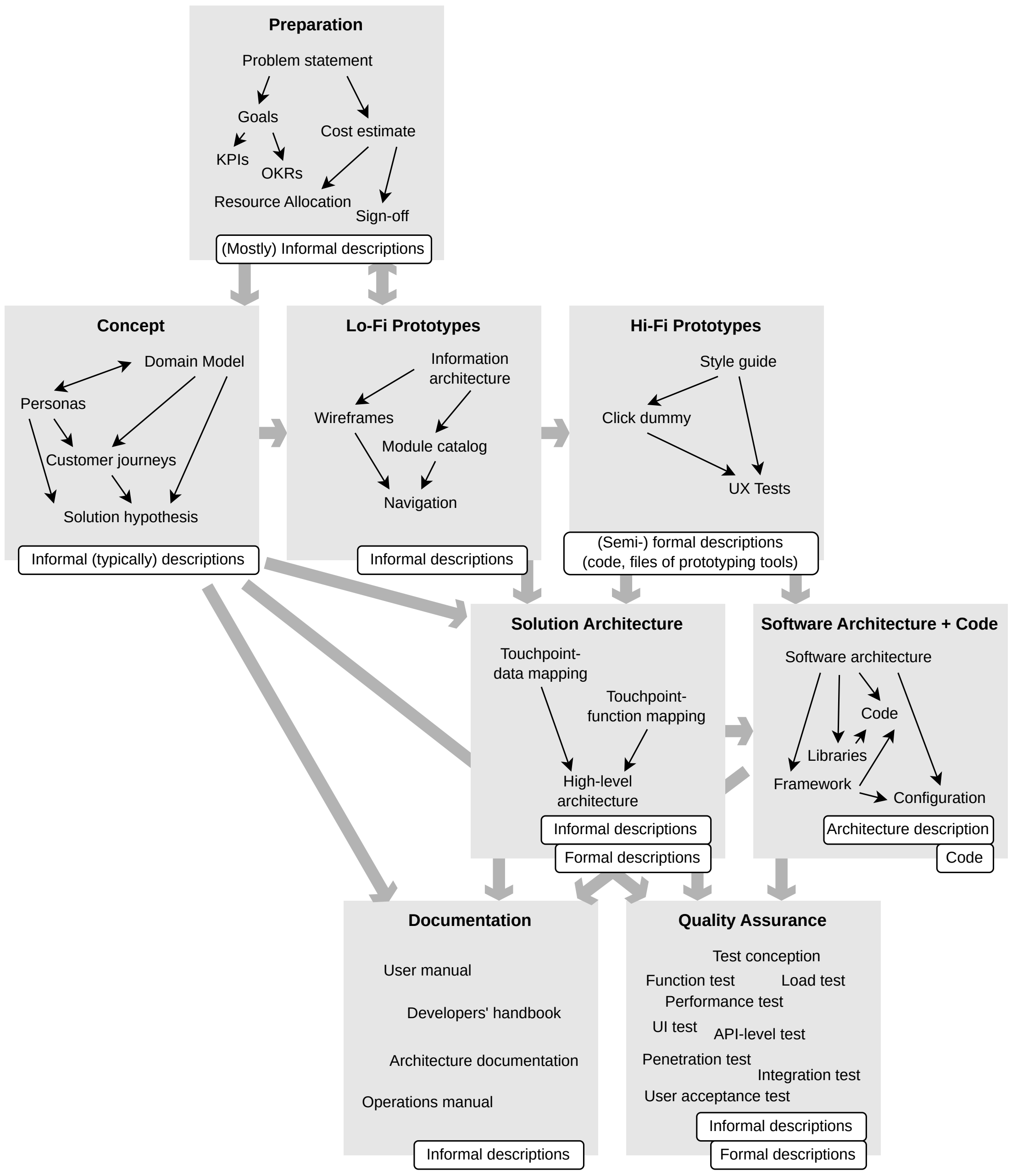


Figure 1. A typical flow of artifacts in a software development process that integrates creative activities, based on [6].

activities leading to *Code* elaborate the models to the point of operable software. There are different definitions of *Software architecture*. In particular, it explains how requirements are adressed in detail. When implementing the software design, *Code* is produced, incorporating *Libraries* and *Frameworks*. Software (also) consists of *Configurations* to a varying degree. The degree depends on the implementation approach chosen: custom development or off-the-shelves software; software that is manually coded, generated, or built in a low code environment.

On top of the software itself, accompanying artifacts such as the various *Documentation* items and *Tests*, more specifically test concepts, test cases, and test scripts, are created as part of a software engineering process.

Models such as the *Domain model*, the *High-level architecture*, and the *Software architecture* can typically be expressed in a suitably formal way as to be derived from each other by model-to-model transformations. However, other documents are typical representatives of informal documents, such as *Personas*, *Customer journeys*, and *Style guides*. There may even be dynamic artifacts, such as a *Click dummy* that needs to be experienced by a human observer who interacts with it.

Informal documents are authored using tools with a focus on graphical presentations. Typical tools are presentation software, collaborative digital whiteboards, issue trackers, and the usual text processors and drawing software. During creative processes, even structured entities, such as user stories and task boards are often represented in unstructured documents like whiteboard drawings.

### C. Creative Artifacts in Model-Driven Processes

Depending on the type of software, there are different steps in the development process that are of an informal nature. Some software solutions require creative development activities. Typical such activities are those from the disciplines of domain modeling, conceptual modeling, and visual design. Such development steps are typically performed manually and lead to subjective results. As a result, tools that support creative activities often produce informal representations and documents. Therefore, software projects that involve creative activities cannot be fully covered by model-driven processes in most approaches.

In order to include creative activities in model-driven processes, the informal documents that are generated have to be interpreted in such a way that their content can be referenced and can be extracted in a defined structure. Through such an interpretation, content may be used in software models or during model transformations.

Interpretations of documents that lack formal structure can be added explicitly. For example, their creators may provide annotations with *content references* and *metadata* to guide access to relevant content. Such annotations, however, refer to specific document instances.

Creative activities typically consist of numerous iterations. As a consequence, documents used in creative activities are subject to constant change. This includes changes to the structure of the documents themselves. Therefore, any fixed reference to content in such documents will potentially become invalid and metadata may become inconsistent as work progresses. As a consequence, documents are required to be constantly reinterpreted.

Due to possible structural changes, the (re-) interpretation of documents must be defined based on some rules or patterns, such as grammars with syntactic and semantic rules.

## III. Referencing Content in Documents

In order to extract content from documents in a form that is suitable for use in a formal development process, parts of that content must be addressable. This requires documents to be structured, or to allow superimposed structures for content references.

Digital documents can be structured to varying degrees. Typically, document formats are categorized as *structured*, *semi-structured*, and *unstructured*.

### A. Structured Documents

Structured documents are created according to a well-defined structure, allowing them to be precisely analyzed. This can be realized in three different ways. First, the structure of documents may be used to query for content, such as object paths based on JSON definitions. Second, specific parts of a document can be addressed if structure elements have stable names (paths) or stable IDs. A third approach uses grammars which can be used to both create documents of a certain form and to parse documents to identify structural elements according to linguistic constructs.

A common structure to which multiple documents conform calls for a schema or document format. Schemas of structured documents differ in the meaning they convey. A format may reflect visual layout like, for example, in the case of HTML, it may use a generic semantics like, for example, XML formats for formal languages, or it may carry domain knowledge as, for example, application-specific XML formats.

### B. Semistructured Documents

Documents that have a recognizable structure, but no common schema to which they conform, are called *semistructured*. Any interpretation rules applied to such documents are fragile in the sense that they may not be applicable to all document instances, or else all possible forms of documents must be considered in all rules.

If there is some technical structure that allows referencing parts of a document, then some pragmatics can be applied to interpret combinations of structure elements and content. For example, in a text document, there may be a recognizable structure of single-line terms written entirely in bold font. That structure may be interpreted as the term being a section heading. If the document is a software architecture description, and if the term is interpreted as a subsection of a section “Software Components”, then the term may be interpreted as the name of a software component.

In this way, semistructured documents are required to expose some recognizable structure. As these examples demonstrate, interpreting them requires some defined domain semantics and pragmatics in order to apply interpretation rules. This includes both domain entities, processes, constraints, etc., as well as typical representations and document layouts that are used in the domain. Note that the term “domain” refers to different levels of abstraction, ranging from a topic domain to a specific project.

### C. Unstructured Documents

Unstructured documents, exhibit no structure that would allow referencing parts of a document. Typical examples of such documents are media files in binary format.

To reference parts of an unstructured document, some technical ways of addressing can be used, for example, pixel ranges in an image or timecode sequences in movies. Such references depend on the concrete document or, more precisely, on the actual presentation of it. For example, areas of an image that are defined by pixel coordinates relate to the resolution of that image. Such references are, therefore, volatile. For example, a selection of pixel coordinates is not valid for an equivalent image in different resolution.

There is no precise way to semantically reference content, although the semantics of unstructured documents can be analyzed by various algorithms.

### D. Aggregating Documents from Different Sources

When accessing document collections that originate from different sources, the problem of different or varying schemas may arise. A typical approach to cope with such a situation is employing adapter components that allow accessing structured documents according to a common schema or by transforming them into a common schema [12].

However, when multiple documents describe the same domain entity, a common schema is not sufficient for precise retrieval of that content [13].

### E. Extracting Content from Mutable Documents

As mentioned earlier, documents created during creative activities in software engineering processes are subject to change, which means they have to be *mutable* (*volatile*, sometimes called *living* documents).

In MDSE processes, the contents of documents are used to create software models from them, or such models are in other ways related to the contents of documents. Changing documents can generally break such relationships.

One solution is to create copies of documents once they are referenced and to keep these copies stable. But this would exclude further work on those documents from the process.

*Parsing* is a standard approach to identifying meaningful content in a document. For formal languages, a parsing process operates on the syntactic structures of a document and applies a defined semantics to interpret those structures.

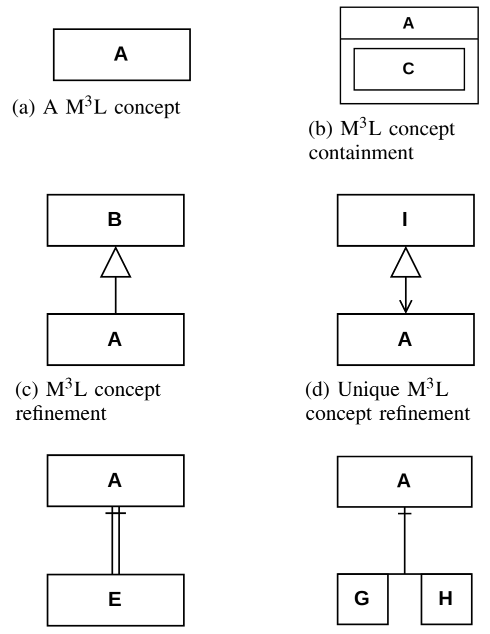
Interpretation is driven by *parsing rules* that are part of a *grammar*. First, these rules are applied to recognize syntactic structures. Once syntactic structures according to a grammar have been detected in a document, a subsequent semantic analysis assigns meaning to those structures. Compilers for programming languages create an *Abstract Syntax Tree* (*AST*) during syntactic analysis and attribute its nodes during semantic analysis to derive a *decorated* AST. Similar techniques can be used for documents.

Documents resulting from creative processes do not follow a fixed semantics. Therefore, classical parsing approaches based on formal languages alone do not work on them. In our current research, we are using domain knowledge to augment document parsing.

Parsing of semistructured documents requires pragmatics since not all parts of a document have an identifiable structure. An open question is whether pragmatics can be provided by domain knowledge: two equally formatted expressions may be distinguished by some significant content. In general, domain knowledge may be necessary to decide on a parsing strategy.

Parsing is well understood for formal and, to a limited extent, semistructured representations, but it is usually applied once. Updating models based on subsequent parsing results of a modified document requires, according to our current findings, an additional relationship between document structure and domain semantics.

Updates can change a document on two levels: If a document's content is changed, then the document needs to be reparsed and reinterpreted. Detected changes lead to updates of subsequent models (documents).



(a) A M$^3$L concept
(b) M$^3$L concept containment
(c) M$^3$L concept refinement
(d) Unique M$^3$L concept refinement
(e) Semantic rules of M$^3$L concepts
(f) Syntactic rules of M$^3$L concepts

Figure 2. A graphical notation of M$^3$L concepts.

If changes affect the structure of a document, then additional actions are required. When the interpretation of structures changes, the underlying parsing rules generally need to be updated because it is impossible to anticipate all possible structures in a grammar.

Apart from classical parsing techniques, GenAI performs document analysis.

## IV. The M$^3$L as a Modeling Language

The *Minimalistic Metamodeling Language* (*M$^3$L*) is a metamodeling language. As such, it can be applied to various modeling tasks. We use the M$^3$L for initial experiments in document recognition, capturing both domain semantics as well as document formatting.

This section provides a short introduction to the M$^3$L to introduce the fundamentals of the modeling approach and to illustrate its application to MDSE through some modeling patterns.

### A. A Short Introduction Into the M$^3$L

The M$^3$L allows defining and deriving *concepts*. Definitions are of the general form

**A** is a **B** { **C** is a **D** } $\models$ **E** { **F** } $\vdash$ **G H** .

Such a statement matches or creates a concept *A*. All parts of such a statement except the concept name are optional.

In the course of this article, we use a graphical notation of the M$^3$L as shown in Figure 2 for the different parts of a concept definition. For concept refinement we borrow notation

```
ConditionalStatement is a Statement {
  Condition     is a Boolean
  ThenStatement is a Statement
  ElseStatement is a Statement
}
```

Figure 3. Sample base model of procedural programming.

```
IfTrueStmt is a ConditionalStatement {
  True is the Condition
} ⊨ ThenStatement

IfFalseStmt is a ConditionalStatement {
  False is the Condition
} ⊨ ElseStatement
```

Figure 4. Sample semantics of conditional statements.

from the *Unified Modeling Language* (*UML*), see Figure 2c for *is a* relationships and Figure 2d for *is the* relationships.

The concept *A* is a *refinement* of the concept *B*. Using the “is the” clause instead defines a concept as the only specialization of its base concept. Conflicting “is the” specializations are considered an error.

The concept *C* is defined in the *context* of *A*; *C* is part of the *content* of *A*. Contexts define (hierarchical) scopes. Concepts, such as *A* are defined in an unnamed top-level context.

There can be multiple statements about a concept visible in a scope. Statements about a concept are cumulated. This allows concepts to be defined differently in different contexts.

For an example of modeling with the the M$^3$L, consider the definition of a conditional statement found in imperative programming languages in Figure 3. It consists of *Condition* to decide whether to execute *ThenStatement* or *ElseStatement*.

*Semantic rules* can be defined on concepts, denoted by “|=”. Figure 2e shows the graphical notation. A semantic rule references another concept that is returned when a concept with a semantic rule is referenced. As with any other reference, a non-existent concept is created on demand.

Context, specializations, and semantic rules are employed for *concept evaluation*. A concept evaluates to the result of its syntactic rule, if defined, otherwise itself, . Syntactic rules are inherited from explicit base concepts (given by *is a* or *is the*) and implicit base concepts (concepts with matching content).

Concept evaluation is used to assign semantics to concepts. The code in Figure 4 uses syntactic rules to assign semantics to the conditional statement from the example above. A concrete statement is matched against the two subconcepts *IfTrueStmt* and *IfFalseStmt*. If one of them is recognized as a derived base concept of the given statement, the semantic rule of the matching concept is inherited. This way, the *ThenStatement* or the *ElseStatement* is determined as the evaluation result of a *ConditionalStatement* and thus a specialization of the respective statement is “executed” (evaluated next).

```
Java is a ProgrammingLanguage {
  ConditionalStatement
    ⊢ if ( Condition ) ThenStatement
      else " " ElseStatement .
}

Python is a ProgrammingLanguage {
  ConditionalStatement
    ⊢ if " " Condition :
      "\n  " ThenStatement
      else:
      "\n  " ElseStatement .
}
```

Figure 5. Sample syntax of the conditional statement.

Concepts can be marshaled/unmarshaled as text by *syntactic rules*, denoted by “|-” or graphically as shown in Figure 2f. A syntactic rule names a sequence of concepts whose representations are concatenated. A concept without a syntactic rule is represented by its name. Syntactic rules are used to represent a concept as a string as well as to create a concept from a string.

Figure 5 shows syntactic rules that map the conditional statement from the example to different programming languages. Chosing the programming language as the context, a syntactic form of the concepts is generated accordingly.

## B. MDSE with the M$^3$L

An MDSE process relies on a series of models where models are created from existing models by means of model-to-model transformations. A model on one stage is created based on the input of models of earlier stages or by refining models from the same stage. This article considers three typical kinds of model transformations:

1) Model combination
2) Model refinement
3) Model creation from existing models

The three model relationships can be used with the M$^3$L to express model transformations. The following examples outline basic modeling approaches.

*a) Combining models:* Domains often rely on base domains. For example, business tasks rely on mathematical models. Accordingly, models are defined by integrating (existing) models of base domains. This way, models are reused.

Let *BaseModel1* and *BaseModel2* be some models of some domains whose concepts can be reused for the domain at hand. Then, for example, concepts *A* and *B* can be integrated into a new model *SomeModel* by definitions like:

```
SomeModel {
  A from BaseModel1
  B from BaseModel2
}
```

For example, on the layer of domain models, the model shown in Figure 6a combines parts of product details that come

```
ProductDescriptions is a DomainModel {
  ProductData
  PaymentMethods        from Commerce
  PackagingInformation  from Logistics
}
```

(a) A sample domain model.

```
OurInfoSys is a PlatformIndependentModel {
  AppServer   from SWComponents
  DBMS        from SWComponents
  DataSchema  from DBModeling
  WebServer   from SWComponents
  WebPage     from WebDesign
}
```

(b) A sample solution architecture.

```
OurInfoSysConcept is an OurInfoSys {
  RDBMS from SWComponents is the DBMS
  ProductDataSchema
    is an  RDBSchema from DBModeling,
       the DataSchema
  WebServer from SWComponents
    is a ServletEngine from Java
}
```

(c) A sample solution architecture refinement.

```
OurInfoSysConcept |= OurInfoSysDataLayer {
 RDBMS
 ProductDataSchema {
  ProductsTable is a Table from DBModeling
 }
}
```

(d) A sample software architecture.

```
OurInfoSysDBIm is an OurInfoSysDataLayer {
 ProductDataSchema {
  ProductsTable ⊢ "PRODUCTS(" Columns ")".
 } ⊢ "CREATE TABLE " ProductsTable .
}
```

(e) A sample software architecture.

Figure 6. Sample flow of software models expressed in the M$^3$L.

from different specialized models (assuming that concepts for models *Commerce* and *Logistics* are given, and that those models combine the named concepts for the data sets).

Likewise, on the layer of solution architecture, the model from Figure 6b combines technical components from different technical descriptions. Here, we assume the availability of a model *SWComponents* that hosts descriptions of typical software components found in the domain at hand, etc.

*b) Refining models:* Within one stage, models are refined to more concrete models of the same stage. This way, the work in each stage starts with first, coarse-grained models, that are then transformed into more concrete models. Different refinements of one model may cover different perspectives on the targeted (software) solution. The process of refining involves decision making. Decisions can be documented by stating delta models that explicitly represent the transformation applied during refinement.

Using the M$^3$L, one model can be created as a refinement of another. Concepts in the content of the refined model are inherited and can be refined further.

An example from the solution architecture layer is shown in Figure 6c. In this example, two aspects of the conceptual model are refined: From a technical perspective, the *DBMS* is more concretely specified to be a relational DBMS (*RDBMS*), and the *WebServer* to be implemented as a Java Servlet engine (*ServletEngine*). Regarding the domain model, it is defined that the data schema is defined to store products (*ProductDataSchema*).

*c) Creating models in subsequent stage:* When processing from one stage to another, initial models are required for the subsequent stage that is entered. Optimally, the most concrete models of the preceding can be transformed to form the initial models of the subsequent stage. If new models have to be created, the model elements should be explicitly linked to the elements from models on which they are based for the sake of traceability. For example, Shaw and Garlan [14] demand for software architecture that solution aspects refer to requirements.

In the M$^3$L, a model can be explicitly created as a transformation of another model using a semantic rule.

Figure 6d continues the example of the information system. *RDMBS* from the source model *OurInfoSysConcept* is re-introduced in the transformed model. The database schema *ProductDataSchema* is additionally redefined by naming one table. *WebServer* from *OurInfoSysConcept* is not considered in the transformed model, since it only models the data layer of the information system.

### C. Software Creation with the M$^3$L

The models in MDSE ultimately reach the stage of generating code. The M$^3$L allows creating code using syntactical rules that can be added to models with sufficient concreteness.

Using the example from above, part of the information system based on a relational database can be defined to create a relational schema by SQL statements as indicated in Figure 6e.

By defining the syntactical rules in the context of an implementation model, different code generation schemes can be defined for one software model. This way, for instance application code, UI code, data models, and data exchange formats can be generated from the same model.

## V. First Experiments Using the M$^3$L

Describing static documents with metadata provided as concepts that make reference to relevant parts of the content has been researched previously in the *Concept-Oriented Content Management* approach [15]. The M$^3$L may offer new way to link abstract concepts with documents. Some initial experiments with simple mutable documents have been conducted to investigate means of linguistic document interpretation.

## A. Static Document References

Figure 7 is an example of a document description using the M$^3$L. It illustrates static references to documents or document fragments. It uses an example from art history as the primary application domain used in the initial studies of document descriptions in the Concept-Oriented Content Management approach. Applications from the humanities were chosen because of the specific need for multifaceted interpretations of documents. Many of the findings also apply to the software domain since the requirements are similar, if not more stringent.

The (digital) picture of a painting at the bottom of Figure 7 is described using (M$^3$L) concepts. An abstract concept hierarchy, beginning with the concept *DocumentReference*, defines references to documents or document fragments. A *DocumentId* defines an address of a document, such as a file name or URL. A *FragmentSelector* defines a part of a document containing interesting content. The example shows a sketch of a refinement hierarchy that specifies concepts for references to two-dimensional images, for those depicting paintings, and paintings showing a ruler specifically.

A second concept hierarchy starts with *DocumentDescription* and contains concepts that describe the subject of a document. The concepts in this hierarchy describe an interpretation of a painting, its content. There is little to no abstraction in general. Instead, the concepts lay a foundation for individual concepts that record observations and interpretations.

The two hierarchies converge at the *PaintingDescription*. The application-specific concept *RulerPaintingDescription* refines it for the area of interest. In this case, it is ruler images that convey a political message. Therefore, there is a reference to a *Ruler* concept describing the ruler that the artwork is about and a fragment selector *RulerDepiction* pointing to the ruler's depiction in the picture. Additional content can be added as needed, such as a description of the artist, the epoch, the creation location, or the exhibitions of the artwork.

Finally, *NapoleonCrossesTheAlps* provides an "instance" of a ruler painting. There are actually multiple paintings of that motive, and even more references to Hannibal in depictions of Alp crossings. The concept *NapoleonCrossesTheAlps* unites multiple views of the (historical) content and the document (painting). It also relates multiple domains, such as history, arts, and political science.

## B. Interpretation of Semistructured Documents

Some general concepts serve as the foundation of the interpretation of certain types of documents. These concepts are used to assign domain semantics to content.

Figure 8 shows an example of documents representing customer journeys that are drawn on and exported from (hypothetical) whiteboard software. This article assumes a hypothetical service because the APIs for accessing board content from actual services differ. However, they typically allow access to graphical shapes. Examples include the APIs of the services Miro [16] and FigJam [17].

The concepts shown Figure 8 in facilitate the interpretation of the graphical components in the UX design domain. Specializations of the concept *Board* allow referencing a whiteboard, and specializations of the concept *Page* allow referencing a page (assuming the whiteboard software allows whiteboards to be subdivided). On a whiteboard, there is no recognizable structure below the page level. Starting with the concept *CustomerJourney*, we look for semantic structures on a whiteboard page. A customer journey is a named (graphical) object that consists of elements that represent *Touchpoint*s and ones that represent *Step*s. A touchpoint is characterized by a *Name* and a *Service*.

Syntactic rules define how these concepts are represented on a whiteboard page. Figure 8a sketches some rules that generate and recognize JSON code as it may be exported from a whiteboard software that is provided as a Cloud service.

*Service* and *Touchpoint* are each represented by one shape. These shapes, and thus these two entities, are reflected in the whiteboard service API. However, compound entities have no counterpart in the API; *CustomerJourney* specializations are defined in M$^3$L concept structures.

Once a customer journey has been developed on a whiteboard of that form, the syntactic rules can be applied to recognize its structure and to extract its content. Content changes, such as renaming an entity, lead to updated concepts. New concepts are created for elements that are added to the board, and they will be added to the customer journey corresponding to a board page.

Figure 8b shows a sample query for customer journeys. The concept for the board is a subconcept of *Board* from Figure 8a. It matches all board specializations that refer to the given board (i.e., the *URL*), have the name *CustomerJourneys*, and contain a page called *CustomerJourney3*. Furthermore, the page must contain a *CustomerJourney* that has one *TouchPoint* with the *Website* service.

If a board matching the query exists, it will be selected. Otherwise, it will be updated accordingly.

When the complete board is interpreted according to the syntactic rules for *Board*s, the result is the concept structure shown in Figure 8c. The concepts that have been created from the board reflect some of the design decisions from the customer journey representation, such as the participating persona and the relationships to the touchpoints it visits and the sequence of touchpoints along the customer journey.

The extracted information can be used in subsequent activities of the software development process. Using the M$^3$L, the resulting concepts can be directly related to concepts that represent models created in such subsequent activities. With the relationships outlined in Section IV-B, the related software models are updated on changes to the board.

## C. Reinterpretation of Mutable Documents

Mutable documents are handled by repeatedly applying the parsing process. When reinterpreting a document after a change, new concept definitions are created in the M$^3$L. Because the M$^3$L matches definitions against existing concepts before creating new ones, previous interpretations are found and used in the parsing process. Depending on the concept model,

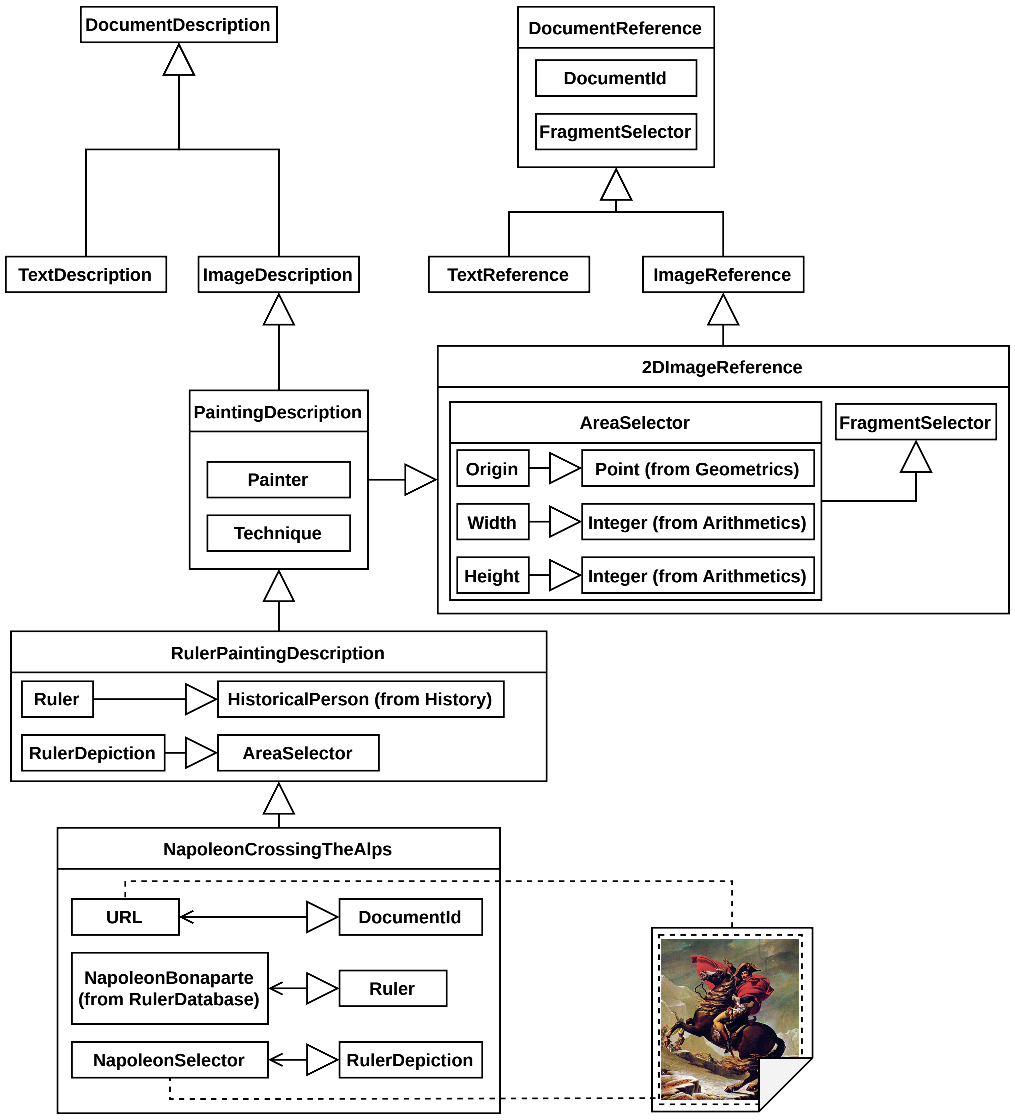


Figure 7. Static references to documents and document fragments.

existing references to concepts in models from subsequent stages that were established by model-to-model transformations are preserved. In this way, documents can be modified even after they have already been interpreted and related to other models during an MDSE process.

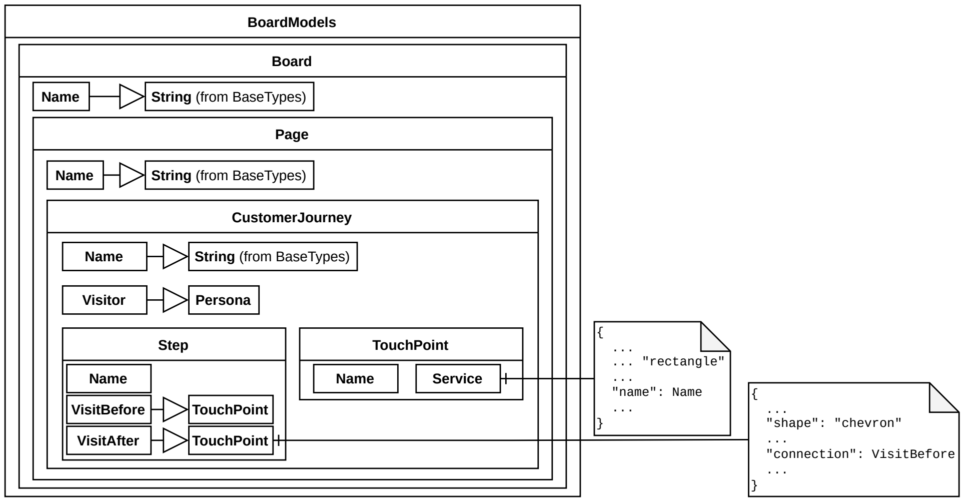
Structural changes can be recognized to a limited extent. As indicated in the previous subsection, newly added shapes lead to newly created concepts. But the correct linking of the concepts, such as the order given by *VisitBefore* and *VisitAfter*, cannot be established using the sample model shown so far. In the example of the customer journey visualization, the order of the service usages might be visualized by the horizontal position of the rectangle shapes. Interpretation requires topological relationships “left of” or “right of”, or the horizontal positions of the shapes have to be interpreted in a pragmatic way.

Alternatively, the order of touchpoint visits may be visualized explicitly by arrows as indicated in Figure 8b. Such a visualization may be harder to analyze, depending on the way arrows are reflected in the board software’s API. Since arrows are expected to be manually drawn in most cases, they will not be represented by curves or splines, making it very hard to analyze them with syntactical rules.

Recognition of existing concepts requires some stable identity information. Otherwise, the associative matching as provided in the M$^3$L might fail to select the right concepts. Such stable identity information may, for example, be unique names as well as a certain location in the document structure where it is placed. In the example of the digital whiteboards above, names might be given in a specially positioned text field. As a consequence, the documents are not completely mutable, at least not in terms of content.

An agreement on some recognizable information constitutes a restriction to the idea of mutable documents. Finding ways of leveraging this situation is subject to future work.

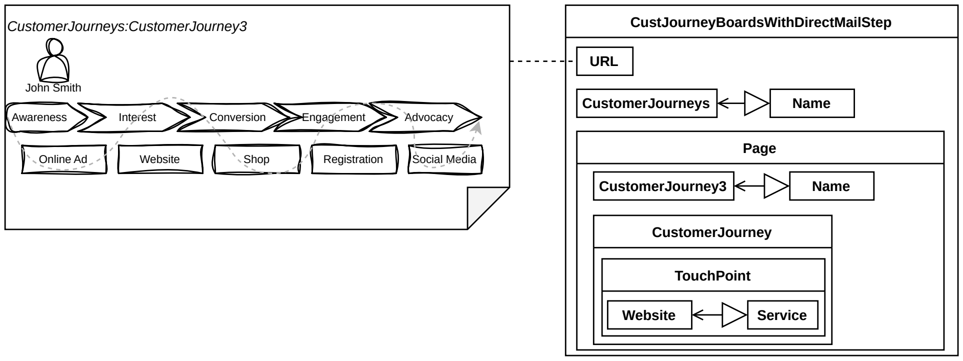
Other structural changes require extensions of the meta model of the current modeling stage. If, for example, a new



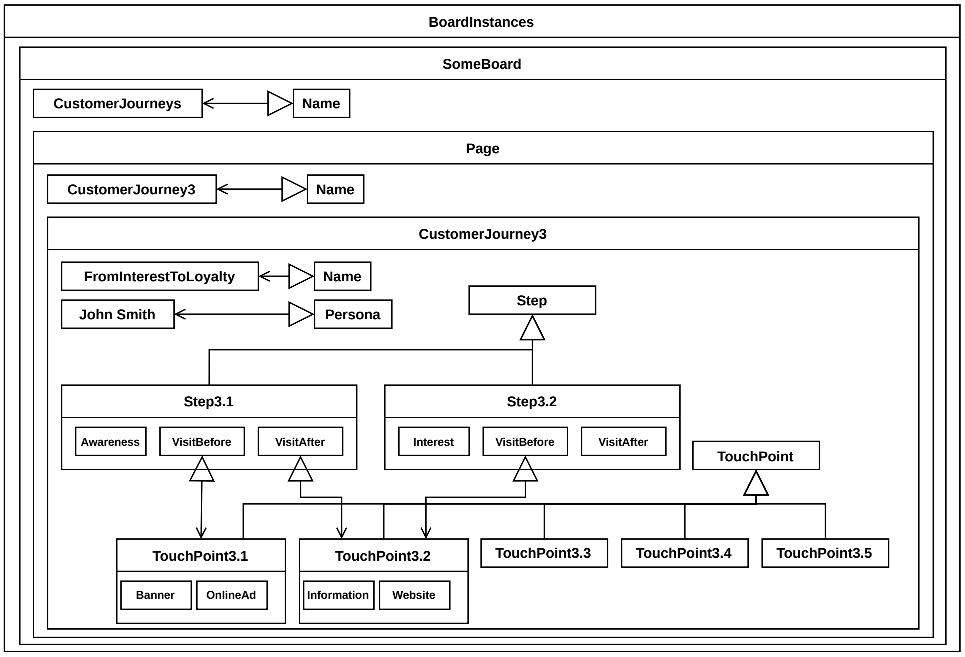
```
{
  ...
  ... "rectangle"
  ...
  "name": Name
  ...
}
```

```
{
  ...
  "shape": "chevron"
  ...
  "connection": VisitBefore
  ...
}
```

(a) Example of a pattern for mutable documents.



(b) Example of a query to mutable documents.



(c) Example result of mutable document recognition.

Figure 8. Parsing of documents and document fragments.

kind of entity is introduced, and if that entity is represented by a new shape in whiteboard drawings, then a new concept for the entity with a syntax rule for its representation has to be added.

By means of model compositions (see Section IV-B), though, models with matching syntax definitions might be looked up from a model repository. Such dynamic extensions are also subject to future research.

## VI. Conclusion and Future Work

In this article, we investigate an approach to integrate semi-structured documents supporting creative activities into MDSE processes. Using the M$^3$L, documents can be parsed based on their syntactic structure in conjunction with the semantics of the concepts represented in such documents. A first simple experiment shows that content can be extracted from a document in a suitably formal form if the document follows some conventions. The concepts recognized in a document can serve as model elements that link the documents to the chain of model-to-model transformations of MDSE processes.

Future work will need to test this approach with a range of existing file formats and service APIs to further investigate the limits of document interpretation and possibly identify additional requirements for parsing technology. There are limits to the extent to which documents can be modified without losing existing links to software models. These limits are not well researched. We need to find the limits, ways to extend them, and notations to describe parts of documents that must not be altered. Another future research direction concerns a form of roundtrip engineering in which documents are not only interpreted, but also generated from models that need to be presented in a form suitable for non-technical stakeholders.

GenAI receives tremendous attention lately. Though not part of the original research, it has to be included in future research. GenAI can support multiple modeling activities, in particular in conjunction with creative artifacts.

Some GenAI applications have already been researched, such as checking completeness of requirements (given in natural language) [18]. After the emerge of “vibe coding” [19], there is an interest in “vibe modeling” as an iterative model transformation approach. It may also help in generating code from models.

Currently, we are looking in low code tools with generic AI support. In the future, it will be interesting to see whether LLMS that have been trained specificially for modeling with the M$^3$L allow automated model transformations by stating a source model and giving delta model instructions.

## Acknowledgement

The author thanks the Nordakademie for granting the opportunity to publish this work.

## References

[1] H.-W. Sehring, “Integrating creative artifacts into software engineering processes”, in *Proceedings of the Seventeenth International Conference on Creative Content Technologies*, ThinkMind, 2025, pp. 1–6.

[2] D. Schmidt, “Guest editor’s introduction: Model-driven engineering”, *IEEE Computer*, vol. 39, no. 2, pp. 25–31, 2006.

[3] G. Liebel et al., “Human factors in model-driven engineering: Future research goals and initiatives for mde”, *Software and Systems Modeling*, vol. 23, no. 4, pp. 801–819, 2024.

[4] E. Herac, L. Marchezan, W. Assunção, R. Haas, and A. Egyed, “A flexible operation-based infrastructure for collaborative model-driven engineering”, in *Modellierung 2024*, ser. Lecture Notes in Informatics, Gesellschaft für Informatik e.V., 2024.

[5] I. Galvao and A. Goknil, “Survey of traceability approaches in model-driven engineering”, in *Proceedings of the 11th IEEE International Enterprise Distributed Object Computing Conference*, 2007, pp. 313–313.

[6] H.-W. Sehring, “Visual artifacts in software engineering processes”, in *Proceedings of the Sixteenth International Conference on Creative Content Technologies*, ThinkMind, 2024, pp. 1–6.

[7] S. Trujillo, M. Azanza, and O. Diaz, “Generative metaprogramming”, in *Proceedings of the 6th international conference on Generative programming and component engineering GPCE ’07*, Association for Computing Machinery, 2007, pp. 105–114.

[8] J. Arnoldus, M. Van den Brand, A. Serebrenik, and J. J. Brunekreef, *Code generation with templates*. Springer Science & Business Media, 2012, vol. 1.

[9] K. Lano and Q. Xue, “Code generation by example using symbolic machine learning”, *SN Computer Science*, vol. 4, Jan. 2023.

[10] W. Ding, X. Lin, and M. Zarro, *Information Architecture and UX Design: The Integration of Information Spaces*. Springer Cham, 2025.

[11] H.-W. Sehring, “Model-supported software creation: Towards holistic model-driven software engineering”, in *Proceedings of the 2023 IARIA Annual Congress on Frontiers in Science, Technology, Services, and Applications*, ThinkMind, 2023, pp. 113–118.

[12] I. Amous, A. Jedidi, and F. Sèdes, “A contribution to multimedia document modeling and querying”, *Multimedia Tools and Applications*, vol. 25, pp. 391–404, 3 Oct. 2005.

[13] A. Roy, K. Ghosh, M. Basu, P. Gupta, and S. Ghosh, “Retrieving information from multiple sources”, in *Companion Proceedings of The Web Conference 2018*, International World Wide Web Conferences Steering Committee, 2018, pp. 43–44.

[14] M. Shaw and D. Garlan, “Formulations and formalisms in software architecture”, in *Computer Science Today: Recent Trends and Developments* (Lecture Notes in Computer Science), Lecture Notes in Computer Science. Springer, 1995, vol. 1000, pp. 307–323.

[15] J. W. Schmidt and H.-W. Sehring, “Conceptual content modeling and management”, in *Perspectives of System Informatics*, Springer, 2003, pp. 469–493.

[16] Miro, “Get specific item on board”, 2025. [Online]. Available: https://developers.miro.com/reference/get-specific-item.

[17] Figma, “Node types”, 2025. [Online]. Available: https://www.figma.com/plugin-docs/api/nodes.

[18] D. Luitel, S. Hassani, and M. Sabetzadeh, “Improving requirements completeness: Automated assistance through large language models”, *Requirements Engineering*, vol. 29, no. 1, pp. 73–95, 2024.

[19] A. Gadde, “Democratizing software engineering through generative ai and vibe coding: The evolution of no-code development”, *Journal of Computer Science and Technology Studies*, vol. 7, no. 4, pp. 556–572, 2025.

# The Kosmosis Approach to Crypto Rug Pull Detection

Philipp Stangl* and Christoph P. Neumann†

*Department of Computer Science
Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany
e-mail: philipp.stangl@fau.de
†Department of Electrical Engineering, Media and Computer Science
Ostbayerische Technische Hochschule Amberg-Weiden, Amberg, Germany
e-mail: c.neumann@oth-aw.de

*Abstract*—**Crypto rug pulls have become a major threat to the integrity of blockchain ecosystems, with illicit activities surging and resulting in significant financial losses. Existing approaches to detect crypto asset fraud are based on the analysis of transaction graphs within blockchain networks. While effective for identifying transaction patterns indicative of fraud, existing approaches do not capture the semantics of transactions and are constrained to blockchain data. Consequently, preventive methods based on transaction graphs are inherently limited. In response to these limitations, we propose the Kosmosis approach, which aims to incrementally construct a knowledge graph as new blockchain and social media data become available. During construction, it aims to extract the semantics of transactions and connects blockchain addresses to their real-world entities by fusing blockchain and social media data in a knowledge graph. This enables novel preventive methods against rug pulls as a form of crypto asset fraud. To demonstrate the effectiveness and practical applicability of the Kosmosis approach, we examine a series of real-world rug pulls. Through this case, we illustrate how Kosmosis can aid in identifying such fraudulent activities by leveraging the insights from the constructed knowledge graph.**

***Keywords*-blockchain; cyber fraud; rug pull; security; smart contracts; knowledge graphs; discovery; pseudonymity; untraceability.**

## I. Introduction

This article is a revised and extended version of [1] and [2]. Crypto assets are digital assets that use distributed ledger technology, such as blockchain, to prove ownership and maintain a decentralized and public ledger of all transactions. Cryptocurrencies, like Bitcoin [3], function as digital currencies and are used for storing or transferring monetary value. Fungible Tokens (FTs), another type of crypto asset, are interchangeable units representing various utilities or assets within a blockchain ecosystem. Lastly, Non-Fungible Tokens (NFTs) are unique digital assets that prove ownership and authenticity of digital or real-world assets [4]. In the rapidly evolving landscape of crypto assets, the incidence of illicit activities has surged. Chainalysis, a leading blockchain analytics firm, reported that illicit transaction volume rose for the second consecutive year in 2022, reaching an all-time high of $20.6 billion in illicit activity [5]. Since the rise of Decentralized Finance (DeFi) in 2020, followed by NFTs in 2021, rug pulls have become a major fraud scheme in terms of amount stolen and frequency [6]. Thus, rug pulls pose a significant risk to investors and undermine the integrity of the crypto asset sector.

The predominant approach for identifying patterns indicative of fraudulent activity is the transaction graph analysis within blockchain networks [7]–[9]. However, this approach presents two key challenges. Firstly, the transacting parties are pseudonymous and only their blockchain addresses are publicly known. This means that, although the transactions of a specific address can be tracked, linking that address to a real-world entity can be challenging since this approach is limited to information or patterns observable in blockchain data. Secondly, this approach is only concerned with the following aspects of a transaction: 1) The transferred asset, 2) the quantity, and 3) the sender and receiver. However, the semantics of a transaction, such as what happened in a transaction that caused the assets to get transferred, is not covered. Thereby limiting the depth of analysis that can be conducted on crypto asset movements.

Knowledge Graphs (KGs) [10] are increasingly recognized as a powerful means to integrate fragmented knowledge from heterogeneous data sources into a graph of data to facilitate semantic querying (e.g., [11][12]) and reasoning (e.g., [13]). A KG provides a holistic view for identifying patterns and hidden connections indicative of fraudulent activities in a highly connected dataset [14]. The KG consists of semantically described entities, each with a unique identifier, and relations among those entities using an ontological representation [15][16]. Their open world assumption allows for the continual integration of new data. By leveraging these capabilities, KGs can enhance crypto asset fraud analysis and aid in predicting future fraudulent activities.

The remainder of this article is organized as follows. We first outline the Kosmosis objectives in Section II. Next, we provide a background on the Ethereum blockchain and graph-based blockchain data mining methods in Section III. Subsequently, in Section IV we propose Kosmosis, our incremental KG construction pipeline. In Section V we provide essential background on rug pulls before we demonstrate the effectiveness and practical applicability of the Kosmosis approach for the use case of NFT rug pull detection in Section VI. Finally, we outline future work in Section VII and conclude the article with a discussion of our findings in Section VIII.

## II. Overarching Objectives of Kosmosis

This section outlines the objectives of Kosmosis, beginning with the primary objective that investigates the potential of a KG in identifying and alerting users before they interact with projects linked to known scammers, addressing a critical need for security and trust in blockchain ecosystems. Following that, we explore the technical implications.

**Objective 1: How can the KG identify and aid in alerting users before interacting with a rug pull project?**

With the rise in illicit activities in the crypto asset market, especially rug pulls, there is a pressing need for effective means to detect and prevent fraudulent activities. Kosmosis aims to integrate fragmented knowledge from blockchains like Ethereum, social media like 𝕏, and potentially other knowledge graphs into one unified KG, enabling semantic querying and reasoning over a graph of entities and the relationships among them. The KG could serve as a knowledge base for a real-time alerting system, warning users of potential risks associated with certain projects or individuals.

**Objective 2: How to incrementally construct the KG from heterogeneous data sources?**

It is imperative to establish a pipeline capable of integrating updates into the KG in both batch- and streaming-like manner. Thereby, maintaining high data freshness by ensuring that the KG consistently reflects the most up-to-date information from the blockchain and other sources. This approach should not entail a complete reconstruction of the KG, but rather concentrate on integrating new information, avoiding the reprocessing of data that is already incorporated.

**Objective 3: How to extract the semantics of blockchain transactions?**

Transaction graphs commonly only display transactions with asset transfers and provide answers to questions such as “what” assets were transferred, and “where” were they transferred to. Understanding transactions semantically is vital in uncovering sophisticated fraudulent schemes that might otherwise go unnoticed. Kosmosis addresses this gap by extracting the semantics of transactions, providing answers to “why” and “how” assets were transferred in a transaction. This extraction of semantic information is primarily achieved through decoding the input data of a transaction using the Application Binary Interface (ABI) of smart contracts a transaction interacts with.

## III. Background

In this section we provide background on blockchain technology, specifically the Ethereum blockchain, in Section III-A. Subsequently, we outline related graph-based blockchain data mining methods in Section III-B. On the social media aspects of Kosmosis, our prior work includes correlating Reddit data with traditional stock market trends [18] and analyzing Twitter/𝕏 data using SPARQL [19].

### *A. The Ethereum Blockchain*

Blockchain technology is based on the principles of immutability, decentralization, transparency, and cryptographic security and has seen various applications in recent years. For instance, in the financial sector (e. g., [3][20]), or supply chain management (e. g., using a single blockchain [21], or using multiple, interoperable blockchains [22][23]). Smart contract platforms represent a subset of blockchains that enable the development of decentralized applications through smart contracts. This section outlines the key concepts of Ethereum, as an example for smart contract platforms, that are essential for the following sections of this work, such as smart contracts, their execution environment, and account-based accounting.

*1) Blockchain Data Structure:* A blockchain is a data structure whose elements called blocks are linked together to form a chain of blocks [17], depicted in Figure 1. Each block comprises two parts: a body and a header. The body of the block contains a set of transactions. A transaction typically involves the transfer of assets between a sender and a receiver. These participants are represented by addresses, which are unique alphanumeric strings that clearly specify the origin and destination of each transaction. Further, the block body is used to generate a unique identifier called the block hash. The block header contains a reference to the unique identifier of its immediate predecessor, known as the parent block.

*2) Smart Contracts:* Through smart contracts, which are executable source codes that enforce the terms and conditions of particular agreements, a smart contract platform like Ethereum facilitates the development of decentralized applications [25]. Once deployed on the blockchain, the smart contract is assigned an address where the code resides and cannot be altered or tampered with. By writing custom smart contracts, developers

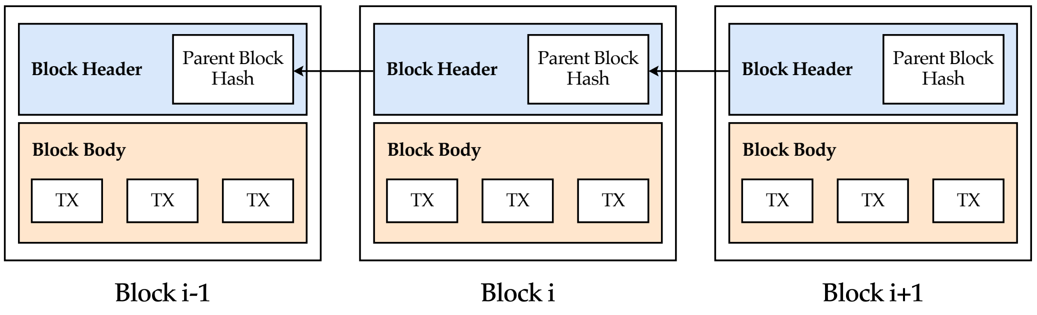


Figure 1. Schematic representation of the blockchain data structure. Adapted from Zheng *et al.* [17].

can create and manage tokens that adhere to the standards ERC-20 for FTs [26] or ERC-721 for NFTs [27]. An ABI specifies the functions and data structures exposed by a smart contract, allowing external applications to understand the capabilities of the contract. Further, an ABI defines a format for encoding and decoding data that is passed between smart contracts and external applications. This ensures a consistent and standardized way to exchange information.

The Ethereum blockchain manages Ether (ETH) as the native cryptocurrency of the platform. It operates with the Ethereum Virtual Machine (EVM) as a fundamental building block, serving as the execution environment for smart contract code. Smart contracts, primarily written in a high-level language such as Solidity, undergo compilation into EVM bytecode. This bytecode is the executable format used by the EVM to enact smart contract functions. To interact with this bytecode, a contract ABI is utilized, which acts as a bridge between the high-level language and the low-level bytecode. In this context, an EVM disassembler plays a crucial role; it reverses the bytecode back into a more readable format, aiding developers in understanding and analyzing the code deployed on the Ethereum blockchain. Figure 2 shows the processes involved in deploying smart contracts to the Ethereum blockchain and reading contract data from it, including compilation and deployment steps, and the interaction between a web application and the Ethereum blockchain. The left side shows the compilation and deployment of a smart contract, and the right side depicts an interaction with the contract (e.g., from a web application).

*3) Externally Owned Account:* Unlike smart contracts, Externally Owned Accounts (EOAs) are controlled by real-world entities through private keys, enabling them to initiate transactions, such as transferring crypto assets or executing functions of a smart contract. When an EOA sends a transaction to a smart contract, it triggers the code of the contract to execute according to its predefined rules.

*4) Account-based Accounting:* For the record-keeping of transactions, blockchains utilize an accounting model. Compared to other blockchains, such as the equally well-known Bitcoin [3] blockchain that uses the Unspent Transaction Output (UTXO) model, or its successor the extended UTXO [28] utilized by the Cardano [29] blockchain, Ethereum [20] employs the account-based accounting model. The account-based model can be best understood through the analogy of a bank account. This approach mirrors how a banking account operates. Like a bank account that tracks the inflow and outflow of funds, thereby reflecting the current balance, the account-based model in Ethereum maintains a state that records the balance of Ether. Thus, it is inherently stateful. Each transaction results in a direct adjustment to this balance, akin to a deposit or withdrawal in a bank account. This model's stateful nature ensures that at any given moment, the system can accurately reflect the total amount of Ether held in each account, offering an up-to-date view of account balances within Ethereum.

*5) Token Minting:* The process of creating new tokens is called token minting. FTs are typically minted by the creator either at the inception of the project or progressively over time. This process is often governed by predefined rules or algorithms embedded within the smart contracts of the project.

In contrast, NFT minting involves other individuals besides the token creator, commonly termed as token minters. They engage by invoking a specific function within a smart contract, in the ERC-721 token standard, called mint. This action results in an increase in the supply of the NFTs and simultaneously assigns these minted tokens to the blockchain address of the minter. The mechanism of minting NFTs often involves utilizing a dedicated minting website. Here, prospective minters or investors are required to invest a predetermined amount, as set by the creator, to initiate the minting process. This investment grants them the ability to mint one or multiple NFTs, depending on the terms set forth in the smart contract.

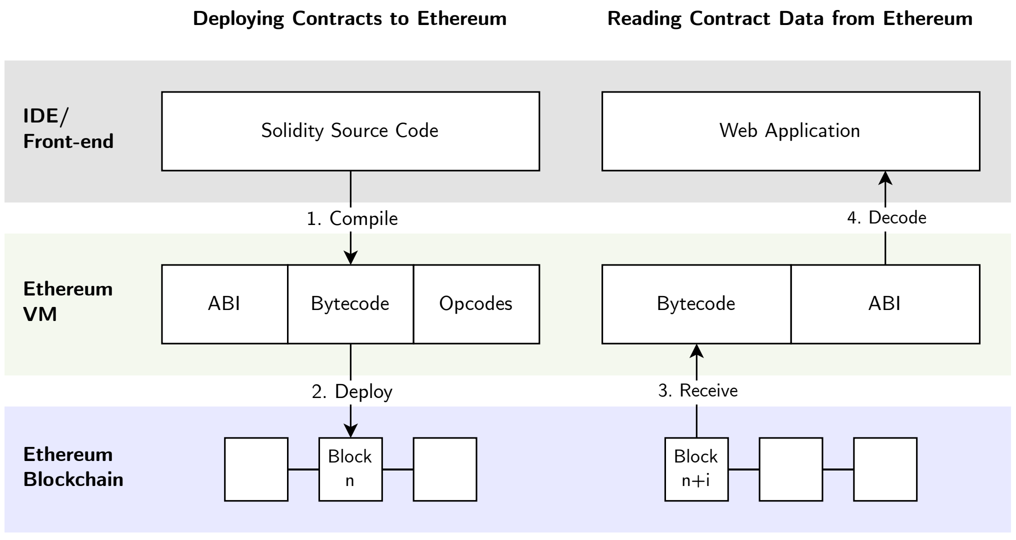


Figure 2. Schematic representation of deploying and reading from smart contracts. Adapted from Takeuchi [24].

### *B. Rug Pull Detection Methods*

Two primary methods have been employed in the past to detect rug pulls: smart contract code analysis and graph-based methods. Smart contract code analysis involves a thorough examination of the contract's code to extract and analyze the semantic behavior of transactions. For instance, [30] utilizes smart contract code analysis to reveal potential vulnerabilities and fraudulent patterns within the contracts. By dissecting the code, their proposed method, dubbed "Tokeer", can identify suspicious patterns and functions that might indicate a predisposition to rug pull scams. Another prominent line of research in smart contract analysis leverages machine learning–based techniques. Mazorra *et al.* [42], for example, employed the XGBoost algorithm as a primary classifier to predict the probability that a liquidity pool will evolve into a rug pull or scam, achieving an accuracy of up to 99.36% using features derived from token propagation patterns and smart contract heuristics. Their dataset and experimental design are restricted to fungible tokens launched exclusively on Uniswap (Ethereum), and the authors argue that directly transferring these learned models to other blockchains is unlikely to yield results of comparable quality. Graph theories and graph-based data mining methods, are applicable for discovering information in blockchain network graphs, because blockchain transactions can be inherently structured into graphs [9]. Elmougy and Liu [31] identified three types of graphs, applicable to any blockchain network: *money flow transaction graphs* visualize the asset flow over time, *address-transaction graphs* showing flow of an asset across transactions and addresses, and *user entity graphs* that clusters the graph for potential linking of addresses controlled by the same user, to deanonymize their identity and purpose. To detect rug pulls with high accuracy, graph-based approaches use network embedding methods to automatically extract features from the blockchain network (e. g., [32]) using a *graph convolutional network*.

## IV. The Kosmosis Approach to Incremental Knowledge Graph Construction

To incrementally construct a KG that integrates data in a continuous and periodic way, we propose a multi-stage pipeline, as illustrated in Figure 3. It originated from a master's thesis [33] and consists of three stages: Data ingestion, data processing, and knowledge storage. We use italics to emphasize on conceptual aspects and typewriter text for technical operations.

The initial stage, data ingestion, captures the raw data from the primary data sources (blockchain and social media) as well as enrichment data sources (e.g., another knowledge base). This phase is characterized by its versatility in the frequency of data acquisition: it can be 1) *continuous*, to capture real-time updates from sources such as blockchain nodes, 2) *incremental* for new posts via the 𝕏 Streaming Application Programming Interface (API), 3) *periodic*, to capture new entries in structured data sources like relational databases at regular intervals, or 4) *event-based*, responding to events that are emitted upon new entity additions to the KG.

Following the ingestion stage, the data processing stage is initiated, which is partitioned into distinct workflows tailored to handle each type of ingested data. This segmentation allows for specialized processing depending on the structure of the raw data. For instance, for text data sources, natural language processing techniques, such as Named Entity Recognition [34], can be used to ensure that the data is accurately interpreted, and contextual relationships are discerned.

In the third and final stage, the refined data is loaded into the knowledge storage, where it is systematically organized within a triplestore, a type of database optimized for storing and retrieving data in Resource Description Framework (RDF) format. The triplestore can then be used for semantic querying capabilities to extract actionable insights from the KG for downstream processes. For the KG, we use the EthOn [35] ontology that formalizes the concepts and relations within the domain of the Ethereum network and blockchain. EthOn is written in RDF and Web Ontology Language (OWL).

### *A. Blockchain Data Processing*

The blockchain data processing workflow continuously ingests new transactions from the blockchain via websocket connections. Websockets enable open, interactive communication sessions between a client and a server, facilitating real-time data transfer without the need for repeated polling. Upon receiving these transactions, the workflow processes and integrates them into the KG by first extracting the address relationship, followed by tagging the addresses, and finally fusing the addresses with the entities of the KG.

*1) Address Relation Extraction:* In order to provide answers to "why" and "how" assets were transferred in a transaction, Kosmosis implements a pipeline module titled *Address Relation Extraction*. The responsibility of this module is to extract the semantic information in a blockchain transaction through decoding the input data of a transaction using the ABI of the smart contract a blockchain address is interacting with.

First, the ABI is requested from Etherscan [36] and Sourcify [37] via their respective REST APIs. If the ABI cannot be successfully fetched from one of the aforementioned sources, the module resorts to reconstructing the ABI from the smart contract byte code, which is available at any time since the bytecode is deployed on the blockchain. This operation enables the decoding of transactions and the interaction with smart contracts beyond their compiled state.

The initial step involves the disassembly of the bytecode of the smart contract. This operation, referred to as `DISASM`, decomposes the bytecode into a series of readable opcodes and associated data. Disassemblers (e.g., *pyevmasm* [38]) facilitate this step by translating the bytecode back into a form that represents the original instructions and operations defined within the smart contract.

Following disassembly, the algorithm initializes by creating an empty array intended to store the ABI and defining lists of opcodes that either change the state or read from the state of the blockchain. These opcodes include `SSTORE`, `CREATE`, `CREATE2` for state-changing operations, and `SLOAD` for state-reading operations,

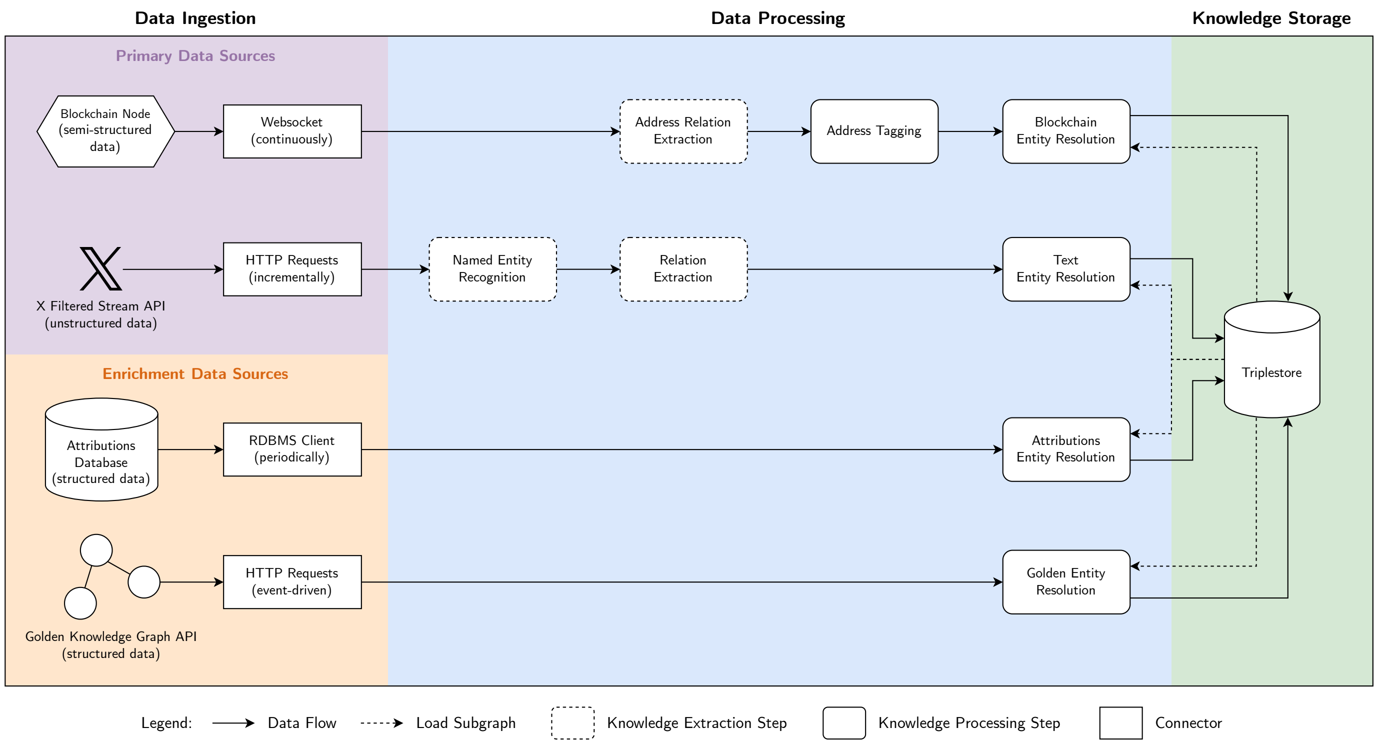

Data Ingestion
Data Processing
Knowledge Storage
Primary Data Sources
Blockchain Node (semi-structured data)
Websocket (continuously)
Address Relation Extraction
Address Tagging
Blockchain Entity Resolution
X Filtered Stream API (unstructured data)
HTTP Requests (incrementally)
Named Entity Recognition
Relation Extraction
Text Entity Resolution
Triplestore
Enrichment Data Sources
Attributions Database (structured data)
RDBMS Client (periodically)
Attributions Entity Resolution
Golden Knowledge Graph API (structured data)
HTTP Requests (event-driven)
Golden Entity Resolution
Legend:
Data Flow
Load Subgraph
Knowledge Extraction Step
Knowledge Processing Step
Connector

Figure 3. A high level overview of the Kosmosis pipeline.

reflecting the fundamental actions a smart contract on the EVM can perform [20].

The core of the algorithm iterates over selector/offset pairs within the disassembled bytecode. Selectors serve as identifiers for functions in the EVM, facilitating the mapping to the corresponding functionality. If a given offset does not match any destination within the program's destinations, the iteration skips to the next pair, ensuring only valid functions are considered.

Upon finding a valid function destination, the algorithm retrieves the function definition and assigns tags based on its behavior. This tagging process involves analyzing the opcodes contained within the function and any related jump destinations. The purpose is to categorize functions according to how they alter the blockchain state, using a depth-first search algorithm to navigate through the function call graph.

An `AbiFunction` object is then created for each valid function, with its payable status determined inversely by the presence of a notPayable marker at the corresponding offset. The algorithm next assigns mutability attributes (nonpayable, payable, view, or pure) based on whether the function alters state, reads state, or neither. This classification is crucial for understanding how functions interact with the blockchain and their implications on transaction costs and permissions.

Finally, the algorithm decides on the inclusion of inputs and outputs in the function signature, informed by the presence of specific tags. For instance, tags indicating data retrieval or state mutation influence whether parameters are classified as inputs or outputs. This granular control ensures that the ABI accurately reflects the interface of the smart contract, allowing for effective transaction decoding.

Currently, the method for extracting semantic information from smart contract transactions relies partly on predefined heuristics, such as recognizing specific function names like "mint." However, we acknowledge that scammers could circumvent these simplistic heuristics by obfuscating or renaming functions. Future improvements will incorporate advanced transaction pattern analysis rather than function naming alone, enhancing resilience against simple obfuscation techniques.

*2) Address Tagging:* Since the exact identity of a real-world entity controlling a blockchain address is often unknown, it can still be categorized and tagged accordingly. The *address tagging module* tags the sender and receiver address based on their extracted relationship from the preceding address relation extraction module. For instance, an EOA deploying a smart contract is tagged as deployer in case of a contract creation transaction. Likewise, if an EOA is sending Ether to an NFT contract $T$ via a contract function containing the word "mint," the EOA is tagged as is tagged as NFT minter of $T$. Tags are subclasses of EOAs and contract accounts, extending the address concept of the EthOn ontology.

*3) Blockchain Entity Resolution:* The blockchain entity resolution module is responsible for resolving blockchain addresses to either new entities or existing ones in the KG, by using the extracted information from preceding steps. It begins with mapping the result data from the preceding steps into the RDF format, adhering to the ontology defined by the KG. This ensures that the data is structured in a way that is compatible with the KG's existing schema.

Following the mapping to RDF, the next phase involves fusing this RDF data with the KG. This is accomplished through a two-step process. Initially, a subgraph that is relevant to the processed data is loaded into the system. This step, commonly referred to as "blocking," narrows down the scope of the resolution process to the most relevant segments of the KG, thereby enhancing the entity resolution process.

Subsequently, the system proceeds to match the newly processed data with the corresponding entities within the KG. This matching process is crucial for identifying where the new data fits within the existing structure and for ensuring that it is integrated in a meaningful way. In certain cases, the fusion process may also involve the clustering of entities. This is particularly relevant for blockchain data, where unique characteristics of the data can be leveraged to enhance the integration process.

For instance, when dealing with blockchains that utilize an account-based accounting model, address clustering heuristics can be employed to further refine the fusion process. One such heuristic is the deposit address reuse, as proposed by Victor [39]. Kosmosis uses deposit address reuse for blockchain data from Ethereum to resolve entities more effectively.

### B. Text Processing

The workflow starts with the input of unstructured data from the 𝕏 Filtered Stream API [40], which is incrementally streamed and parsed via a long-lived HTTP request into the pipeline. The first step in processing this data is Named Entity Recognition, where the system identifies and classifies named entities present in the text into predefined categories, such as the names of persons, organizations, and locations.

The next step is relation extraction. This process involves identifying and extracting relationships between the named entities that were previously recognized. For instance, it could determine that a person named "Alice" works for a company named "Acme."

The final step in the text processing workflow is the entity resolution, achieved through blocking and matching. For each new entity, the system identifies all other entities within the KG that need to be considered for matching. Considering the growing size of the KG, through the incremental updates, it is important to limit the matching process to as few candidates as possible [16]. The method of limiting candidates is known as blocking, which confines the matching process to entities of the same or most similar entity type.

Following the blocking that serves as a preliminary filtering step, the matching is performed. This involves a pairwise comparison of the new entities with those existing entities in the KG identified during the blocking phase. Its objective is to identify all entities that are sufficiently similar and, therefore, potential candidates for matching. This pairwise comparison relies on a nuanced assessment of similarity that encompasses both the properties of the entities and their relational connections within the KG. By evaluating both property values and the nature of relationships to other entities, the system determines the degree of similarity between entities.

### C. Enrichment Data Processing

Enrichment data enhances the data obtained from primary data sources with supplementary context regarding real-world entities. Attributions involve the mapping of blockchain addresses to their corresponding real-world entities. This task is largely dependent on data sourced from a network of experts, such as team members from blockchain projects. The input data for the attribution process is typically not consistent in its timing, as it depends on when the experts provide updates or when new information becomes available. As a result, the enrichment data processing workflow is designed to operate at regular intervals, ensuring that the KG is updated systematically and remains as up-to-date as possible.

To further enrich the KG, data from external knowledge bases is integrated. In our case, we use the *Golden Knowledge Graph* due to its concentrated information on tech startups and cryptocurrencies. This external graph offers a wealth of information about crypto projects, including details about their founders, team members, and project descriptions. Such depth of data provides a valuable context that can significantly improve the understanding of entities in the constructed KG.

The workflow for integrating knowledge from an external KG is event-driven, activated once the knowledge storage indicates the addition of new entities from the social media platform 𝕏. Then, the workflow triggers a process to pull in additional background information from the Golden Enrichment API [41]. It uses the 𝕏 username that has been newly included in the KG as unique identifier to fetch relevant data.

### D. Quantity Structure of the Knowledge Graph Data

In our prototype implementation, data was ingested at rates averaging 10-15 transactions per second (each averaging 5KB) from Ethereum blockchain nodes and roughly 200 tweets per minute (each averaging 2KB) from the 𝕏 filtered stream API. This combined ingestion rate corresponds approximately between 3.4 to 4.9 MB per minute of raw data. Our prototype runs on a standalone cloud server instance with 32 GiB RAM and 8 vCPUs (AWS EC2 m5.2xlarge) with a 512GB SSD, managing real-time data ingestion and processing workloads. The semantic enrichment introduces minimal latency (less than 5 seconds per transaction batch), thus allowing for near-real-time KG updates. The KG constructed by Kosmosis accumulates triples at an approximate rate of 2.5 to 6 million triples per day, depending on transaction activity and the level of detail extracted from social media.

While the described hardware configuration proved adequate for prototype-level or small- to medium-scale deployments, a production implementation aimed at analyzing multiple blockchain networks or higher data volumes would necessitate scaling to multiple compute nodes, each handling dedicated tasks such as blockchain data ingestion.

## V. Rug Pulls and Serial Fraudsters

A rug pull can be categorized as a scam, i. e., the victim authorizes the transaction. This type of scam is typically carried out in five stages, according to [6]: (1) Project creation with roadmap and total supply of tokens (optional), (2) pre-mint hype, (3) set token mint price, (4) token mint, accumulation of more capital and increase in popularity, and finally (5) the creators cash out, abandon the project, and leave the investors defrauded. To attract users and investments for rug pulls, Sharma *et al.* [6] suggest the involvement of individuals or groups that possess substantial technical skills and knowledge of blockchain technology and demonstrate a proficiency in marketing techniques. This specific use case is particularly relevant given the findings in [6] and [42]: Mazorra *et al.*, who analyzed ERC-20 tokens listed on decentralized exchanges in their 2022 study, labeled 97.7% out of 27,588 analyzed tokens as rug pulls [42]. Likewise, Sharma *et al.* analyzed NFTs and identified a cluster of 168 NFTs associated with what they termed the "Rug-Pull Mafia," a group of creators responsible for orchestrating multiple and repeated rug pulls [6]. There is a growing trend in both the frequency and the financial impact of crypto rug pulls and scams [43]. Notably, the year 2021 marks a peak in the amount stolen, while 2022 shows a sharp rise in the frequency of these fraudulent activities and remains elevated since.

## VI. The Use Case of Rug Pull Prevention

To illustrate the vision of Kosmosis-enabled rug pull prevention methods, this section introduces a hypothetical user story centered around a character we name Bob, a crypto market participant. The story telling method of use case illustration was adopted from our previous work in [44, pp. 207–209]. The Kosmosis user story is designed to provide a relatable perspective on how individuals like Bob are affected by such fraudulent activities. The fictional story of Bob is grounded in a series of real-world rug pulls that took place in 2021. All rug pulls were carried out by the same fraudulent NFT creator and Twitter user known as Homer_eth. In Section VI-C, we outline how the series of rug pulls experienced by Bob might have unfolded differently had he been equipped with a Kosmosis-enabled fraud prevention mechanism at the time.

### A. Past User Story

In the span of two months, from October to November 2021, a fraudulent NFT creator and 𝕏 user known as Homer_eth executed five different NFT project rug pulls within two months, accumulating over $2.8 million in profits. Table I provides an overview of Homer_eth's rug pull projects, each with launch date and the estimated profit.

TABLE I. Rug Pull Projects by Homer_eth

| Project Name | Launch Date | Estimated Profit |
|---|---|---|
| Ether Bananas | 10/07/2021 | $125k |
| Ether Monkeys | 10/11/2021 | $1.77m |
| Zombie Monkeys | 10/15/2021 | $413k |
| Ether Reapers | 10/20/2021 | $282k |
| ETH Banana Chips | 11/23/2021 | $208k |

The basis of this user story is the transaction graph depicted in Fig. 4 that provides a simplified visualization of the transaction flow across multiple NFT projects linked to Homer_eth. It highlights key components, including EOA Nodes (Externally Owned Accounts), which represent the multiple wallet addresses of the rug puller, and Deployer Nodes (Smart Contract Creators), with the 0xc8a6 address being the deployer for multiple fraudulent contracts. The links between addresses are established through various transaction, such as mint transactions (e.g., mintReaper, mintBananaChips), which indicate purchases; fund transfers (e.g., Transfer 65.61 ETH to 0xc8a6), showing proceeds flowing to personal wallets or exchanges; and contract deployments (e.g., Deploy Ether Reapers). The transaction graph makes a critical indication of fraudulent intent visible. Instead of using a multisig treasury or project contract, funds were immediately funneled to a single address controlled by the rug puller.

Bob's story begins with a common enthusiasm for the burgeoning world of NFTs. His journey into the NFT market is marked by excitement and optimism, spurred by the success stories he sees online. Homer_eth, an NFT creator and 𝕏 user, has caught the attention of many like Bob by sharing his NFT

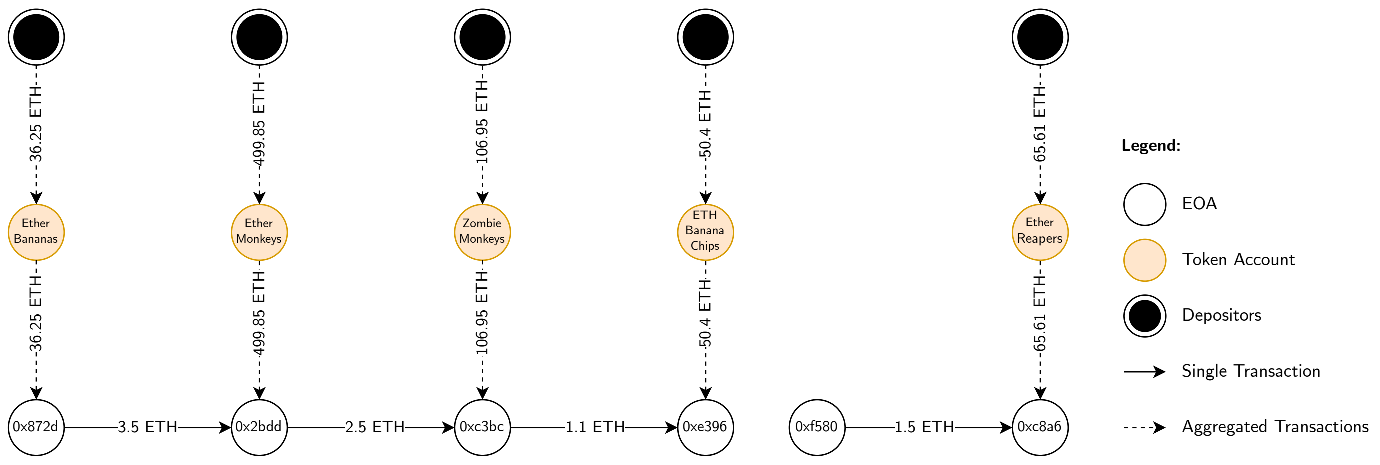


Figure 4. Simplified transaction graph of Homer_eth's NFT rug pulls.

projects on 𝕏. His first NFT collection was *Ether Bananas*, consisting of 750 NFTs, was launched on October 7, 2021. Only four days later, on October 11, Homer_eth continued with the release of *Ether Monkeys*, followed by the release of *Zombie Monkeys*. The buzz around Homer_eth's projects, especially *Ether Monkeys*, which promised additional utility through a casino to gamble and a decentralized autonomous organization to govern the NFTs, according to [45], draws Bob into the fray. Being relatively new to the NFT market, Bob views this as an opportunity not to be missed. Bob bought his first NFT from Homer_eth, an *Ether Reapers*, and with that purchase, he was no longer just a bystander; he was now an active participant in Homer_eth's growing community.

Bob's involvement in the community deepened over time. He engaged in discussions, shared his excitement with fellow members, and reveled in the rumors of more NFT launches in the future. His commitment paid off when he earned himself a whitelist spot that allows Bob to mint the upcoming NFT project *ETH Banana Chips* by Homer_eth. Convinced of its potential, Bob didn't hesitate to mint an *ETH Banana Chips* NFT when the opportunity arose. With a click to confirm the transaction in his browser wallet (e. g., MetaMask [46]), Bob became the proud owner of an *ETH Banana Chips* NFT, unaware of the underlying risks associated with his investment.

However, the reality of the situation was far from the optimistic scenario Bob had envisioned. Unknown to him, since Bob had a limited understanding of blockchain transactions, the proceeds from the *Ether Reapers* mint were not being locked in the smart contract for future development as promised. Instead, they were directly funneled into Homer_eth's deployer address. From there, Homer_eth will later transfer those mint proceeds either to his next deployer address for a future rug pull or to an exchange in order to pay out his profits made from rug pulling the projects.

After the launch of *ETH Banana Chips*, a tense silence enveloped the community. For months, there was no news from Homer_eth, no updates on the project, leaving everyone to wonder about the future. It wasn't until March 2022, that Homer_eth broke the silence with the announcement of one last NFT project, dubbed *Froggy Frens*. However, due to backlash from the community, Homer_eth deleted his 𝕏 account and vanished [45].

## B. Kosmosis Extension

The basis of the extended user story is the Kosmosis KG, depicted in Figure 5. Kosmosis identifies potential rug pulls by semantically analyzing transaction patterns encoded within smart contract interactions and cross-referencing blockchain addresses with real-world entity data from social media and other external sources. Our approach is grounded in the assumption that scammers publicly disclose or explicitly link blockchain addresses in their social media posts to promote their scams. This linkage is crucial for Kosmosis, as it provides

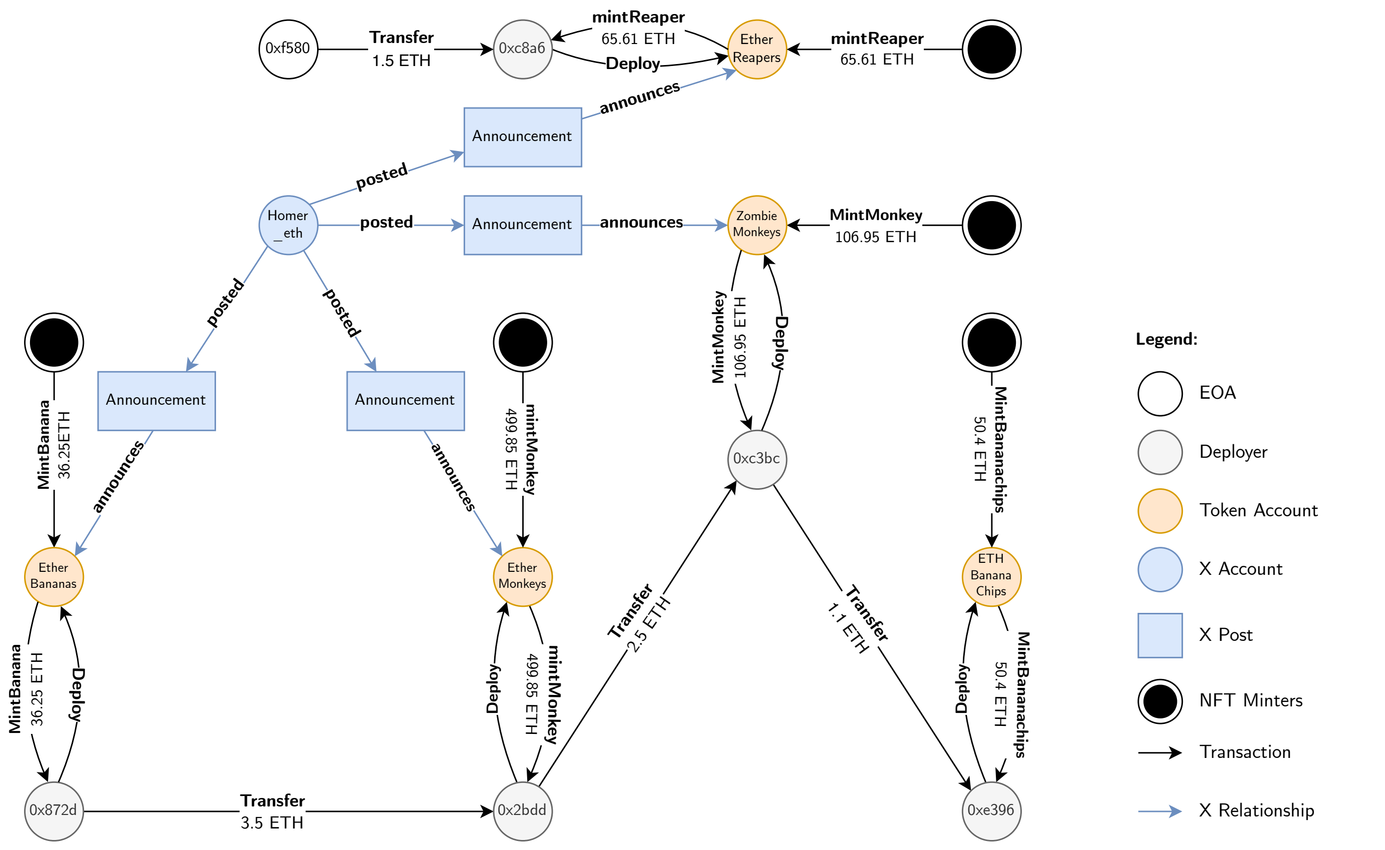


Figure 5. Knowledge graph of Homer_eth's NFT rug pulls, constructed using Kosmosis.

the primary method of associating blockchain transactions with social identities, which enhances the semantic richness of the constructed KG. The Kosmosis KG for this specific user story in Figure 5 is a direct enhancement over the basic transaction graph from Figure 4 as it was discussed in the previous subsection. The enhancement comprises semantically annotated edges and the incorporation of data from the social media platform 𝕏.

In order to detect potentially suspicious activity, we construct a Kosmosis subgraph using a multi-part SPARQL query. Listing 1 provides the overall CONSTRUCT query statement, capturing Ethereum transfers, mint transactions, smart contract deployments, bridging activities between chains, and social media references to blockchain addresses. Optional name labels help to identify the connection between blockchain or social media accounts and their associated real-world entities.

```
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
PREFIX kos:  <http://oth-aw.de/kosmosis#>

CONSTRUCT {
  # transfers
  ?sender    kos:transferTo ?receiver .
  ?trEdge  a kos:TransferEdge ;
             kos:from  ?sender ;
             kos:to    ?receiver ;
             kos:value ?sumValueETH .

  # contract deployments
  ?deployer  kos:deployed ?contract .

  # mint flows
  ?minter     kos:mint ?nftContract .
  ?mintEdge a kos:MintEdge ;
              kos:from  ?minter ;
              kos:to    ?nftContract ;
              kos:value ?sumMintETH .

  # UTXO unlock
  ?utxoIn      kos:unlocks ?contractUtxo .

  # bridging
  ?bridgeFrom  kos:depositToBridge ?bridge .
  ?bridge      kos:bridgeTransfer  ?bridgeTo .

  # social layer
  ?xAccount    kos:posted   ?xPost .
  ?xPost       kos:announces ?announcedContract .

  ?anyAccount  kos:accountName ?accName .
}
```

Listing 1. Construction of the RDF subgraph.

To enhance the interpretability of raw blockchain transactions, we introduce semantic annotations in our knowledge graph. This process involves using the Application Binary Interface (ABI) of smart contract transactions. Transactions interacting with NFT minting contracts contain function calls embedded in input data. Using the ABI, we extract function names (e.g., mintMonkey, mintBananaChips) and parameters. This enables labeling edges as minting transactions rather than generic asset transfers. Transaction type classification is done by categorizing transfers into value transactions, such as *mintMonkey* and *Transfer*, and non-value transactions, like contract deployments denoted as *Deploy*. These semantics allow describing (i. e., tagging) sender and receiver addresses as NFT minter (previously depositor) and deployer (previously EOA).

As part of the subgraph construction, multiple Ethereum value transactions between EOAs are aggregated in Listing 2. The query calculates the total ETH transferred from each sender to each receiver and assigns a unique semantic identifier to these aggregated transfers. This process reduces transactional complexity while preserving critical relationships for identifying significant value flows indicative of suspicious activity.

```
WHERE {
  {
    SELECT ?sender ?receiver (SUM(?value) AS ?sumValueETH)
    WHERE {
      ?tx a kos:ValueTransaction ;
            kos:from    ?sender ;
            kos:to      ?receiver ;
            kos:value   ?value ;
            kos:minedOn kos:Ethereum .
      ?sender   a kos:ExternallyOwnedAccount .
      ?receiver a kos:ExternallyOwnedAccount .
    }
    GROUP BY ?sender ?receiver
    HAVING (?sumValueETH > 0)
  }
  BIND( IRI(CONCAT("urn:tx-agg:",
        SHA256(CONCAT(STR(?sender),STR(?receiver))))) AS ?trEdge )
```

Listing 2. Aggregated ETH transfers between EOAs on Ethereum.

Due to the categorization of blockchain transactions by their semantic functions further important transaction types can be captured. Specifically, Listing 3 collects direct Ethereum smart contract deployments, aggregates Ethereum NFT mint transactions according to function names embedded in transaction data, and incorporates Bitcoin UTXO transactions. Each aggregated transaction type is semantically annotated to clarify the nature of the underlying blockchain interaction.

Cross-chain interactions through bridge protocols are captured explicitly in Listing 4. Ethereum deposits made into bridge smart contracts and their subsequent transfers to Polygon-based externally owned accounts are identified and annotated. This detailed semantic labeling assists in tracing asset movements across blockchains, crucial for identifying potentially risky bridging behavior.

Finally, the data integrated from platform 𝕏 enriches the KG with detailed information about user accounts, labeled as *X Account*, and specific announcements or posts, referred to as *X Post*. This integration facilitates a deeper understanding of the context and relationships surrounding these rug pulls. For

```
# Direct deployments
UNION
{
  ?deployer bco:deployed ?contract .
}

# Mint transactions
UNION
{
  SELECT ?minter ?nftContract (SUM(?val) AS ?sumMintETH)
  WHERE {
    ?mintTx a kos:CallTransaction ;
              kos:from  ?minter ;
              kos:calls ?nftContract ;
              kos:value ?val ;
              kos:action ?funcName .
    FILTER regex(?funcName, "^mint", "i")
    ?nftContract a kos:TokenAccount .
  }
  GROUP BY ?minter ?nftContract
}
BIND( IRI(CONCAT("urn:mint-agg:",
      SHA256(CONCAT(STR(?minter),STR(?nftContract))))) AS
    ?mintEdge )

# UTXO transactions
UNION
{
  ?utxoIn a kos:TransactionInput ;
            kos:unlocks ?contractUtxo .
}
```

Listing 3. Contract deployments and aggregated ‘mint’ calls on Ethereum, as well as UTXO transactions on Bitcoin.

```
UNION
{
  ?depTx  a         kos:ValueTransaction ;
          kos:from  ?bridgeFrom ;
          kos:to    ?bridge ;
          kos:value ?depVal .
  ?bridge a kos:ContractAccount .

  ?transTx a         kos:ValueTransaction ;
           kos:from  ?bridge ;
           kos:to    ?bridgeTo ;
           kos:value ?depValPolygon .
  ?bridgeTo a kos:ExternallyOwnedAccount ;
            kos:existsOn kos:Polygon .
}
```

Listing 4. Bridge deposits and transfers from and to the Ethereum blockchain.

```
UNION
{
  ?xAccount a kos:XAccount ;
              kos:posted ?xPost .
  ?xPost    a kos:XPost ;
              kos:announces ?announcedContract .
}
OPTIONAL { ?anyAccount bco:accountName ?accName }
}
```

Listing 5. 𝕏 Posts that announce contracts. Optionally, carrying over any account name literals, if present.

instance, the KG can establish a connection between previously unrelated entities, such as the deployer address *0xc8a6* and the user Homer_eth. This connection is made through a social media announcement in which Homer_eth claims to have created the Ether Bananas project, as well as through semantic annotation, which identifies *0xc8a6* as the deployer address of the Ether Bananas smart contract.

The final part of the query in Listing 5 retrieves social media data integrated from platform 𝕏, focusing on posts that explicitly announce (e.g., direct claim of creation: “Proud to announce the launch of Ether Bananas!”) or reference blockchain contracts in the comments section (e.g., “The CA is 0xCeF4CCb03dbc7D87B388407e381a949bE6d00E3b,” where CA stands for contract address of the NFT). It associates user identities (𝕏 accounts) and their posts with blockchain addresses they mention or promote. Optionally, the query includes usernames or labels from social accounts, further enhancing the contextualization of blockchain entities that allows to directly to real-world social identities, if possible.

In conclusion, the Kosmosis KG provides the semantic data foundation necessary for sophisticated detection logic. By aggregating transactions and enriching them with semantic annotations, the system can detect suspicious patterns, such as rapid bulk transfers following token minting events. Such transactions can be assigned elevated risk scores based on correlated indicators (e.g., rapid withdrawal to external accounts controlled by the deployer), triggering timely automated alerts and significantly reducing the reaction time required to prevent potential rug pulls.

## C. *Future User Story*

In an alternative scenario where Bob would have used Kosmosis, it would have analyzed the transaction history. The system would have issued a rug pull warning based on patterns of fund diversion to deployer addresses. Bob’s journey in the NFT market would have been safer, beginning with his initial transaction to purchase an *Ether Reapers* NFT.

As soon as Bob initiated his transaction, the rug pull prevention mechanism would have accessed the KG, to analyze the rug pull risk of the contract. Based on the integrated knowledge from 𝕏, the system would have been able to link the contract, Bob is about to interact with, to all of Homer_eth’s prior blockchain activity. The KG would have revealed a critical anomaly. Instead of the mint proceeds being transferred to the contract address of the project for future development, they were being diverted to the *Ether Reapers* deployer address via the *MintReaper* function. With smart contracts acting as an automated and trustless intermediary, where the code of the contract dictates the flow of funds according to predefined rules, this pattern of fund diversion is absent in legitimate projects. When funds are sent directly to a team member's address, in this case the deployer address, the funds can be moved to exchanges or other addresses with ease (i. e., pulling liquidity from the project without fulfilling the promises). This is a common tactic in rug pulls, where the developers abandon the project and disappear with the investor funds. Therefore, signaling a potential rug pull behavior. Upon detecting this anomaly, the system would have immediately issued a rug pull warning to Bob, prompting Bob to make an informed decision by asking whether he wishes to proceed with the transaction despite the identified risk. This proactive approach empowers Bob to reconsider his decision with full awareness of the potential danger, offering him a chance to opt-out before potentially falling victim to a rug pull.

## VII. Current Limitations

We aim to translate the Kosmosis approach into a practical implementation. The initial findings of our research on Kosmosis have shown promising results, indicating the potential of our approach in identifying and preventing rug pulls. However, there are ample improvement opportunities for Kosmosis in future work.

The generalization, from the exemplary use case to a sophisticated general rug pull classification method, covering various data patterns in the KG, is open research. Our subsequent endeavor involves the development of an algorithm capable of discerning rug pull warnings at varying confidence levels. This pursuit commences with the formulation of an intricate SPARQL query. Furthermore, an alerting system that utilizes the KG, constructed with Kosmosis, to alert users before interacting with a potential rug pull project, as described in the user story of Section VI, requires future efforts.

It will be necessary to refine the filters used in the ingestion of data from the 𝕏 Filtered Stream API. The current process of data ingestion depends on the presence of direct links to blockchain addresses in social media posts. For instance, the ability to link the user Homer_eth with the *EtherReapers* smart contract was solely facilitated by the explicit mention of the smart contract address in Homer_eth's announcement post on 𝕏. This example underscores the limitations of the current approach, which may overlook relevant connections in the absence of direct references. Consequently, a more sophisticated approach is required to ensure a broader and still relevant dataset is captured to associate 𝕏 users with their respective blockchain addresses.

Additionally, the implementation of knowledge fusion, the process of identifying true subject-predicate-object triples [47], sourced from the blockchain and social media stands out as a critical next step. By fusing multiple records representing the same real-world entity into a single and consistent representation [48], knowledge fusion would allow for a more accurate representation of real-world entities in the knowledge graph.

Currently, our prototype is limited to blockchains utilizing the account-based accounting model, like Ethereum. Recognizing the diversity in blockchain architectures and their unique features, we aim to allow for the integration of blockchains using a different accounting system, like Bitcoin. This expansion is essential for broadening the applicability and utility of Kosmosis across different blockchain platforms.

## VIII. Conclusion and Future Work

The Kosmosis approach represents a significant advancement in addressing the challenges associated with crypto rug pulls. Our proposed approach offers enhanced capabilities for semantic analysis, allowing the identification of fraud patterns that traditional transaction graph methods cannot detect.

We outlined a user story, where a threat actor known as Homer_eth executed five NFT project heists within two months, accumulating over $2.8 million in profits. In such a scenario, we showed that Kosmosis provides a knowledge graph that improves the detection of such fraudulent schemes carried out through sophisticated transaction patterns that might otherwise go unnoticed in related approaches, such as smart contract code analysis. This capability helps users make informed decisions and avoid becoming victims of fraud.

We also demonstrated how to build a knowledge graph from blockchain and social media data using the Kosmosis approach to incremental knowledge graph construction. Kosmosis becomes the basis for semantic querying and reasoning over a graph of entities and the relationships among them, facilitating analyses for cybercrime and fraud prevention, with the current focus on rug pulls as a major fraud scheme.

Kosmosis pipeline supports the ingestion of unstructured, semi-structured, and structured data, as well as the ingestion of new data at different time intervals. During construction, the semantics of blockchain transactions are extracted to address "why" and "how" crypto assets were transferred. Thus, Kosmosis extends the traditional transaction graph into a *semantically enhanced transaction graph* in which the sender and recipient are still pseudonyms. By incrementally constructing a *knowledge graph from blockchain and social media data*, Kosmosis also bridges the gap between pseudonymous transactions and real-world entities.

## References

[1] P. Stangl and C. P. Neumann, "Kosmosis: Crypto Rug Pull Detection and Prevention by Fusing On- and Off-Chain Data in a Knowledge Graph," in *Proc of the 16th International Conference on Cloud Computing, GRIDs, and Virtualization (Cloud Computing 2025)*, Valencia, Spain, Apr. 2025, pp. 1–8. DOI: 10.5281/zenodo.17272133.

[2] P. Stangl and C. P. Neumann, "The Kosmosis Use Case of Crypto Rug Pull Prevention by an Incrementally Constructed Knowledge Graph," in *Proc of the 2nd Workshop on Data Engineering for Data Science (DE4DS) in conjunction with the 21st Conference on Database Systems for Business, Technology and Web (BTW 2025)*, Bamberg, DE, Mar. 2025. DOI: 10.18420/BTW2025-131.

[3] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008, [Online]. Available: https://bitcoin.org/bitcoin.pdf (visited on 12/17/2023).

[4] S. Alizadeh, A. Setayesh, A. Mohamadpour, and B. Bahrak, "A network analysis of the non-fungible token (nft) market: Structural characteristics, evolution, and interactions," *Applied Network Science*, vol. 8, no. 1, p. 38, 2023.

[5] Chainalysis Inc., "The 2023 crypto crime report," Chainalysis, Feb. 2023, [Online]. Available: https://go.chainalysis.com/2023-crypto-crime-report.html (visited on 12/17/2023).

[6] T. Sharma, R. Agarwal, and S. K. Shukla, "Understanding rug pulls: An in-depth behavioral analysis of fraudulent nft creators," *ACM Trans. Web*, vol. 18, no. 1, Oct. 2023, ISSN: 1559-1131. DOI: 10.1145/3623376.

[7] A. Khan, "Graph analysis of the Ethereum blockchain data: A survey of datasets, methods, and future work," in *2022 IEEE International Conference on Blockchain (Blockchain)*, IEEE, Espoo, Finland: IEEE, 2022, pp. 250–257.

[8] F. Béres, I. A. Seres, A. A. Benczúr, and M. Quintyne-Collins, "Blockchain is watching you: Profiling and deanonymizing Ethereum users," in *2021 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*, Online: IEEE, 2021, pp. 69–78. DOI: 10.1109/DAPPS52256.2021.00013.

[9] H. Huang, W. Kong, S. Zhou, Z. Zheng, and S. Guo, "A survey of state-of-the-art on blockchains: Theories, modelings, and tools," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–42, 2021.

[10] A. Hogan *et al.*, "Knowledge Graphs," *ACM Computing Surveys*, vol. 54, no. 4, pp. 1–37, May 31, 2022, ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3447772. arXiv: 2003.02320 [cs.AI]. [Online]. Available: http://arxiv.org/abs/2003.02320 (visited on 12/11/2023).

[11] A. Chernysheva *et al.*, "SGDb Semantic Video Game Database: Svelte- und Ontotext-basierte Webanwendung mit einer Graphen-Suche für Videospiele," German, Ostbayerische Technische Hochschule Amberg-Weiden, Technische Berichte CL-2023-02, Mar. 2023. DOI: 10.13140/RG.2.2.11272.60160.

[12] J. Halbritter *et al.*, "Graphvio: Eine Graphdatenbank-Webanwendung für integrierte Datensätze von Streaminganbietern," German, Ostbayerische Technische Hochschule Amberg-Weiden, Technische Berichte CL-2022-01, Mar. 2022. DOI: 10.13140/RG.2.2.12111.46244.

[13] C. P. Neumann, T. Fischer, and R. Lenz, "OXDBS – Extension of a native XML Database System with Validation by Consistency Checking of OWL-DL Ontologies," in *Proc of the 14th International Database Engineering & Applications Symposium (IDEAS'10)*, Montreal, QC, CA, Aug. 2010, pp. 143–148. DOI: 10.1145/1866480.1866502.

[14] X. Zhu *et al.*, "Intelligent financial fraud detection practices in post-pandemic era," *The Innovation*, vol. 2, no. 4, 2021.

[15] C. Feilmayr and W. Wöß, "An analysis of ontologies and their success factors for application to business," *Data & Knowledge Engineering*, vol. 101, pp. 1–23, 2016.

[16] M. Hofer, D. Obraczka, A. Saeedi, H. Köpcke, and E. Rahm, "Construction of Knowledge Graphs: State and Challenges," 2023, eprint: 2302.11509 (cs.AI).

[17] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *2017 IEEE International Congress on Big Data (BigData Congress)*, Boston, MA, USA: IEEE, 2017, pp. 557–564. DOI: 10.1109/BigDataCongress.2017.85.

[18] T. Bauer *et al.*, "Reddiment: Eine SvelteKit- und ElasticSearch-basierte Reddit Sentiment-Analyse," German, Ostbayerische Technische Hochschule Amberg-Weiden, Technische Berichte CL-2022-06, Jul. 2022. DOI: 10.13140/RG.2.2.32244.12161.

[19] B. Hahn *et al.*, "Twitter-Dash: React- und .NET-basierte Trend- und Sentiment-Analysen," German, Ostbayerische Technische Hochschule Amberg-Weiden, Technische Berichte CL-2022-07, Jul. 2022. DOI: 10.13140/RG.2.2.15466.90564.

[20] G. Wood, "Ethereum: A Secure Decentralised Generalised Transaction Ledger," (Ethereum project yellow paper), Parity Technologies, 2024, [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf (visited on 01/29/2024).

[21] S. Wang, D. Li, Y. Zhang, and J. Chen, "Smart contract-based product traceability system in the supply chain scenario," *IEEE Access*, vol. 7, pp. 115 122–115 133, 2019.

[22] P. Stangl and C. P. Neumann, "FoodFresh: Multi-Chain Design for an Inter-Institutional Food Supply Chain Network," in *Proc of the 14th International Conference on Cloud Computing, GRIDs, and Virtualization (Cloud Computing 2023)*, Nice, France, Jun. 2023, pp. 41–46. DOI: 10.48550/arXiv.2310.19461.

[23] P. Stangl, "Design and Implementation of a Heterogeneous Blockchain Consortium for a Food Supply Chain Network," Bachelor's Thesis, Ostbayerische Technische Hochschule Amberg-Weiden, Jan. 2022. [Online]. Available: https://www.cyberlytics.eu/theses/all/OTH-AW/BT_2022_Stangl_Philipp_Thesis/BT_2022_Stangl_Philipp_Thesis.pdf.

[24] E. Takeuchi, "Explaining Ethereum contract ABI & EVM bytecode," Medium, Jul. 16, 2019, [Online]. Available: https://medium.com/@eiki1212/explaining-ethereum-contract-abi-evm-bytecode-6afa6e917c3b (visited on 12/07/2023).

[25] O. Marin, T. Cioara, L. Toderean, D. Mitrea, and I. Anghel, "Review of Blockchain Tokens Creation and Valuation," *Future Internet*, vol. 15, no. 12, p. 382, Nov. 27, 2023, ISSN: 1999-5903. DOI: 10.3390/fi15120382. (visited on 12/17/2023).

[26] F. Vogelsteller and V. Buterin, "ERC-20: Token Standard," Ethereum, Nov. 19, 2015, [Online]. Available: https://eips.ethereum.org/EIPS/eip-20 (visited on 12/17/2023).

[27] W. Entriken, D. Shirley, J. Evans, and N. Sachs, "ERC-721: Non-Fungible Token Standard," Ethereum, Jan. 24, 2018, [Online]. Available: https://eips.ethereum.org/EIPS/eip-721 (visited on 12/16/2023).

[28] M. M. Chakravarty *et al.*, "The extended UTXO model," in *Financial Cryptography and Data Security: FC 2020 International Workshops, AsiaUSEC, CoDeFi, VOTING, and WTSC, Kota Kinabalu, Malaysia, February 14, 2020, Revised Selected Papers 24*, Springer, 2020, pp. 525–539.

[29] C. Hoskinson, "Why we are building Cardano," IOHK, 2017, [Online]. Available: https://api-new.whitepaper.io/documents/pdf?id=HkUIhFWhL (visited on 12/16/2023).

[30] Y. Zhou *et al.*, "Stop pulling my rug: Exposing rug pull risks in crypto token to investors," in *Proceedings of the 46th International Conference on Software Engineering: Software Engineering in Practice*, ser. ICSE-SEIP '24, Lisbon, Portugal: ACM, 2024, pp. 228–239. DOI: 10.1145/3639477.3639722.

[31] Y. Elmougy and L. Liu, "Demystifying fraudulent transactions and illicit nodes in the bitcoin network for financial forensics," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ser. KDD '23, Long Beach, CA, USA: Association for Computing Machinery, 2023, pp. 3979–3990. DOI: 10.1145/3580305.3599803.

[32] L. Chen *et al.*, "Phishing scams detection in Ethereum transaction network," *ACM Trans. Internet Technol.*, vol. 21, no. 1, Dec. 2020, ISSN: 1533-5399. DOI: 10.1145/3398071.

[33] P. Stangl, "Design and Implementation of an Incremental Knowledge Graph Construction Pipeline for Investigating Crypto Asset Fraud," Masterarbeit, Ostbayerische Technische Hochschule Amberg-Weiden, Apr. 2024. DOI: 10.5281/zenodo.14518573.

[34] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 50–70, 2020.

[35] J. Pfeffer, "Ethon: Ethereum ontology," ConsenSys Software Inc., Dec. 7, 2023, [Online]. Available: https://ethon.consensys.io/index.html (visited on 12/07/2023).

[36] Etherscan, "Etherscan: The Ethereum blockchain explorer," Etherscan LLC, Dec. 7, 2023, [Online]. Available: https://etherscan.io/ (visited on 12/07/2023).

[37] Sourcify, "Sourcify: Source-verified smart contracts for transparency and better ux in web3," 2023, [Online]. Available: https://sourcify.dev/ (visited on 12/07/2023).

[38] F. A. Manzano and J. Little, "Pyevmasm: Ethereum virtual machine disassembler and assembler," Crytic, 2024, [Online]. Available: https://github.com/crytic/pyevmasm (visited on 01/25/2024).

[39] F. Victor, "Address Clustering Heuristics for Ethereum," in *Financial Cryptography and Data Security*, J. Bonneau and N. Heninger, Eds., vol. 12059, Cham: Springer International Publishing, 2020, pp. 617–633, ISBN: 978-3-030-51279-8. DOI: 10.1007/978-3-030-51280-4_33. (visited on 12/03/2023).

[40] X Corp., "Filtered stream introduction," X Corp., 2024, [Online]. Available: https://developer.twitter.com/en/docs/twitter-api/tweets/filtered-stream/introduction (visited on 01/25/2024).

[41] Golden Recursion Inc., "Golden Enrichment API: Enrich research, sales, and marketing with fresh, canonical knowledge.," Golden Recursion Inc., 2024, [Online]. Available: https://golden.com/product/api (visited on 01/25/2024).

[42] B. Mazorra, V. Adan, and V. Daza, "Do not rug on me: Zero-dimensional scam detection," 2022, eprint: 2201.07220 (cs.CR).

[43] R. Moody, "Worldwide crypto & nft rug pulls and scams tracker," Comparitech, Nov. 2023, [Online]. Available: https://www.comparitech.com/crypto/cryptocurrency-scams/ (visited on 11/19/2023).

[44] C. P. Neumann, *Distributed Case Handling*. München: Verlag Dr. Hut, 2013, ISBN: 9783843909198.

[45] ZachXBT [@zachxbt], "Homer.eth (formerly @homer_eth) rug pull analysis," X, X Corp., May 26, 2022, [Online]. Available: https://x.com/zachxbt/status/1529973318563946496 (visited on 12/05/2023).

[46] MetaMask, "Metamask: A crypto wallet & gateway to blockchain apps," ConsenSys Software Inc., 2023, [Online]. Available: https://metamask.io/ (visited on 12/15/2023).

[47] X. L. Dong *et al.*, "From data fusion to knowledge fusion," *Proc. VLDB Endow.*, vol. 7, no. 10, pp. 881–892, Jun. 2014, ISSN: 2150-8097. DOI: 10.14778/2732951.2732962.

[48] J. Bleiholder and F. Naumann, "Data fusion," *ACM Computing Surveys*, vol. 41, no. 1, pp. 1–41, Jan. 15, 2009, ISSN: 0360-0300, 1557-7341. DOI: 10.1145/1456650.1456651.

# Integer Sequence Refinement Using Language Models and Reinforcement Learning

1st Carolina Carvalho
*Department of Computer Science*
*University of Évora*
Évora, Portugal
carolina.rcxc@gmail.com

2nd Paulo Quaresma
*Department of Computer Science*
*University of Évora*
Évora, Portugal
pq@uevora.pt

***Abstract*—This article extends the applicability domain of language models to problems where candidate solutions can be expressed as an encoded integer sequence. Considering this sequence, language models can operate in the neural machine translation setting and leverage their optimization power for heuristic search techniques. Reinforcement Learning (RL) is applied to Language Models (LM), regardless of whether character-level or word-level models are used as a basis. To stabilize the learning, several approaches are explored, including functional and architectural decoupling. The framework is then applied to two combinatorial problems, namely the Traveling Salesman Problem benchmark and Neural Architecture Search, which is used to generate a hierarchical (tree-based) text classifier where the blocks are inspired by the InceptionV1 architecture. The decoupling results are the main contribution of this paper, easing the RL and LM stabilization requirements while expanding the resolution domain beyond Markov Decision Processes to non-causal normative heuristic problems, such as Neural Architecture Search (NAS).**

***Keywords*- *Heuristic Optimization; Reinforcement Learning; Language Model; Task Semantic Segmentation; Artificial Neural Network; Neural Architecture Search; Unordered Markov Decision Processes; Bellman Operator.***

## I. INTRODUCTION

Neural Machine Translation models are capable to generate text by mapping it from the origin language to a target one. Some training metrics which evaluate the translation quality are non differentiable such as the BLEU score, what makes it impossible to use as a training cost function when using Error Back Propagation and bringing this score's usage only to the evaluation stage.

By re-framing performance scores as heuristics, the training process can then be seen as an Heuristic Search methodology. To access this problem is known that the usage of Reinforcement Learning (RL) allows to optimize a non-differentiable metric, the reward, but its restricted to Markov Decision Processes (MDPs). This kind of training, RL, is also known as challenging to train in complex tasks.

This work proposes two decoupling philosophies: architectural and functional as well as solution encoding techniques which can be used in RL training. The proposed decoupling techniques allow to extend the RL training scope to more complex problems which are not MDPs and also enables the usage of more complex models, such as Language Models, in the RL training loop.

Regarding the learning process, it works in a similar to the Generative Adversarial Networks (GANs). By using a specialized critic network which learns the problem's features and rewards a task-preforming network, the agent network such as a professor and a student learning about the same problem.

Considering the sequence encoding of the candidate solutions generated by the language model, an ontology must be defined to encode and serialize the candidates, allowing the architectures to generate data structures and refine them during training, similar to a GAN training setting. In contrast to traditional Heuristic Search methods, where the candidate solution can be an array of various degrees of freedom in the problem (e.g., variables in a multivariate optimization problem), language models can capture the data structure or ontology with the help of special characters. These characters are used in the sequence encoding, signaling the evaluation methodology to build and assess a diversity of data structures. This capability is referred to in this paper as Semantic Encoding. It is then applied to the Neural Architecture Search downstream problem.

The rest of this paper is organized as follows: II. Related Work, III. Proposed Approach, IV. Sequence Encoding, V. Semantic Encoding, VI. Proposed Architectures, VII. Reinforcement Learning as a Search Methodology, VIII. Decoupled Asynchronous Advantage Actor-Critic, IX. Decoupled Soft Q-Learning, X. Decoupling's Mathematical Formalization, XI. Proposed Training Formulation, XII. Accessed Problems, XIII. Results, XIV. Error Analysis. Finally, the paper concludes with XV. Conclusions and Future Work and XVI. Acknowledgments.

## II. RELATED WORK

The present work is an extension of [1], an eKNOW 2025 conference paper with focus on formalizing the approaches.

Regarding the Natural Language Processing domain, the auto-encoder Language Models are typically trained on a large corpus. To evaluate the language model, the pretrained encoder, along with a custom decoder tailored to the downstream task, is then fine-tuned to address the specific task. The encoder part of the language model retains knowledge and maps semantics to a reduced latent dimension. This learned mapping keeps, in the encoder's weights, general type features, such as how to speak a language. This work explores the language model's encoder capability to retain the semantics of other problems beyond merely speaking a language. Additionally, the generative capability of language models is examined.

There are instances where the intention is to model the dataset's probability density function rather than the data itself. For example, in generative models, the goal is to generate data similar to the dataset. To achieve this objective, variational models come into play, specifically Variational Auto-Encoders (VAE) [2]–[12] and Generative Adversarial Network (GAN) architectures [13]–[19]. The GAN architectures use a Generator and a Discriminator network and employ min-max training. During training, the generator network produces data samples of better quality at each time step to trick the discriminator, which learns to distinguish real data from fake data generated by the Generator network. In this manner, both networks engaged in min-max training learn to perform their respective tasks. The Generator produces more realistic data samples as the Discriminator becomes increasingly difficult to deceive. In terms of VAEs, these models approximate the dataset's probability density function by modeling its parameters [20] or by assigning an odds to each output [21], generating data from the random variable where each output holds the model's estimated odds. The resulting binary text classifier positioning of a dataset, this work posits that any problem for which the solutions can be encoded in an integer sequence can also be addressed in a generative manner. It is essential to assert that the optimization goal is expressible through a heuristic function, akin to the fitness function in the context of Genetic Algorithms (GAs) [22]–[31]. Heuristic search using Language Models suffers from a lack of exploration due to the well-known difficulty of stabilizing complex neural models when trained using Reinforcement Learning. Traditional issues include training convergence and subsequent hyperparameter tuning. Furthermore, RL is usually applied sequentially to causal problems. This paper proposes decoupling-based RL training techniques and network architecture design principles that enable the application of RL to new problem types, as well as the incorporation of Language Models' feature capture capabilities to address problems beyond linguistic ones.

The proposed heuristic search relates mainly to evolutionary algorithms, such as Genetic Algorithms. The adopted models are neural Language Models, and the training is based on Reinforcement Learning. In this section, all the aforementioned methods are detailed. Evolutionary algorithms can be seen as heuristic search engines in the sense that they generate candidate solutions, which are evaluated on the fly using a heuristic function, such as the fitness function in the case of Genetic Algorithms. Neural Language Models (LMs) are used for language modeling [32]–[39]. They learn meaningful features from text data through embedding generation techniques. When an LM is used in the context of Neural Machine Translation [37], [40]–[45], LMs can be viewed as generative models because they generate tokens that, when decoded, are words in the target language domain. This problem can be generalized into a Sequence2Sequence problem when considering the same language model architecture generating a sequence with a different semantic encoding than target language tokens, always restricted to a differentiable loss function. Neural Machine Translation (NMT) architectures are generative by nature because they produce tokens in the model's target language, although their training typically requires a differentiable loss function that might not accurately express the training goals. The same occurs in Neural Architecture Search (NAS) tasks, where the primary objective is to increasingly enhance the candidate network's performance metric. In [46], a Recurrent Neural Network (RNN) is trained using Reinforcement Learning (RL) with the candidate network's performance almost directly serving as the reward function, employing various techniques to reduce the training's variance and facilitate learning through the described method. To relax the differentiable metric constraint, a new type of training is necessary; this is where Reinforcement Learning (RL) becomes relevant. RL techniques are primarily based on Markov Decision Processes (MDPs). Several training approaches attempt to optimize non-differentiable metrics in a deep model, such as surrogate losses [47], minimum risk training [48], and reinforcement learning [46]. All these training methodologies have their limitations: surrogate losses and reinforcement learning are difficult to stabilize, and minimum risk training is too computationally expensive when applied to a language model like an NMT architecture. Focusing on RL training, this article explores methods to stabilize the training and establish a robust optimization framework.

## III. Proposed Approach

To ease the training, several models are used inside the training loop. Both character-level and word-level language models are explored with different architectural and functional decoupling strategies. Regarding the character-level language model, it is proposed in [46] and is based on one-dimensional convolutional layers. In the proposed architecture, two models are used in this Soft Q-Learning training loop: one for acting on the environment (the target network) and another that learns the Soft Q-values. The models have the same architecture, so the weights with the learned Q-values can be set in the target network. In this way, transfer learning occurs during training. The problem of heuristic sequence building, also known as refinement, is broken into position and value generation; therefore, each model has two dedicated outputs. One output generates the position of the new element in the sequence, while the other generates the assigned value. In this way, there are dedicated architectural components: the core network learns the problem features and has connected dedicated Dense layers for each position and value generation, respectively. Notice that since Soft Q-Learning is being used, the target network outputs are odds, so a random experiment with the calculated odds distribution should be performed. Also, the SoftQ-Net has the same core architecture and learns the Q-values for the position and value parameters using the same core architecture as the Char-Conv. Thus, the target network has two learning sources: from interaction with the environment and from the transferred weights of the Q-Network.

For the word-level language model, the Transformer's encoder [49] was chosen to integrate into the Vector Quantized Variational Autoencoder proposed architecture. The quantized layer is derived from [49], and the dedicated outputs are based on Dense layers. In this case, one full model generates the position, while another is used for the value. Each full model is a VQ-VAE with two dedicated outputs: one for the actor and another for the critic's output. The two complete models are trained with a dedicated gradient tracker, but as they belong to the same problem, the reward and the state are the same for both models. One tracking the position and the other the value. The training of each VQ-VAE follows the Asynchronous Advantage Actor-Critic (A3C) methodology. By using the Vector Quantized layer proposed in [50], the search space is divided while the Transformer's encoder learns the problem's features in a parallelized manner, reducing the amount of time needed to explore it. An epsilon-greedy technique is also used to boost the algorithm's exploration. In this case, the model has a dedicated architectural core, the encoder, to learn the problem features, which are shared between the actor and the critic outputs. Consequently, the model learns from the actor-critic dynamic and through the heuristic reward. It is important to note that, in this case, the actor's outputs represent odds, and with this probabilistic representation, a random experiment is performed to choose the final selected action. The next step is to explain why does this decoupled architecture learn using RL.

## *A. Problem Formulation*

Considering a Markov Decision Process (MDP) with an initially unknown functioning that is learned using Reinforcement Learning and Artificial Neural Networks. This decision process can be of difficult representation because of it's complexity and this uncertainty can be modeled with randomness for more complex problems. Usually the state space is large and therefor its beneficial to use compression techniques such as latent spaces leanings as well as to make probabilistic learning by considering the model output as odds in a random experiment.

Usually in RL, the learning process is made incrementally and sequentially, what forces causality in the ANN model which is not desirable in some downstream problems such as Neural Architecture Search (NAS), where this incremental and sequential behavior does not adequately the problem dynamics. In such cases, decoupling the state generation task into position and value generation in order to change the next value in the sequence. This decoupling methodology allows to extend the RL training's applicability to larger problems which are not necessarily causal.

Next the impact of position-value decoupling, which is formally known as factorization, is accessed.

*1) Classical MDPs and Temporal Dependency:* Starting from classical MDPs:

$$M = (S, A, P, R, \gamma)$$

Being $(S, A, P, R, \gamma)$ the state, action, policy and reward spaces and $\gamma$ the learning factor. A policy $\pi$ maps states to actions, and the goal is to find $\pi^*$ maximizing:

$$J(\pi) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]$$

The value function $V^\pi(s)$ and action-value function $Q^\pi(s, a)$ satisfy:

$$V^\pi(s) = \mathbb{E}_{a \sim \pi(\cdot|s)}[Q^\pi(s, a)]$$

$$Q^\pi(s, a) = R(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[V^\pi(s')]$$

This framework is inherently sequential: each decision depends on the prior state and action.

*2) Decomposed MDP via Value-Position Factorization:* Now we introduce a decomposed representation of an MDP trajectory

$$\tau = \{(s_t, a_t, r_t)\}_{t=0}^T$$

into position-independent events.

**Definition (Decomposed MDP):**

Define the set:

$$Z_\tau = \{Z_t = (s_t, a_t, Q_t) \mid t = 0, \ldots, T\}$$

where:

$$Q_t := Q^\pi(s_t, a_t)$$

Time $t$ is now a latent index, not an explicit generator.

We view the trajectory as a set:

$$Z_\tau \in \mathcal{P}(S \times A \times \mathbb{R})$$

This is now a permutation-invariant object; that is, it lies in the space of unordered tuples, and hence non-sequential modeling is possible.

*3) RL Objective on Decomposed MDPs:*

*Step 1: Define the Learning Objective:* We define a modified objective:

$$J_{\text{decomp}}(\pi) = \mathbb{E}_{Z_\tau \sim \pi} \left[ \sum_{(s,a,Q) \in Z_\tau} w(s, a) Q \right]$$

where $w(s, a)$ is a weighting function (possibly uniform or importance-weighted).

This allows learning over unordered samples, with Q-values acting as surrogate return signals.

*Step 2: Bellman Consistency Holds:* We now prove: Bellman consistency can be enforced over unordered samples.

Let $Z_t = (s_t, a_t, Q_t)$, and suppose we estimate:

$$Q_\theta(s, a) \approx R(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[V_\theta(s')]$$

Define the loss:

$$L(\theta) = \mathbb{E}_{(s,a,Q) \sim \mathcal{D}} \left[ (Q_\theta(s, a) - Q)^2 \right]$$

If the samples $(s, a, Q)$ are drawn from any replay buffer or dataset, regardless of temporal order, the loss is still valid because the Bellman operator:

$$\mathcal{T}Q(s, a) := R(s, a) + \gamma \mathbb{E}_{s'} \left[ \max_{a'} Q(s', a') \right]$$

is defined pointwise.

This justifies learning Q-values independently of temporal order, so long as:

- $Q(s, a)$ is sampled accurately
- The agent uses valid updates

Now the above mentioned usage of odd regression in the neural architectures' output while making non-sequential sequence generation is analyzed.

*4) Enabling Odd Regression (Non-Sequential Generation):* Let's define *odd regression* as the ability to predict or generate parts of a trajectory non-sequentially, e.g., generating action $a_t$ before observing $a_{t-1}$.

## Non-Sequential Learning in Value-Decomposed MDPs

*Definitions*

Let an MDP be defined as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$ with:

- $s_t \in \mathcal{S}, a_t \in \mathcal{A}$
- $P(s_{t+1} \mid s_t, a_t)$: transition kernel
- $r_t = R(s_t, a_t)$
- $\pi(a \mid s)$: stationary policy
- $Q^\pi(s, a) = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R(s_k, a_k) \,\middle|\, s_0 = s, a_0 = a \right]$

Define a trajectory $\tau = \{(s_t, a_t, r_t)\}_{t=0}^T$ and its value-decomposed representation:

$$Z_\tau = \{Z_t := (s_t, a_t, Q_t) \mid Q_t = Q^\pi(s_t, a_t)\}_{t=0}^T$$

as an *unordered multiset*. We allow access to any subset $Z_{\setminus t}$ for conditional generation or regression.

*Step 1: Bellman Operator is Pointwise*

The Bellman operator $T^\pi$ is pointwise:

$$\begin{aligned} Q^\pi(s, a) &= (T^\pi Q^\pi)(s, a) \\ &:= R(s, a) + \gamma \, \mathbb{E}_{s' \sim P(\cdot|s,a)} \mathbb{E}_{a' \sim \pi(\cdot|s')} \left[ Q^\pi(s', a') \right] \end{aligned} \tag{1}$$

By definition, $(T^\pi Q)(s, a)$ depends only on the local state-action pair $(s, a)$ and the expected future values. That is, for any trajectory $\tau$, the update at $(s, a)$ does not require sequential access to previous or future transitions:

$$Q^\pi(s, a) = R(s, a) + \gamma \, \mathbb{E}_{s', a'} \left[ Q^\pi(s', a') \right] \tag{2}$$

Thus, learning $Q^\pi$ via Temporal DIfference (TD) methods or fitted Q-iteration is inherently *pointwise* and does not require temporal ordering. □

*Step 2: Learning from Unordered Sets*

Consider the Q-learning loss over a dataset $\mathcal{D}$ of transitions sampled in arbitrary order:

$$\begin{aligned} \mathcal{L}_Q(\theta) := \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}} \Big[ \big( Q_\theta(s, a) - r \\ - \gamma \, \mathbb{E}_{a' \sim \pi(\cdot|s')} Q_\theta(s', a') \big)^2 \Big] \end{aligned} \tag{3}$$

Minimizing $\mathcal{L}_Q$ over unordered tuples $(s, a, r, s')$ yields convergence to $Q^\pi$ under standard assumptions (ergodicity, coverage, Robbins-Monro conditions).

This follows directly from the convergence of TD learning and fitted Q-iteration. The temporal index $t$ is irrelevant: each update applies the Bellman operator $T^\pi$ pointwise. Therefore, transitions may be sampled in any order without affecting convergence. □

*Step 3: Permutation-Invariant Conditional Decoding*

Let a decoder $\pi(Z_t \mid Z_{\setminus t})$ be trained via the set-based loss:

$$\mathcal{L}_{\text{set}} := \mathbb{E}_{Z_\tau} \sum_{Z_t \in Z_\tau} - \log \pi(Z_t \mid Z_{\setminus t})$$

[Non-Sequential Generation is Bellman-Consistent] Let $Z_\tau = \{(s_t, a_t, Q^\pi(s_t, a_t))\}_{t=0}^T$. If

1) $Q_\theta$ is learned with Bellman-consistent TD updates,
2) $\pi(Z_t \mid Z_{\setminus t})$ is trained to predict individual transitions from context,

then

- The decoder can generate valid trajectory components in arbitrary order,
- Q-values remain consistent with the Bellman equations,
- Reinforcement learning objectives are correctly optimized even without sequential ordering.

Each $Z_t = (s_t, a_t, Q_t)$ satisfies $Q_t = Q^\pi(s_t, a_t)$ by pointwise TD learning. The conditional decoder $\pi(Z_t \mid Z_{\setminus t})$ is permutation-invariant: sampling any $Z_t$ conditioned on the rest respects the learned distribution of state-action-value triples.

Because $Q$-values are Bellman-consistent:

$$Q_t = R(s_t, a_t) + \gamma \mathbb{E}_{s', a'} [Q_\theta(s', a')]$$

any sampled element from $\pi$ will satisfy the Bellman equation relative to other elements. Therefore, one can reconstruct valid trajectories non-sequentially or fill missing entries conditionally. □

## IV. Sequence Encoding

The sequence which, in both of the proposed approaches, is the RL state and is refined during the training process, can have a learnable structure or ontology. For example, it can be a serialized image or text or an optimization problem candidate solution. For example, if a solution with a maximum accepted length is required, a padding char can be used. Special chars can be used to guide the solution's builder and evaluator and generate a proper reward according to it's performance, similarly to what is made in other Heuristic Search methodologies such as Genetic Algorithms.

In this document two problems were accessed, Neural Architecture Search (NAS) where the sequence is the Network Structure Code (NSC) and the reward is the built classifier's accuracy. The second problem is the Traveling Salesman Problem (TSP) in which the sequence is an ordered succession of visited cities. The RL reward is then an inverted total distance.

## V. Semantic Encoding

Sequence semantic encoding is one of the core subjects in this proposal. When applied to the sequence generated by a Neural Machine Translation model, the problem can be transposed into an optimization problem where the candidates can be encoded as a sequence of integers [46]. The candidate solutions' meta-format can be a single value or a sequence of values, depending on the downstream problem. Special characters such as separators or sequence terminators can also be used to help specify the solution's evaluator behavior. The optimization problem structure that this kind of semantic encoding enables is a heuristic search, since the candidate solution's quality is evaluated by a reward function that can be non-differentiable, and its value can be generated during the search execution.

For example, in order to access the Neural Architecture Search (NAS) problem using the proposed technique, the sequence can be the Network Structure Code (NSC), which encodes the candidate neural network hyperparameters. The network is then built and trained so that the performance metric can be extracted and the candidate sequence evaluated. Figure 1 highlights the proposed heuristic search architecture.

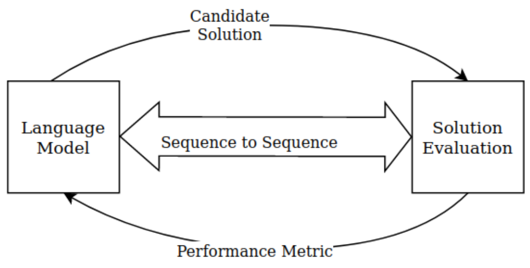


Figure 1: Heuristic Search Architecture: Similar to an evolutionary algorithm, this neural network–based candidate-solution refinement technique allows the solution evaluation block to be non-differentiable. This block only needs to produce an empirical performance metric, such as a RL reward.

## VI. Proposed Architectures

Depending on the nature of the problem, it can be beneficial to generate the sequence iteratively or through composition. As this article's subject is the usage of language models in optimization problems, and language models can encode semantics based on characters or words, both approaches will be explored further.

With the RL training enabled by the decoupling, based on unitary and semantically segmented tasks assigned to unitary model parts, the proposed architectures consist of two models inspired by character-level and word-level language models, respectively. With this training possibility, these models' generalization capability, as well as the proposed modeling principle, will be assessed.

### A. *Transformer-based Vector Quantized Variational Auto-Encoder with Asynchronous Advantage Actor Critic*

The word-level model is based on the Transformer architecture proposed in [49], which includes both the Transformer encoder and decoder architectures, along with the vector quantization layer proposed in [21]. This Vector Quantized Variational Autoencoder (VQ-VAE) architecture was decoupled as well; however, in this case, its outputs correspond to the actor and the critic. The actor outputs log probabilities for the possible actions of the RL agent, and the critic rates the inputs. Maintaining the sequence composition decoupling strategy, two models are employed to compose the target sequence. Once again, one model specializes in generating the position, while the second model generates the value to be assigned. Each model has two dedicated outputs, one to act an another for the critic while both share the core encoder that captures features from data. Figure 2 illustrates the architecture utilized for the VQ-VAE.

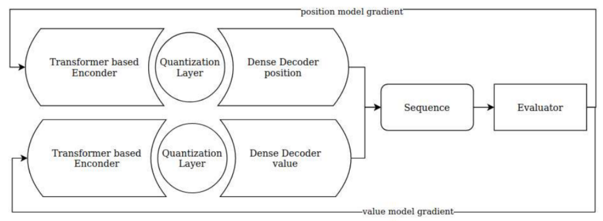


Figure 2: Decoupled Vector Quantized Variational Auto-Encoder proposed architecture using A3C. The encoder comes from the Transformer architecture, the quantization layer from [21], and the decoder is made of stacked dense layers. In this training architecture, each VQ-VAE has a actor output and a critic output. Two VQ-VAEs are represented in this figure and each one learns how to generate the value and position of the new element in the sequence.

### B. *Char-Conv with DeepQNet-Policy Learning*

Starting from the model proposed in [51], two output kernels were used to decouple the tasks into position and value generation. In this way, one model pair, Q-Network and Policy-Net, is used to compose the candidate sequence. Regarding the Traveling Salesman Problem, a benchmark problem, the proposed training setting works without issues. When considering the Neural Architecture Search problem, the reward signal presents high variance and the training did not converge to zero. In addressing this problem, two changes were made: entropy regularization was added, and the output activation function was changed to linear so that the model output is interpreted as log probabilities for each output position.

## VII. Reinforcement Learning as a Search Methodology

Since the search for optimal solutions is guided by reinforcement learning, the model generates multiple candidate solutions and iteratively improves them based on feedback. A heuristic evaluation function $g : \mathcal{Y} \to \mathbb{R}$ assigns a quality score to candidate solutions, serving as a reward signal:

$$R(y) = g(y). \tag{4}$$

Given any problem where the solution space $\mathcal{Y}$ is structured as integer sequences, the proposed methodology guarantees:

- **Expressibility**: The model $\hat{f}_\theta$ can learn to generate valid sequences from $\mathcal{X}$ using neural networks that are trained on $\mathcal{D}$.
- **Optimization Capability**: The reinforcement learning-based search ensures that generated solutions are iteratively improved using $g(y)$.
- **Generalization**: The auto-regressive nature of the model allows it to generate variable-length solutions applicable to different instances of the problem since a special character can be used as a sequence terminator.

Thus, for any integer-encoded problem, the formulation is sufficient to obtain high-quality solutions through iterative refinement. The proposed formulation applies to a wide range of problems where solutions are represented as integer sequences, including:

- Combinatorial optimization problems (e.g., the Traveling Salesman Problem, Knapsack Problem).
- Scheduling and planning tasks where actions are encoded as integer sequences.
- Code synthesis and symbolic regression.
- Game strategies with discrete action spaces.
- Non-sequential problems that benefit from value-position decoupling.

For any such problem, the integer-encoded representation ensures that the model can map problem instances to structured sequences and refine them over iterations using reinforcement learning. The search methodology follows a reinforcement learning-based approach such as DQN-PL [52], [53], A3C [54], and SoftQ-Learning [55]. The exploration methodology is epsilon-greedy for all the approaches. The different training methodologies are described in the next subsections.

## VIII. Decoupled Asynchronous Advantage Actor Critic

The main concept in decoupling is to create a problem feature extraction core and decoupled output decoders to model the output value according to the problem's required output. For example, in the VQ-VAE with the A3C training case, the same model generates the action and its corresponding critic value. To generate a sequence, two models with the specified decoupling are used: one generates the position of the new element, and the second generates its value. The resulting sequence is then updated and iteratively refined. Next, the formal formulation of this kind of decoupling is provided.

### A. Policy and Value Functions

Let $S$ be the state space, $A$ be the action space, and $P(s'|s,a)$ be the transition probability. The reward function is defined as $R(s,a,p)$, where $p$ is the selected position.

The policy consists of two independent components:

$$\pi(a|s;\theta_a) \quad \text{and} \quad \pi(p|s;\theta_p) \tag{5}$$

where:

- $\pi(a|s;\theta_a)$ selects an action based on state $s$.
- $\pi(p|s;\theta_p)$ selects a position based on state $s$.

The value functions are defined as:

$$V_{\text{act}}(s;\theta_v) = \mathbb{E}\left[R(s,a,p) + \gamma V_{\text{act}}(s')\right] \tag{6}$$

$$V_{\text{pos}}(s;\theta_p) = \mathbb{E}\left[R(s,a,p) + \gamma V_{\text{pos}}(s')\right] \tag{7}$$

### B. Exploration-Exploitation Strategy

The exploration rate for both action and position selection follows an epsilon-greedy decay:

$$\epsilon_a(t+1) = \max(\epsilon_a(t) \cdot d, \epsilon_{\min}) \tag{8}$$

$$\epsilon_p(t) = \epsilon_a(t) \tag{9}$$

where $d$ is the decay factor and $\epsilon_{\min}$ is the minimum exploration rate.

### C. Advantage Function and Returns

The advantage function for actions is given by:

$$A_{\text{act}}(s,a) = r + \gamma V_{\text{act}}(s') - V_{\text{act}}(s) \tag{10}$$

The advantage function for positions is:

$$A_{\text{pos}}(s,p) = r + \gamma V_{\text{pos}}(s') - V_{\text{pos}}(s) \tag{11}$$

The discounted return at timestep $t$ is:

$$G_t = \sum_{k=0}^{T-t} \gamma^k R(s_{t+k}, a_{t+k}, p_{t+k}) \tag{12}$$

The returns are then normalized:

$$\hat{G}_t = \frac{G_t - \mu(G)}{\sigma(G) + \epsilon} \tag{13}$$

### D. Loss Functions

The critic losses for action and position value networks are:

$$L_{\text{critic-act}} = \sum_t \left(A_{\text{act}}(s_t,a_t)\right)^2 \tag{14}$$

$$L_{\text{critic-pos}} = \sum_t \left(A_{\text{pos}}(s_t,p_t)\right)^2 \tag{15}$$

The actor losses are:

$$L_{\text{actor-act}} = -\sum_t \log \pi(a_t|s_t) A_{\text{act}}(s_t,a_t) \tag{16}$$

$$L_{\text{actor-pos}} = -\sum_t \log \pi(p_t|s_t) A_{\text{pos}}(s_t,p_t) \tag{17}$$

The total losses are:

$$L_{\text{total-act}} = L_{\text{actor-act}} + L_{\text{critic-act}} \tag{18}$$

$$L_{\text{total-pos}} = L_{\text{actor-pos}} + L_{\text{critic-pos}} \tag{19}$$

### E. Gradient Updates

Gradients for action and position networks are computed separately:

$$\nabla_{\theta_a} L_{\text{total-act}} = \sum_t \nabla_{\theta_a} L_{\text{total-act}} \tag{20}$$

$$\nabla_{\theta_p} L_{\text{total-pos}} = \sum_t \nabla_{\theta_p} L_{\text{total-pos}} \tag{21}$$

These gradients are applied using an optimizer:

$$\theta_a \leftarrow \theta_a - \alpha \nabla_{\theta_a} L_{\text{total-act}} \tag{22}$$

$$\theta_p \leftarrow \theta_p - \alpha \nabla_{\theta_p} L_{\text{total-pos}} \tag{23}$$

where $\alpha$ is the learning rate.

This content was generated with the help of generative artificial intelligence.

## IX. Decoupled SoftQ-Learning

Regarding the CharConv model in the NAS problem assessment, it was not possible to stabilize the training using the traditional DQN-PL approach. In the NAS setting, it was found beneficial for training stability to use stochastic outputs followed by a random experiment with the model's predicted output odds to generate the predicted action. To help stabilize the training in a stochastic environment, entropy regularization was employed.

Concerning the decoupling technique used in this context, two models were utilized. One model features a CharConv core and two decoupled outputs: one for value and another for the position of the new element in sequence generation. The second model is the target network, which generates the stochastic SoftQ-values for each output.

Additionally, an epsilon-greedy exploration strategy was applied in conjunction with an experience replay buffer. The proposed SoftQ-Learning approach uses a different decoupling when compared to the method presented in the previous section. This is specified in the subsequent subsections.

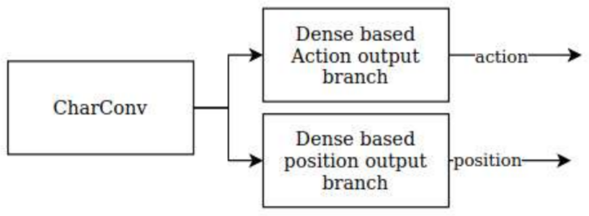

CharConv
Dense based Action output branch
action
Dense based position output branch
position

Figure 3: Proposed Decoupling SoftQ-Learning using character-based language model. This architecture is used twice in the training loop, one is specialized in generating new values for the action and position in the sequence to refine, the target-network and the other generates their corresponding Q-values, the Q-network.

### A. State and Action Representation

Let $s \in \mathcal{S}$ be the state space and $a \in \mathcal{A}$ be the action space. Additionally, let $p \in \mathcal{P}$ denote the position selection space. The agent selects both an action and a position at each time step.

### B. Soft Q-Function

Define the Q-function as:

$$Q(s, a, p) = Q_{\text{action}}(s, a) + Q_{\text{position}}(s, p). \tag{24}$$

This decoupling allows independent learning of action and position values.

### C. Soft Q-Learning Update Rule

The update rule follows the soft Bellman equation:

$$\begin{aligned} Q_{\text{action}}(s, a) &\leftarrow (1-\alpha) Q_{\text{action}}(s, a) \\ &+ \alpha \left[ r + \gamma \tau \log \sum_{a'} \exp\left( \frac{Q_{\text{action}}(s', a')}{\tau} \right) \right], \end{aligned} \tag{25}$$

$$\begin{aligned} Q_{\text{position}}(s, p) &\leftarrow (1-\alpha) Q_{\text{position}}(s, p) \\ &+ \alpha \left[ r + \gamma \tau \log \sum_{p'} \exp\left( \frac{Q_{\text{position}}(s', p')}{\tau} \right) \right]. \end{aligned} \tag{26}$$

where:

- $\alpha$ is the learning rate,
- $\gamma$ is the discount factor,
- $\tau$ is the temperature parameter for soft Q-learning,
- $r$ is the received reward,
- $s'$ is the next state.

### D. Action and Position Selection

The action and position are selected using the softmax policy:

$$P(a|s) = \frac{\exp(Q_{\text{action}}(s, a)/\tau)}{\sum_{a'} \exp(Q_{\text{action}}(s, a')/\tau)}, \tag{27}$$

$$P(p|s) = \frac{\exp(Q_{\text{position}}(s, p)/\tau)}{\sum_{p'} \exp(Q_{\text{position}}(s, p')/\tau)}. \tag{28}$$

This formulation allows efficient and structured learning by decoupling position and value, improving performance in reinforcement learning tasks that require both action selection and spatial positioning.

In the next section the position-value decoupling for integer-based sequences is formalized.

## X. Decoupling's Mathematical Formalization

When considering an iteratively generated sequence, in which the elements are generated one after another, the position is fixed and incremental, which implies causality in the sequence generation. By decoupling the functionality into position generation and value generation, thereby composing a single sequence (RL state), it is possible to break the causality implication and still utilize the reinforcement learning capability of optimizing heuristic functions. In this article, the decoupling is achieved at an architectural level; in a multi-branch architecture, each output branch is responsible for one single decoupled task in the non-causal sequence generation. To optimize a single sequence using two models, the state must be shared, and the RL techniques must still be applied to each model, utilizing separate optimizers guided by the same resulting reward.

In incremental sequence generation, this type of sequence generation allows for imposing causality in the RL agent's behavior, leading to a succession of actions generated throughout the training. Regarding compositional sequence generation, where the problem focus is to generate a candidate answer encoded in the sequence rather than a set of actions, decoupling can come into play to divide and conquer the generation problem into two sub-problems, enabling the composition of the sequence without needing to condition on the previous actions.

To extend RL beyond causal MDPs, we decompose the Q-function as follows:

$$Q(s,a) = P(s) + A(s,a), \tag{29}$$

where:

- $P(s) = \mathbb{E}[R|s]$ is the **position value**, which captures the expected reward at state $s$ independent of actions.
- $A(s,a) = Q(s,a) - P(s)$ is the **advantage function**, representing the additional benefit of taking action $a$ beyond merely being in state $s$.

If actions have no influence (a fully non-causal setting), then $A(s,a) = 0$, reducing RL to pure statistical inference:

$$V(s) = P(s) = \mathbb{E}[R|s]. \tag{30}$$

The objective function is defined as:

$$J(\pi) = \mathbb{E}_{s\sim D}[P(s)], \tag{31}$$

where $D$ is a dataset of observed states and rewards. If actions have partial influence, it is optimized as follows:

$$J(\pi) = \mathbb{E}_{s,a\sim D}[P(s) + A(s,a)]. \tag{32}$$

This formulation bridges RL and supervised learning, enabling RL in non-causal settings, such as:

- Counterfactual reasoning.
- Offline and batch RL.
- Decision-making in complex, non-Markovian environments.

## XI. Proposed Training Formulation

In this section, two training algorithms for Heuristic Optimization are proposed: the VQ-VAE model with A3C training and Char-Conv with DQNet-PL, so both character-level and word-level language models are explored.

We define the problem as a Markov Decision Process (MDP) with:

- State space: $S$
- Action space: $A$
- Transition dynamics: $P$
- Reward function: $R$

The objective is to learn a policy $\pi$ that maximizes the cumulative expected reward.

### A. State Representation

The state at time $t$, denoted as $s_t$, represents the environment state:

$$s_t \in S. \tag{33}$$

### B. Action Selection

A neural network models the probability distribution for action selection:

$$a_t \sim \pi(a_t|s_t;\theta). \tag{34}$$

The chosen action $a_t$ is sampled from this distribution.

### C. Critic Network (Value Estimation)

A critic network estimates the value function $V(s_t)$, representing the expected return from state $s_t$:

$$V(s_t) = \mathbb{E}\left[\sum_{k=0}^{\infty}\gamma^k r_{t+k}\right]. \tag{35}$$

### D. Reward and Return Calculation

The immediate reward $r_t$ is received from the environment. The discounted return is computed as:

$$G_t = r_t + \gamma G_{t+1}. \tag{36}$$

The returns are then normalized:

$$\hat{G}_t = \frac{G_t - \mu}{\sigma + \epsilon}. \tag{37}$$

### E. Advantage Estimation

The advantage function measures how much better the taken action was compared to the expected return:

$$A_t = \hat{G}_t - V(s_t). \tag{38}$$

### F. Actor-Critic Loss Functions

The loss for the actor (policy gradient) is:

$$L_{\text{actor}} = -\sum_t \log\pi(a_t|s_t)A_t. \tag{39}$$

The critic is updated using the Huber loss:

$$L_{\text{critic}} = \sum_t \text{Huber}(V(s_t),\hat{G}_t). \tag{40}$$

The Huber loss is defined as:

$$\text{Huber}(x,y) = \begin{cases}\frac{1}{2}(x-y)^2, & \text{if } |x-y| < \delta\\ \delta(|x-y| - \frac{1}{2}\delta), & \text{otherwise}\end{cases} \tag{41}$$

### G. Gradient Update

The gradients of the total loss function are computed as:

$$\nabla_\theta L_{\text{total}} = \nabla_\theta(L_{\text{actor}} + L_{\text{critic}}). \tag{42}$$

The parameters are updated using the Adam optimizer:

$$\theta \leftarrow \theta - \alpha\nabla_\theta L_{\text{total}}. \tag{43}$$

### H. Termination Criteria

Training stops when the running reward exceeds a threshold:

$$\sum_t r_t > R_{\text{target}}. \tag{44}$$

This indicates that the agent has effectively learned an optimal policy for the given task. In this A3C setting, two models are used in order to generate the sequence. In each step a new value and its corresponding position in the sequence (RL state) are generated. Each model has two outputs: one for the action and another for the critic score.

### I. Char Conv + DQN-PL

#### 1. Q-Function Approximation

We approximate the action-value function (Q-function) by a neural network with parameters $\theta$:

$$Q(s,a;\theta) \approx \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \,\middle|\, s_0 = s, a_0 = a\right],$$

where:

- $s$ is the state,
- $a$ is the action,
- $r_t$ is the reward at time $t$,
- $\gamma$ is the discount factor.

#### 2. Experience Replay

Experiences are stored in a replay buffer as tuples:

$$(s, a, r, s', d),$$

where $d$ is an indicator that equals 1 if $s'$ is terminal and 0 otherwise.

A mini-batch of $N$ experiences is sampled uniformly at random from the replay buffer for training.

#### 3. Target Calculation

For each sampled experience $(s, a, r, s', d)$, the target value $y$ is computed as:

$$y = r + \gamma \max_{a'} Q(s', a'; \theta^-) \cdot (1 - d),$$

where $\theta^-$ denotes the parameters of the target network, which are periodically updated to match the primary network parameters $\theta$.

#### 4. Loss Function

The loss function for a mini-batch is defined as the mean squared error between the target and the current Q-value estimate:

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N} \left(y_i - Q(s_i, a_i; \theta)\right)^2.$$

This loss is minimized to update the parameters $\theta$ of the Q-network.

#### 5. Gradient Descent Update

The parameters $\theta$ are updated via gradient descent:

$$\theta \leftarrow \theta - \alpha \nabla_\theta L(\theta),$$

where $\alpha$ is the learning rate.

#### 6. Action Selection (Epsilon-Greedy Policy)

At each step, the action $a$ is chosen according to the epsilon-greedy strategy:

$$a = \begin{cases} \text{random action}, & \text{with probability } \epsilon, \\ \arg\max_{a'} Q(s, a'; \theta), & \text{with probability } 1 - \epsilon, \end{cases}$$

with $\epsilon$ decaying over episodes from an initial value $\epsilon_{\text{start}}$ to a minimum value $\epsilon_{\text{min}}$.

#### 7. Periodic Target Network Update

Every fixed number of episodes (or steps), the target network parameters are updated by copying the weights from the primary network:

$$\theta^- \leftarrow \theta$$

## XII. Accessed Problems

For each of the two described ways to generate sequences, causal or non-causal, and regarding the Reinforcement Learning (RL) usage along with the proposed architectures, one benchmark problem was selected. The Traveling Salesman Problem (TSP) for causal generation and Neural Architecture Search (NAS) for non-causal generation.

### A. Traveling Salesman Problem

The TSP consists of generating a tour from a given starting city that passes through all the other cities while minimizing the overall path distance. The considered cities have the following coordinates:

TABLE I: Cities' coordinates used in the Traveling Salesman Problem.

| X | Y |
|---|---|
| 23 | 45 |
| 57 | 12 |
| 38 | 78 |
| 92 | 34 |
| 45 | 67 |
| 18 | 90 |
| 72 | 55 |
| 66 | 24 |
| 83 | 62 |
| 49 | 40 |

A distance matrix is calculated based on the euclidean distance between all the cities. A boolean array is used to track the cities already visited. If a generated city is already visited, the reward function gets the value of -100, in contrary, if a city is not visited, then the reward function gets the value given by:

$$\text{normalized\_reward} = 100 \cdot \left(1 - \frac{\text{distance}}{\text{max\_distance}}\right)$$

With:

$$\text{distance} = \text{distance\_matrix}[\text{current\_city}][\text{action}]$$

The cities road is generated iteratively, one city after another until the generated city is already visited. When this final condition is met, the obtained road is evaluated and the current episode ends.

### *B. Neural Architecture Search*

For the NAS problem, the sequence is interpreted as the Network Structure Code (NSC), meaning that it encodes an Artificial Neural Network (ANN). In this case, it is intended to generate a neural text classifier architecture built by several InceptionV1 blocks [56]. The NSC is composed by two decoupled models which contribute to the same RL final state, also known as NSC. The reward function is the child-network training accuracy which, in the current problem's case, is a classifier network. This classifier is built from an inverted n-ary tree encoded in Depth-First-Search (DFS).

## XIII. RESULTS

In this section, the performance plots for the NAS problem are presented. The adopted search space is an encoded n-ary tree using Depth First Search (DFS). The tree is encoded using $[0, 1, 2]$ in a sequence with a maximum of five positions. A zero encodes a change in the tree branch, a one encodes a deeper instruction, and the two is interpreted as a padding character. Each tree element is a Conv1D version of an InceptionV1 block [56]. When constructed, the tree is inverted so that the root node represents the classifier's final decision kernel. The search focuses on a text classifier, where the embeddings are provided by a Keras embedding layer. For evaluation purposes, this layer is replaced by the RoBERTa large model from Hugging Face [57], achieving state-of-the-art results with the IMDB sentiment analysis dataset [58]. The resulting model from the search was trained using a learning rate scheduler and presents the training curves shown in Figure 3.

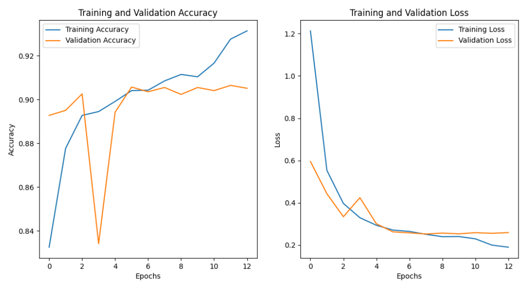


Figure 4: Classification accuracy and binary cross-entropy loss when using the generated NAS classifier and RoBERTa as embedding model.

The resulting binary text classifier positioning in the state of the art is presented in Table III.

TABLE II: Final model results on imdb sentiment analysis dataset.

| | |
|---|---|
| Test Loss | 0.2521449327468872 |
| Test Accuracy | 0.9054897427558899 |

The results presented were obtained by replacing the embedding layer with a pre-trained model from [59].

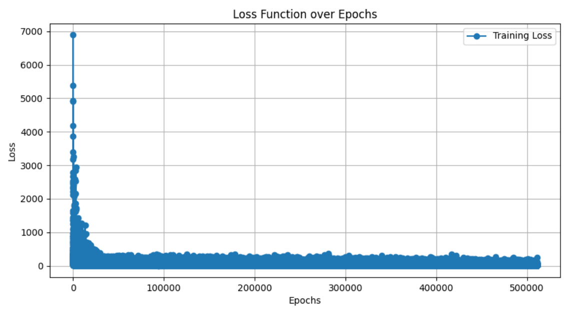
TABLE III: IMDb Sentiment Analysis Test Set Accuracy for Different Models in the Literature

| Model | Accuracy (%) | Reference |
|---|---|---|
| Naive Bayes (Baseline) | 83.5 | [60] |
| LSTM (Long Short-Term Memory) | 89.0 | [61] |
| BiLSTM with Attention | 91.2 | [62] |
| FastText | 88.5 | [63] |
| **RoBERTa+NAS Tree-based Classifier** | 90.5 | - |
| CNN (Convolutional Networks) | 90.6 | [64] |
| ULMFiT (Pretrained LSTM) | 94.0 | [65] |
| BERT-base (Fine-tuned) | 95.2 | [66] |
| RoBERTa (Fine-tuned) | 96.3 | [67] |
| DistilBERT | 95.1 | [68] |
| GPT-2 (Fine-tuned) | 95.0 | [69] |
| XLNet | 96.4 | [70] |
| ALBERT | 95.8 | [71] |
| ELECTRA | 96.6 | [72] |
| T5 (Text-to-Text Transfer) | 96.1 | [73] |
| GPT-3 (Few-shot) | 94.7 | [74] |
| DeBERTa (Fine-tuned) | 96.7 | [75] |
| ChatGPT (Prompting) | 96.0* | [76] |

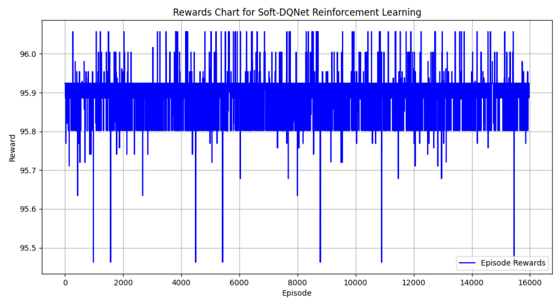
The neural architecture search task was performed using both language models: character-level using SoftQLearning and word-level using asynchronous advantage actor-critic training. In both cases, the problem's probability density function for each output was predicted by the models, and the final output is a result of a random experience with the model-predicted odds. This feature allows the models to represent more complex problems, such as NAS. This behavior also enables the model to learn the probabilistic aspects of a dataset; by fixing a serializable data ontology, it can generate datasets. Returning to the scope of this article, more specifically regarding these models' optimization capability in the NAS task, the learning and performance curves are presented below.

In the above experiment the model presented in [51], has two decoupled outputs which are used to compose the sequence - Network Structure Code. The Soft Q-Learning training method was adopted instead of DQNet-PL because the latest presents a very high training variance, making the training to not converge. Additionally the entropy regularization also helped to attain training convergence.

The transformer-based Vector Quantized Variational Auto Encoder (VQ-VAE) follows the same decoupling logic to compose the sequence, as described previously. In this case, the model has two outputs: the actor and the critic. The actor predicts odds for each possible model action, and the second output, the critic rates the overall model performance. In terms

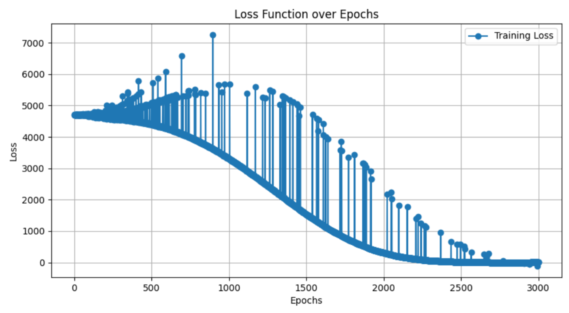


(a) Loss function of SoftQ-Learning using Char-Conv inspired architecture.

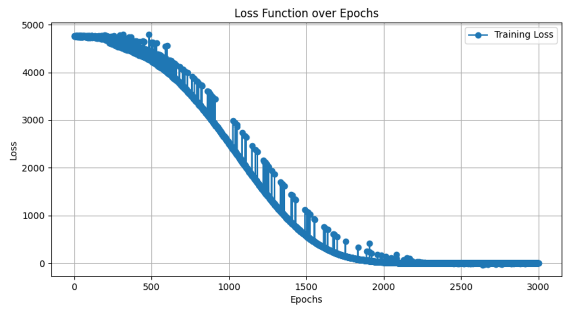


(b) RL reward function, child network's training accuracy, using SoftQ-Learning with Char-Conv inspired architecture.

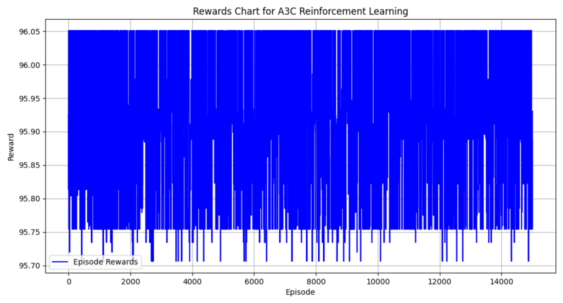
Figure 5: Comparison of loss and reward during training.



(a) Loss function of action sequence composing parameter during A3C training using Transformer-inspired architecture.



(b) Position loss function of A3C using Transformer-inspired architecture.



(c) RL reward and child network accuracy as functions of A3C using a Transformer-inspired architecture.

Figure 6: Training metrics during A3C using Transformer-inspired architecture: (a) Action sequence loss, (b) Position loss, and (c) Reward and accuracy.

of architecture, the actor-critic decoupling is made only in the model's last layer to shape the output according to the needs to generate the critic score and actor's odds.

Two Transformer-based VQ-VAE models were used to compose the sequence, one to generate the action and another to generate the position in the candidate sequence where the action value will be assigned. Bellow, the obtained training curves are presented:

The observable outliers are due to the epsilon-greedy technique used to introduce exploration in the algorithm's behavior.

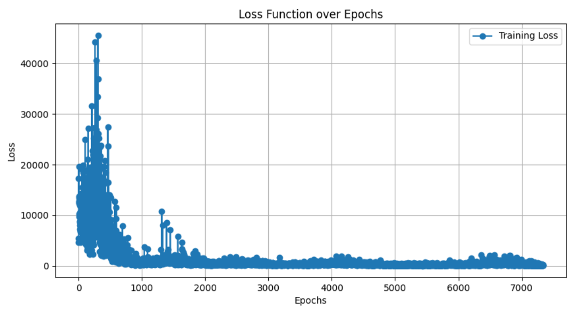
All the loss function plots in the presented results converge to zero, and the reward signals reflect the overfitting tendency of the proposed NAS methodology. The decoupling strategies are effective in stabilizing the training methodologies in both character and word-level approaches. Additionally, the sequence generalization and problem modeling capabilities are verified when observing the obtained training curves; both approaches exhibit stable behavior.

Next, the Traveling Salesman Problem results are presented. Experiments with both the architectures are presented bellow.

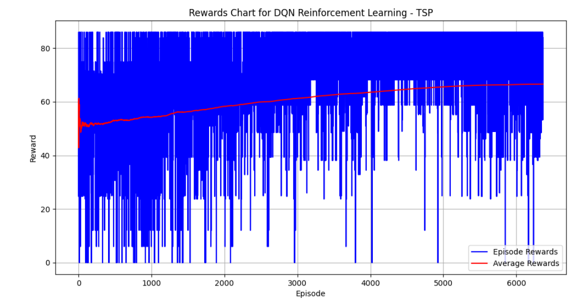
Starting from the Char-Conv as DQNet and as well as Policy network, the results were the following:

Both curves indicate that the RL agent is learning, as evidenced by the loss function's convergence to zero and the reward function's increasing behavior during training. Next, the VQ-VAE model is used in conjunction with A3C training to generate the salesman route:

The loss function chart exhibits zero convergence; therefore, training stability is concluded, and the generally increasing reward function reflects the VQ-VAE agent's learning. Depending on the problem complexity, generating action odds might be preferred rather than generating the agent's actions directly, as occurred with the Char-Conv architecture in NAS, where Soft-Q Learning was used, and in the TSP where DQNet-PL was utilized. The Transformer-inspired VQ-VAE demonstrates overall better training behavior compared to the Char-Conv architecture, as this model can map the search space into

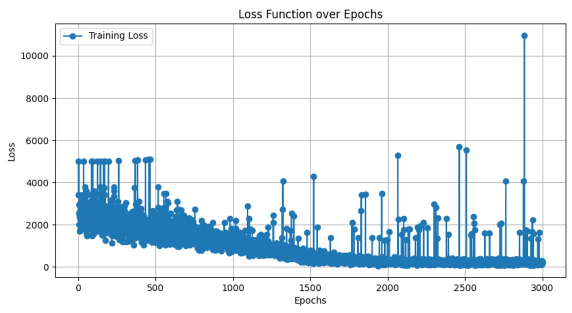


(a) Char-Conv architecture's loss function during DQNet-PL training, while solving the Traveling Salesman problem.

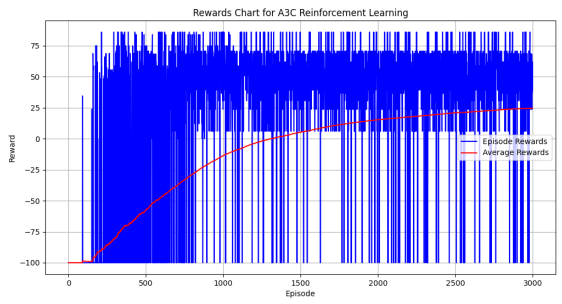


(b) Reward function of Char-Conv architecture during DQNet-PL training, while solving the TSP problem.

Figure 7: Training metrics of the Char-Conv architecture in DQNet-PL while solving the TSP: (a) Loss function and (b) Reward function.



(a) Loss function during A3C training using Transformer-inspired architecture in the TSP problem resolution.



(b) Reward function obtained by the VQ-VAE based agent in A3C.

Figure 8: Training metrics for TSP problem using A3C with Transformer-inspired architecture: (a) Loss function and (b) Reward function.

several sub-regions by utilizing the Vector-Quantized layer, thereby parallelizing the search.

## XIV. Error Analysis

During the experimentation phase of this work, the Traveling Salesman benchmark problem was addressed using A3C training together with the Transformer-based VQ-VAE model. Additionally, the Char-Conv model was tested alongside DQNet-PL training on the same problem. After several unsuccessful experiments resulted from the usual issues of high variance in the reward signal and a non-converging loss function, a functional decoupling methodology was developed and successfully applied to the TSP problem. The training results are presented in Figures 9 and 11 for the Char-Conv and VQ-VAE models, respectively.

In considering the NAS problem, the combination of Char-Conv with DQN-PL training did not succeed in solving this issue, as the loss function did not converge to zero. In contrast, the combination of VQ-VAE, A3C, and the respective decoupling effectively solved the problem (Figures 6 and 7). To address the limitations of solving the NAS problem using CharConv, SoftQ Learning with entropy regulation was employed, as it enables modeling the odds of each output and reduces the variance of the reward signal.

## XV. Conclusions and Future Work

Many problems are non-sequential and do not require strict left-to-right order dependency. To handle such cases, a value-position decoupling strategy is proposed. Considering the Transformer-based VQ-VAE trained with A3C, the model has two outputs: an actor output and a critic output. Instead of using two models, a single model is employed. In this way, the network weights are updated on both occasions: when the actor learns and when the critic learns. Two A3C models with a shared state and reward are used; one generates the new element's position, and the other generates the new element's values. The VQ-VAE architecture has the capability to divide the latent space into quantized subspaces and perform a parallelized search in each subspace. The deep convolutional network, trained using Deep Q-Learning for value generation and Soft Q-Learning for sequence generation, applies similar reasoning to design the training. One model with two outputs is responsible for generating the new element's position, while another model generates the new element's value. To make this training generative, the output odds are modeled, and the outputs are generated using a random experiment in which each output odd is defined by the Deep Q-models. Additionally, to reduce training variance, an entropy term is added to the loss function. This process is called entropy regularization and promotes training convergence toward zero. This study demonstrates that it is possible to generate sequences without causality constraints while still employing slightly adapted Reinforcement Learning techniques. Training convergence improves if the same model with two outputs is used to perform actions and critique its performance, regardless of its architecture.

Complex ontologies describing the candidate solutions can be encoded and serialized into integer sequences. The encoded sequences can then be optimized by this type of solver when used with a performance metric designed as the Reinforcement Learning reward. Since sufficient decoupling is achieved, the language models can absorb the problem's semantics and generate admissible candidate solutions of increasing quality. The position-value decoupling must be employed in the NAS scenario to avoid imposing causality in the sequence generation during the RL training. Additionally, using variational models in complex RL environments such as NAS is more efficient since they model the environment's unknown properties. The Transformer-based VQ-VAE is also capable of parallelizing the search due to the vector quantization layer. Looking toward the future, the models presented, along with the proposed training techniques, can be used to generate more than encoded solutions for a given problem. By selecting an appropriate reward function, the generated sequence can be utilized in the standard format to produce content similar to Generative Adversarial Networks. A comparison of the proposed solvers with other state-of-the-art heuristic search algorithms can be made to systematically explore the limitations of this proposal and extend its applicability domain. An analysis of the problem's degrees of freedom versus processing time will be conducted, focusing on solver quality analysis based on degrees of freedom, the solver's scaling with DoF, and the algorithm's parallelization. Going further in problem factorization into smaller problems can be automated, and the proposed learning techniques can be applied together with the parallelization capabilities of Vector Quantized layers using problem dissociation transforms such as the Laplace Transform. If a problem resolution process can be transposed into a computing tree, the currently generated n-ary tree for classifier construction can be reapplied into a problem resolution framework.

Regarding the algebraic properties of the Bellman operator, it is also possible to approach problem decomposition and simplification by decomposing the problem into Bellman blocks. Once these learnable blocks are identified, the proposed decoupling strategies can be applied to each block. This approach can be generalized, and the goal of searching for these blocks from given data can be replaced by the proposed NAS-based perspective.

## XVI. Acknowledgments

The English writing in this paper was reviewed by GPT-0 [77] and the mathematical formulations were Chat-GPT aided. We also thank Évora University and IARIA's reviewers for all the meaningful critiques.

## References

[1] C. Carvalho and P. Quaresma, "Heuristic search using language models and reinforcement learning," in *Proceedings of the Seventeenth International Conference on Information, Process, and Knowledge Management (eKNOW 2025)*. IARIA, 2025, pp. 1–12, © IARIA, 2025.

[2] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[3] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," *arXiv preprint arXiv:1401.4082*, 2014.

[4] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations*, 2017.

[5] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *Advances in Neural Information Processing Systems*, 2014, pp. 3581–3589.

[6] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, "Understanding disentangling in beta-vae," *arXiv preprint arXiv:1804.03599*, 2018.

[7] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Advances in Neural Information Processing Systems*, 2015, pp. 3483–3491.

[8] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," *arXiv preprint arXiv:1606.03657*, 2016.

[9] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[10] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," *arXiv preprint arXiv:1505.05770*, 2015.

[11] N. Dilokthanakul, P. A. Mediano, M. Garnelo, M. C. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan, "Deep unsupervised clustering with gaussian mixture variational autoencoders," *arXiv preprint arXiv:1611.02648*, 2016.

[12] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schölkopf, "Wasserstein auto-encoders," *arXiv preprint arXiv:1711.01558*, 2017.

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[14] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *International Conference on Learning Representations*, 2016.

[15] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[16] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, 2017, pp. 5767–5777.

[17] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.

[18] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.

[19] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 53–65, 2018.

[20] H. Ishfaq, A. Hoogi, and D. Rubin, "TVAE: Triplet-Based Variational Autoencoder using Metric Learning," no. 2015, pp. 1–4, 2018. [Online]. Available: http://arxiv.org/abs/1802.04403

[21] S. Paul, "Vector-Quantized Variational Autoencoders," 2021. [Online]. Available: https://keras.io/examples/generative/vq_vae/

[22] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press, 1975.

[23] D. E. Goldberg, "Genetic algorithms in search, optimization, and machine learning," in *Addison-Wesley*, 1989.

[24] K. A. De Jong, "An analysis of the behavior of a class of genetic adaptive systems," *Doctoral dissertation, University of Michigan*, 1975.

[25] J. H. Holland, "Building blocks, fringe search, and genetic algorithms," in *Foundations of Genetic Algorithms*, vol. 1, 1992, pp. 1–8.

[26] M. Mitchell, "An introduction to genetic algorithms," *MIT Press*, 1996.

[27] J. R. Koza, "Genetic programming: On the programming of computers by means of natural selection," in *MIT Press*, 1992.

[28] D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, no. 2, pp. 65–85, 1994.

[29] D. E. Goldberg and K. Deb, "Comparative analysis of selection schemes used in genetic algorithms," *Foundations of Genetic Algorithms*, vol. 1, pp. 69–93, 1991.

[30] K. Deb, "Multi-objective optimization using evolutionary algorithms," *Wiley*, 2001.

[31] T. Bäck, "Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms," *Oxford University Press*, 1996.

[32] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[33] J. L. Elman, "Finding structure in time," in *Cognitive Science*, vol. 14, 1990, pp. 179–211.

[34] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[35] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," *OpenAI preprint*, 2018.

[36] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[37] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[38] Y. Goldberg, "A primer on neural network models for natural language processing," *Journal of Artificial Intelligence Research*, vol. 57, pp. 345–420, 2016.

[39] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *IEEE Computational Intelligence Magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[40] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in Neural Information Processing Systems*, pp. 3104–3112, 2014.

[41] K. Cho, B. Van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734.

[42] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[43] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[44] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado *et al.*, "Google's multilingual neural machine translation system: Enabling zero-shot translation," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 339–351, 2017.

[45] C. Chu, R. Wang, and R. Dabre, "A survey of domain adaptation for neural machine translation," *arXiv preprint arXiv:1806.00258*, 2018.

[46] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pp. 1–16, 2017.

[47] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L. J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive Neural Architecture Search," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11205 LNCS, 2017, pp. 19–35.

[48] T. Shen, A. Kuncoro, X. Liu, O. Firat, P. Barham, and Q. V. Le, "Minimum risk training for neural machine translation," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016, pp. 1689–1699.

[49] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 2017-December, no. Nips, pp. 5999–6009, 2017.

[50] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," *arXiv preprint arXiv:1711.00937*, 2017.

[51] X. Zhang and Y. LeCun, "Text Understanding from Scratch," 2015. [Online]. Available: http://arxiv.org/abs/1502.01710

[52] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[53] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *International Conference on Learning Representations (ICLR)*, 2016.

[54] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning (ICML)*. PMLR, 2016, pp. 1928–1937.

[55] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine, "Reinforcement learning with deep energy-based policies," in *International Conference on Machine Learning (ICML)*. PMLR, 2017, pp. 1352–1361.

[56] C. Szegedy, S. Reed, P. Sermanet, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," pp. 1–12.

[57] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019. [Online]. Available: http://arxiv.org/abs/1907.11692

[58] S. Tripathi, R. Mehrotra, V. Bansal, and S. Upadhyay, "Analyzing sentiment using imdb dataset," in *2020 12th International Conference on Computational Intelligence and Communication Networks (CICN)*, 2020, pp. 30–33.

[59] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150. [Online]. Available: http://www.aclweb.org/anthology/P11-1015

[60] B. Pang and L. Lee, "Thumbs up? sentiment classification using machine learning techniques," in *Proceedings of the ACL*. Association for Computational Linguistics, 2002.

[61] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 conference on empirical methods in natural language processing (EMNLP)*, 2015, pp. 1422–1432.

[62] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu, "Attention-based bidirectional long short-term memory networks for relation classification," *Proceedings of ACL*, 2016.

[63] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, vol. 2, 2017, pp. 427–431.

[64] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization," *ACL*, 2017.

[65] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 328–339.

[66] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

[67] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[68] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2020.

[69] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, 2019.

[70] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.

[71] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2020.

[72] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators," in *ICLR*, 2020.

[73] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *JMLR*, 2020.

[74] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 1877–1901.

[75] P. He, X. Liu, J. Gao, and W. Chen, "Deberta: Decoding-enhanced bert with disentangled attention," *ICLR*, 2021.

[76] OpenAI, "Gpt-4 technical report," *OpenAI Technical Reports*, 2023.

[77] (2025) Zerogpt – freegrammar checker & spell checker ai-powered. Accessed: 2025-07-13. [Online]. Available: https://www.zerogpt.com/grammar-checker?r=gwords&gad_source=1&gad_campaignid=21008842773&gclid=Cj0KCQjwxJvBBhDuARIsAGUgNfiL-gX33yP9aHgXBh39K6i8q45ykTzyznD-uFuDr_pZs-5L7rivI9MaAhjZEALw_wcB

# Estimating When Leaves Turn Yellow and Fall for Tourists via Multimodal Monitoring with IoT Devices

Takumi Kitayama
Electronic Information Course
The Polytechnic University of Japan
Kodaira-shi, Tokyo, Japan
e-mail: b21306@uitec.ac.jp

Kenji Terada, Masaki Endo, Tsuyoshi Tanaka,Shigeyoshi Ohno
Division of Core Manufacturing
The Polytechnic University of Japan
Kodaira-shi, Tokyo, Japan
e-mail: {k-terada, endou, t-tanaka, ohno}@uitec.ac.jp

Hiroshi Ishikawa
Graduate School of Systems Design
Tokyo Metropolitan University
Hino-shi, Tokyo, Japan
e-mail: ishikawa-hiroshi@tmu.ac.jp

*Abstract*— **Japan's complex and varied topography gives rise to distinct seasonal landscapes, which serve as a major attraction for domestic and international tourists. Among these natural phenomena, the autumnal transformation of foliage—particularly the vivid yellowing of ginkgo leaves—holds considerable appeal. However, the phenological timing of leaf senescence and abscission exhibits substantial spatial variability, often leading to visitor dissatisfaction when travel coincides with either the premature stage prior to coloration or the post-abscission phase. To mitigate this issue, we propose a predictive system designed to estimate the timing of autumnal leaf coloration. This system employs Internet of Things (IoT) technologies to collect environmental data, including photographic imagery of ginkgo trees and measurements of solar radiation. The acquired data are then processed to forecast the imminent onset of leaf yellowing and subsequent abscission. A prototype implementation of the system was developed, and its predictive performance was empirically validated, demonstrating its efficacy in estimating key phenological transitions.**

***Keywords-Yellow Leaves Tourism; Internet of Things; Artificial Intelligence; Estimating.***

## I. INTRODUCTION

This paper is an extended version of earlier published work [1]. This paper added a quantitative evaluation of the results when deep learning was not used for image classification, clarifying the effectiveness of using deep learning.

They also made modifications to the definition of the yellowing rate and recalculated the time series analysis, resulting in improved accuracy.

Japan has a rugged landscape and is blessed with a diverse range of flora and fauna. These offer fascinating and unforgettable experiences for many tourists, both from Japan and abroad. Ginkgo trees, with their brilliant golden and vibrant yellow leaves, are a particularly popular tourist attraction. However, the period during which ginkgo trees shimmer in their beautiful golden hue is short. At most, they last about a week, and this period only occurs once a year in the fall. The time when ginkgo leaves turn yellow varies depending on the location. Even in the same location, weather conditions vary from year to year, so the leaves may not turn yellow on a specific day. If tourists visit before the ginkgo trees turn yellow, they will only see the green trees and miss out on the excitement. Even after the leaves have fallen, they will be sad to see them. Knowing when the leaves turn yellow is very important.

Biological seasonal observations [2] are conducted as a systematic approach to observing seasonal changes in various plants and animals, such as the yellowing of ginkgo leaves and the blooming of cherry blossoms. The Japan Meteorological Agency began this observation in 1953 and has covered 34 species and 41 phenomena. Biological seasonal observations rely on visual observation, which poses a challenge due to the enormous human cost involved. The scope of observations is being significantly reduced. It has also been pointed out that in urban environments, shading caused by buildings may affect biological seasonality. However, there is little research on the biological seasonal observation of local plants in urban environments [3][4][5] that could reduce human costs.

To address these challenges, we propose a method that reduces human labor while enabling the detection and prediction of yellowing in specific ginkgo trees within urban environments. First, we developed an IoT device capable of automatically collecting fixed-point photographs of ginkgo trees along with local meteorological data. From the captured images, the number of yellow leaf pixels is extracted to quantify the degree of yellowing. However, the apparent color of ginkgo leaves in images varies depending on factors such as cloud cover, camera performance, and leaf density. Simple pixel-based extraction cannot adequately account for these variations, often misclassifying green or other regions as yellow leaf pixels. To overcome this limitation, we employ deep learning–based color identification that is robust to such image variations. Moreover, recognizing that natural phenomena such as leaf yellowing and leaf fall progress continuously rather than as binary states, we aim to predict ginkgo phenology with higher granularity by extracting indicators representing the degree of yellowing and the extent of leaf fall, and conducting regression analyses using these indicators as objective variables.

This paper is organized as follows: Section II introduces related research. Section III explains the observation data. Section IV explains the observation system built using IoT devices. Section V explains how to analyze the collected observation data. Section VI presents the classification results, and Section VII discusses the prediction of the time when leaves will turn yellow. Section VIII summarizes this paper and discusses future challenges.

## II. Related Research

As biological phenological observations have been reduced, various studies on biological phenological observations have been reported to solve the problem. Below, we will discuss research related to the development of biological phenological observation methods.

In Endo et al.'s research [6][7], we proposed a method to estimate the timing of relic season changes in biological phenological observations at low cost from X (formerly Twitter) location-attached posts. By analyzing the names of organisms such as ginkgo and maple in the posts and co-occurring words indicating their location and state, the timing of biological phenological changes was estimated from the frequency of posts. Furthermore, the effectiveness of the proposed method was verified by comparing with observation data from the Japan Meteorological Agency.

In Iha et al.'s research [8], we used post data related to cherry blossoms from March to the end of April 2022 as a dataset and performed time series prediction of the number of posts using machine learning. As a result, we confirmed an improvement in the precision and recall of the time series prediction model of the number of posts compared to conventional methods.

In Ito et al.'s research [9], they developed a robot that can automatically measure plant growth information by utilizing low-cost IoT devices and open source image processing libraries. This robot was used to periodically capture images of spinach growth, demonstrating its potential for application in growth prediction and detection of poor growth.

In a study by Sato et al. [10], multispectral observations using a drone and IoT devices were used to observe the growth status of wheat using vegetation indices.

As described above, many methods have been researched for efficiently observing biological phenologies and plants using SNS(Social Networking Service) and IoT devices, but there has been no research on a system that can estimate the best time for yellow leaves to appear.

There are studies such as Meier et al. [11] and Kim et al. [12] that have attempted to predict the period of leaf yellowing and leaf fall based on long-term observation data from a botanical perspective, but these predictions are not based on data that can be observed using simple IoT devices. These studies rely on expensive observation data, such as human observation, and are different from the goal of this research, which is to develop a system that collects data at low cost and predicts the period of leaf yellowing.

## III. Observation Data

Biological phenological observations were performed in accordance with the Japan Meteorological Agency's biological phenological observation guidelines, and fixed-point photography was performed on ginkgo trees at the Polytechnic University as specimen trees (Figure 1). In addition, meteorological information from the surrounding area (hereinafter referred to as sensor measurements) is measured as a feature used to predict the yellowing and falling of ginkgo leaves. The sensor measurements are temperature, humidity, air pressure, carbon dioxide concentration, and illuminance.

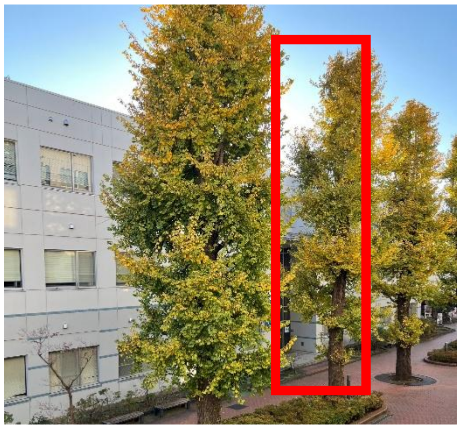


Figure 1. Specimen Trees.

Green and yellow leaves are related to photosynthesis. Photosynthesis is greatly affected by illuminance and carbon dioxide concentration. For this reason, in addition to basic sensors such as temperature, we also prepared sensors for illuminance and carbon dioxide concentration for observation.

The measurement sensor needs to be installed near the specimen tree. Because it also requires a power supply, the sensor was installed outside a window of a building near the specimen tree. The ginkgo tree on the far left of Figure 1 is closest to the sensor. However, because this tree reflects light and is prone to casting shadows, the ginkgo tree enclosed in a red frame was used as the specimen tree.

The observation period is from November 1, 2024 to January 10, 2025. The measurement frequency was one image and sensor measurement value set per minute. However, each sensor measurement value was taken for 24 hours, but images were taken only from 6:00 to 18:00. This is because it gets dark after 6pm, making it difficult to determine the color of the leaves. It is possible to take pictures at night using expensive, specialized cameras. However, visual inspection requires personnel costs, so we are trying to use IoT devices to reduce the cost. We avoided using expensive equipment. Furthermore, image data from times when no photography is taking place will be substituted with the last image, i.e., the image taken at 6pm. We did not consider using the next image taken, i.e., the image taken at 6am, because the image data cannot be determined until a photo can be taken the next day at 6am.

## IV. Observation System

The configuration of the measurement system is presented in Figure 2. Measurements related to the ginkgo trees are obtained using devices from the M5Stack series. Fixed-point photographs of the trees are captured with a Timer Camera. Temperature, humidity, and atmospheric pressure are recorded using the HAT-YUN module, $CO_2$ concentration is measured with an SGP30 sensor, and illuminance is measured with a BH1750FVI-TR sensor. All sensor data are collected and processed by an M5Stick microcontroller.

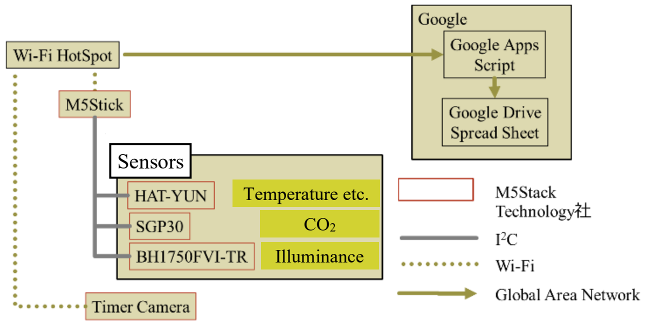


Figure 2. Overview of the observation system and data flow.

Images of the ginkgo trees will be saved to Google Drive and the sensor measurements will be saved to a spreadsheet using a script written in Google Apps Script provided by Google.

## V. Analysis Method

In preliminary experiments, we performed pixel-by-pixel determination of yellow leaves in acquired images. We defined the green and yellow ranges and classified them into three classes, including the rest. Specifically, the green range was defined as (160, 20, 15) to (210, 55, 50) in HSV, and the yellow range was defined as (36, 15, 30) to (60, 40, 70) in HSV. However, green and yellow leaves shine due to reflected light and do not fall within the predetermined green and yellow ranges. Conversely, shadows sometimes resulted in actual green or yellow leaves that also did not fall within the predetermined green and yellow ranges. We manually reviewed 5,000 images that did not fall within the yellow or green range of HSV. 13.5% were green leaf images and 9.6% were yellow leaf images. This indicates a 20% or greater chance of misclassifying leaves as green or yellow due to glare or shadows. Conversely, we manually reviewed 5,000 images that fell within the yellow or green range of HSV. 16.2% were not green or yellow, but were recorded as green or yellow due to glare or shadows. In other words, determining the green and yellow ranges in advance does not accurately determine green or yellow leaves. Therefore, we used machine learning to classify images into three classes using the following procedure.

Furthermore, we defined the yellowing and fallen leaf rates. From the acquired image, an area that only contains ginkgo leaves (hereafter, ginkgo image) is cut out. Furthermore, the ginkgo image is divided into 10x10 pixel images (hereafter, square images), and each square image is classified into "green", "yellow", and "other". The classification method is to first select only ginkgo images at times when the illuminance, one of the sensor measurement values, is between 1000 and 10000 lux. These images are divided into square images and labeled as "green", "yellow", and "other". These square images are used as learning data for training, and a model is generated that classifies the images into three classes: "green", "yellow", and "other". When an image containing only ginkgo leaves is divided into 10x10 pixels using this model, the number of images classified into each class is counted (Yellow Classification Count : y, Green Classification Count : g). In calculating the index, the ratio of the number of yellow class classifications to the total number of green and yellow class classifications (hereafter referred to as leaf amount), which indicates the entire ginkgo leaf, is defined as the "yellow leaf rate," Eq. (1) and the "fallen leaf rate" Eq.(2) is defined as the rate at which the leaf amount at the time of measurement has decreased from the maximum leaf amount obtained up to the time of measurement (hereafter referred to as maximum leaf amount : max(y + g)).

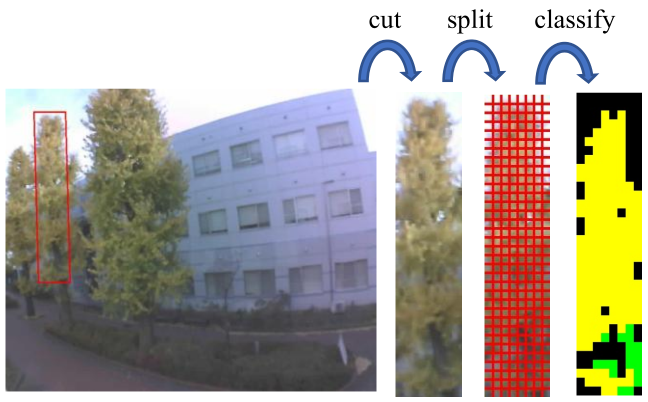


Figure 3. Learning and Classification Flow.

In earlier published work [1], we used the leaf yellowing rate given by Eq. (1). This is a natural definition, as it is the percentage of yellow leaves. However, it changes when leaf fall begins. When leaf fall begins, not only does the number of yellow leaves decrease, but the leaf amount also decreases. Therefore, if we use the yellowing leaf rate in Eq.(1), the value will change abnormally large when leaf fall begins. It is even possible for the yellowing leaf rate to increase even though the number of yellow leaves is decreasing. This has a negative impact on time series estimation. Time series estimation in earlier published work also did not achieve high accuracy. Therefore, we will modify the definition to Eq.(3), which can accurately represent the change in the decrease in yellow leaves after leaf fall begins. In this study, we recalculated using the yellowing leaf rate in Eq.(3).

$$\text{Yellow leaf rate (old)} = \frac{y}{g+y} \times 100[\%]\,. \tag{1}$$

$$\text{Fallen leaf rate} = \left(1 - \frac{g+y}{\max(g+y)}\right) \times 100[\%]\,. \tag{2}$$

$$\text{Yellow leaf rate (new)} = \frac{y}{\max\,(g+y)} \times 100[\%]\,. \tag{3}$$

According to the observation conditions of ginkgo in the biological phenology observation of the Japan Meteorological Agency [2], the yellow leaf day refers to the first day when the majority of the leaves have turned yellow when viewed as a whole and almost no green parts are visible. The defoliation day refers to the first day when approximately 80% of the leaves of the specimen tree have fallen. Therefore, the leaves are judged to be yellow when the yellow leaf rate is 80% or more of the maximum leaf amount, and the leaves are judged to be fallen when the defoliation rate is 80% or more.

Multivariate time series prediction is performed using LightGBM. If we can tell from camera images that the leaves are turning yellow or starting to fall, it will lead to a reduction in human costs. However, tourists need more information. What will the yellowing of the leaves be like in the next few days? When will the leaves start to fall? Information like that. For this, time series forecasting is necessary. Accuracy is verified using the yellow leaf rate and defoliation rate as the objective variables, and sensor measurements and processed data from them as explanatory variables.

## VI. Classification Results and Discussion

The results of three-class image classification using the image classification model ResNeXt are shown below. The three classes are yellow, which means yellow leaves, green before the leaves turn yellow, and other colors, which mean fallen leaves. Evaluation data was classified using the model determined to be the best by generalized k (k=5)-fold cross-validation, and the evaluation results shown in Table I were obtained. According to Table I, the precision rate for the green class classification is a little low at 0.892. However, all other colors were above 0.92. We counted the number of areas that were green, yellow, and other colors in the ginkgo image, and we believe that we were able to calculate the indices of yellow leaf rate and fallen leaf rate with high accuracy.

TABLE I. Evaluation of image classification of square images

| Class Name | Precision | Recall | F-Value |
|---|---|---|---|
| Green | 0.892 | 0.928 | 0.909 |
| Yellow | 0.945 | 0.927 | 0.935 |
| Others | 0.921 | 0.928 | 0.924 |

Details of the precision rate are shown in Table II. The precision rate is an evaluation index that indicates how accurate the prediction was. Looking at Table II, we see that the proportion of images classified as green that were actually labeled as yellow was 0.082, and the proportion of images labeled as other was 0.027. In other words, there were more images erroneously predicted to be green that were labeled as yellow than as other. This suggests that while the system was relatively accurate in classifying images where the correct answer was other classes such as trunks and branches, it is possible that the classification of green and yellow classes did not capture the subtle changes that occur when leaves change from green to yellow. The classification accuracy is significantly better than the classification performed in the preliminary experiment using a specified HSV value range. In particular, there are almost no cases where leaves that should have been classified as green or yellow because they were shining in the light are misclassified as neither green nor yellow. However, with the specimen tree, after the day the leaves turned yellow, images were observed in which shadows were cast by the sunlight, and these parts were mistakenly classified as green. Therefore, it is necessary to aim to improve accuracy by devising photography methods and image processing methods that are not affected by the direction of the sun or shadows.

TABLE II. Details of Precision Rate

| | | Classification results | | |
|---|---|---|---|---|
| | | Green | Yellow | Others |
| Actual | Green | 0.892 | 0.042 | 0.040 |
| | Yellow | 0.082 | 0.945 | 0.039 |
| | Others | 0.027 | 0.013 | 0.921 |

TABLE III. Objective and explanatory variables of the prediction model

| Objective variables | Explanatory variables |
|---|---|
| Yellowing and fallen leaf rates three days later | Average yellowing rate for the past three days |
| | Average falling leaf rate for the past three days |
| | Illuminance |
| | Average of illuminance and $CO_2$ integrated value for three days |
| | Additional value of average illuminance from 6:00 to 18:00 on the same day |

## VII. Prediction Results and Discussion

Using the ResNeXt image classification model, we were able to classify images into three classes: green leaves, yellow leaves, and others, with an accuracy of over 90%. Using these results, it became possible to calculate the rate at which leaves turn yellow and fall to determine whether the best time to see the yellow leaves is. Tourists need more information than this. Not just whether the leaves are turning yellow now, but also predictions about when they will turn yellow and when they will fall. We will build a predictive model using changes in past image data and weather information from sensors.

The objective variables and explanatory variables used in the LightGBM analysis were defined as shown in Table III for the leaf yellowing rate prediction model and leaf fall rate prediction model. Note that temperature, humidity, and air pressure, which were planned to be used as explanatory variables, were not used as explanatory variables because only fixed values were recorded from the middle of the observation period. For the explanatory variables in Table III, the objective variables were predicted based on the average leaf yellowing rate over the past three days and the average leaf fall rate over the past three days. In addition, the objective variables were predicted based on the average illuminance and the integrated value of illuminance and $CO_2$ over three days, as sunlight and photosynthetic activity would affect the objective variables. Furthermore, the average illuminance value from 6:00 to 18:00 on the same day was added to replace the integrated temperature to improve the prediction accuracy. Regarding the objective variables, the prediction period for yellowing and leaf fall judgment in previous studies was three days later, so the objective variables in this study were set to the leaf yellowing rate and leaf fall rate three days later from the last day of the average value of the past three days. Note that these data are saved every minute. Therefore, for the data at 12:00 on November 10, 2024, the average value of the explanatory variables over the past three days is the average value of the data from 11:59 on November 8 to 11:59 on November 10, 2024, and the target variables are the yellowing and falling leaf rates at 12:00 on November 13, 2024.

Figures 4 and 5 show the measured and predicted values of the percentage of yellowing leaves defined by Eq. (1) and Eq. (3), respectively, and Figure 6 shows the measured and predicted values of the percentage of defoliation. Furthermore, Table IV shows the evaluation of the yellowing leaf rate prediction model and the defoliation rate prediction model.

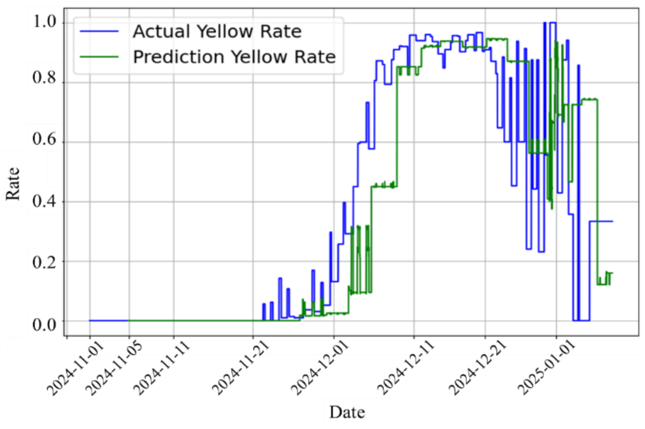


Figure 4. Actual value (blue) and predicted value (green) of the yellowing rate (Eq.(1)).

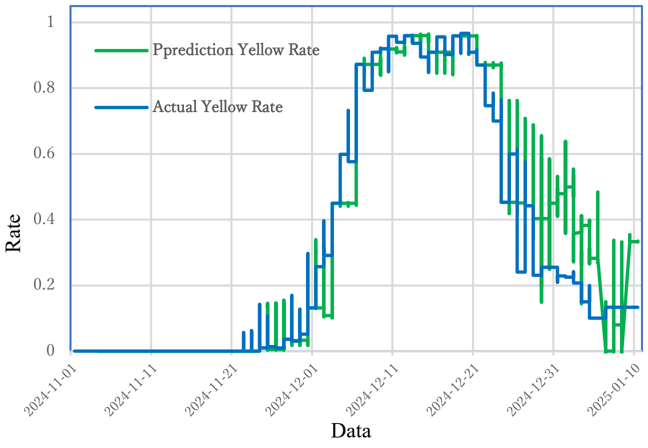


Figure 5. Actual value (blue) and predicted value (green) of the yellowing rate (Eq.(3)).

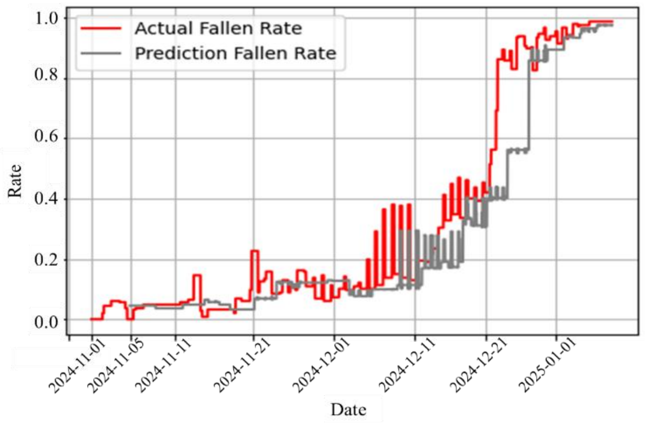


Figure 6. Actual value (red) and predicted value (gray) of leaf fall rate.

Figures 4, 5 and 6 show that the predicted values of the yellowing and falling leaf rates fluctuate about five days later than those calculated from the ginkgo image. In addition, after the fluctuation, there is no fluctuation and the rate remains flat for about three days. The reason for the approximately five-day delay in the fluctuation is thought to be that the data learned when the model output the predicted value of the evaluation data was from five days ago, so it was not possible to predict it as time-series data. In addition, the reason for the leveling off is that the yellowing and falling leaf rates were restricted to extract only the ginkgo image and only the time when the illuminance was within a certain range, leaving the yellowing and falling leaf rates blank for the time when the illuminance was outside the certain range. To fill this gap, the leaf yellowing rate and leaf fall rate from the most recent time when the illuminance was within a certain range were used. As a result, while the explanatory variables fluctuated during the learning data period, the objective variables, the leaf yellowing rate and leaf fall rate, did not fluctuate and remained flat during the period when the illuminance was outside of a certain range, so it is thought that the predicted values also produced similar outputs.

The data used in earlier published work contained approximately 10% overlapping timestamps. This is thought to be due to delays in data communication, so in this study, the duplicated data was removed.

TABLE IV. RMSE OF EACH REGRESSION MODEL

| Model | RMSE |
|---|---|
| The yellowing leaf rate (Eq.(1)) prediction model | 0.163 |
| The yellowing leaf rate (Eq.(3)) prediction model | 0.114 |
| The defoliation rate prediction model. | 0.083 |

Table IV shows that the RMSE (Root Mean Squared Error) of the leaf yellowing prediction model based on the percentage of yellowing leaves defined in Eq. (3) is 0.114. RMSE is an index that indicates the difference between the predicted value and the actual percentage of leaf yellowing or falling. This means that there is an average error of 0.114 between the predicted and actual values. This is a significant improvement over the RMSE of 0.163 for the leaf yellowing prediction model based on the percentage of yellowing leaves defined in Eq. (1). Comparing Figures 4 and 5, the difference becomes more pronounced as the percentage of falling rate increases, as shown in Figure 6. However, an RMSE of 0.114 means that even if the leaf yellowing rate is predicted to be 0.8, the actual result is 0.686 if it is low, or 0.914 if it is high, which is still not sufficient accuracy.

## VIII. Conclusion and Future Work

In this study, we first proposed a method for quantifying the degree of leaf yellowing and defoliation in ginkgo trees using deep learning–based image classification. Although variations in observation dates and leaf density influence the apparent leaf color in ginkgo images, the developed high-accuracy classification model enabled reliable extraction of yellowing and falling rates.

We then evaluated a prediction method that estimates the yellowing and falling rates three days in advance, using processed meteorological data as explanatory variables.

Several challenges remain for observation methods utilizing IoT devices. First, the accuracy of image classification must be improved by developing imaging strategies that minimize the influence of sunlight direction and shadows. Additionally, it will be necessary to detect anomalies in meteorological measurements and to address sensor failures through sensor redundancy.

In this study, malfunctions occurred in the temperature, humidity, and atmospheric pressure sensors, rendering their measurements unusable. Incorporating these sensor values, in addition to illuminance and carbon dioxide concentration data, is expected to further improve prediction accuracy. In particular, atmospheric pressure is closely related to weather conditions and may complement illuminance measurements.

Future challenges also remain in the method for estimating yellowing and falling rates from ginkgo images. The dataset must be expanded by developing image processing techniques capable of accurately extracting color information from images that are excessively bright or dark. Moreover, because images were captured only between 6:00 and 18:00, a future task is to obtain continuous image data throughout the day without relying on costly equipment. Since the measurement site in this study was limited to a single location, it will also be necessary to verify the applicability of the proposed method to trees in different environments.

By addressing these issues, we aim to achieve fully automated prediction of yellowing and falling timing for individual specimen ginkgo trees.

## References

[1] T. Kitayama, K. Terada, M. Endo, T. Tanaka, S. Ohno, and H. Ishikawa, “Proposal for a System to Estimate the Best Time to See Yellow Leaves Using IoT Devices for Tourists,” MMEDIA 2025 Proceedings of The Seventeenth International Conference on Advances in Multimedia, pp. 1-4, 2025.

[2] Japan Meteorological Agency. [Online]. Available from: https://www.jma.go.jp/jma/indexe.html [retrieved: 1, 2025].

[3] Y. Mizutani and K. Yamamoto, “A sightseeing spot recommendation system that takes into account the change in circumstances of users,” ISPRS International Journal of Geo-Information, vol. 6, no. 10, pp. 303, 2017.

[4] Y. Wang and W. Yue, “Proposal and Evaluation of a Pictorial Map Generation Method Based on the Familiarity and Satisfaction of Sightseeing Spots Calculated from Photos Posted on SNS (in Japanese),” Journal of the Japan Personal Computer Application Technology Society, vol. 16, no. 1, pp. 1–10, 2021.

[5] G. Fang, S. Kamei, and S. Fujita, “How to extract seasonal features of sightseeing spots from Twitter and Wikipedia (preliminary version),” Bulletin of Networking, Computing, Systems, and Software, vol. 4, no. 1, pp. 21–26, 2015.

[6] M. Endo, M. Takahashi, M. Hirota, M. Imamura, and H. Ishikawa, “Analytical Method using Geotagged Tweets Developed for Tourist Spot Extraction and Real-time Analysis,” International Journal of Informatics Society (IJIS), vol. 12, no. 3, pp. 157–165, 2021.

[7] M. Takahashi, M. Endo, S. Ohno, M. Hirota, and H. Ishikawa, “Automatic detection of tourist spots and best-time estimation using social network services,” International Workshop on Informatics 2020, pp. 65–72, 2020.

[8] A. Iha, K. Terada, M. Endo, T. Tanaka, S. Ohno, and H. Ishikawa, “Relation between theTtime-series Forecasting Methods and the Best Times to See Cherry Blossoms Estimation,” 17th International Workshop on Informatics (IWIN2023), pp. 145-152, 2023.

[9] J. Ito et al., "Development and Performance Evaluation of a Plant Phenotyping Platform Using Low-cost IoT Devices," Agricultural Information Research 30 (2), pp. 13–23, 2021.

[10] T. M. Sato, J. Kurihara, T. Yumura, and Y. Kakinami, "Predicting Optimal Harvest Time for Wheat Based on Multispectral Drone Images and Meteorological Data Collected Using IoT Sensors," Journal of Hokkaido Information University, vol. 36, pp. 39-52, 2024.

[11] M. Meier and C. Bigler, "Process-oriented models of autumn leaf phenology: ways to sound calibration and implications of uncertain projections," Geoscientific Model Development, Volume 16, pp. 7171–7201, 2023.

[12] S. Kim, M. Moon, and H. S. Kim, "Changes in the Timing of Autumn Leaf Senescence of Maple and Ginkgo Trees in South Korea over the Past 30 Years: A Comparative Assessment of Process-Based, Linear Regression, and Machine-Learning Models," Forests, Volume 16, Issue 1, 2025.

# SQL vs NoSQL in Cloud OLTP: A Case Study in Systematic Performance Evaluation

Carl Camilleri, Joseph G. Vella, and Vitezslav Nezval
Computer Information Systems, Faculty of ICT, University of Malta, Malta
e-mail: {ccami14 | joseph.g.vella | vitezslav.nezval}@um.edu.mt

*Abstract*—**The selection of appropriate DBMS for document-oriented applications significantly impacts system performance and operational efficiency. While prior studies have compared SQL and NoSQL technologies, many focus on narrow use cases or outdated system builds, offering limited guidance for today's multifaceted application requirements. This paper presents a systematic case study comparing PostgreSQL and MongoDB in a cloud-based environment, considering the needs of Cloud OLTP applications. By using an extended YCSB benchmark across 80 scenarios that vary dataset size, workload composition, and concurrency, the study highlights how systematic evaluation reveals nuanced performance trade-offs. PostgreSQL exhibited consistent strengths in mixed and high-concurrency workloads, while MongoDB demonstrated advantages in low-concurrency read-heavy and write-intensive scenarios. The findings underscore the importance of aligning workload characteristics with DBMS capabilities, and illustrate how structured, reproducible evaluation can inform more balanced database selection for Cloud OLTP applications.**

***Keywords-Database Management Systems; Performance Evaluation; Cloud OLTP; Relational Databases; Document Databases.***

## I. Introduction

Information systems (ISs) have become so prevalent that they have become crucial, if not critical, in facilitating the day-to-day human activity.

Systems must meet users' stringent Quality of Service (QoS) expectations. For example, studies indicate that response times of e-Commerce ISs exceeding two seconds can decrease user satisfaction and result in lost business [1], [2].

In tandem, users of transactional ISs also expect a high level of Quality of Data (QoD). For example, the customer experience on an e-Commerce IS will degrade if an order is placed online, only to be subsequently cancelled because it cannot be fulfilled due to depleted stocks. Business can be lost if the e-Commerce retailer cannot publish the latest prices or the newest products: prospective customers will look elsewhere to find the newest and cheapest alternatives.

Effectively, one of the core functions of a transactional IS is in line with the adages "Data is the new oil" [3] and "Data is the soul of the real world" [4]. Prospective customers demand fast access to the most up-to-date version of the dataset.

Data must be stored in some location, typically referred to as a database, and managed by a dedicated database management system (DBMS). Here, QoS and QoD can become competing objectives. Updating the data as fast as possible improves QoD, but this can have a negative impact on QoS: users' operations that read data and which must complete as fast as possible to guarantee the necessary QoS, start competing for hardware resources with write operations that are required for effective QoD.

The proliferation of data-intensive applications has fundamentally transformed the landscape of DBMSs, creating new challenges for system architects in selecting appropriate technologies for workloads sustained by ISs. Modern applications increasingly require systems capable of handling diverse data models, high transaction volumes, and stringent performance requirements while maintaining data consistency and availability. Furthermore, a new class of applications has been defined. "Cloud Online Transaction Processing" ("Cloud OLTP") applications [5] prioritise QoS, without necessarily requiring complex query capabilities and sophisticated transaction models (e.g., ACID).

This evolution has sparked considerable debate between traditional relational database systems and Not Only SQL (NoSQL) alternatives, particularly in scenarios involving document-oriented data structures. Modern applications increasingly demand DBMSs that can efficiently handle semi-structured data while maintaining the reliability and consistency guarantees traditionally found in relational systems. This creates a technology selection dilemma: system architects must choose between mature relational DBMSs with document capabilities and purpose-built document-oriented NoSQL systems, often without sufficient realistic evidence to guide their decisions. Despite the growing body of literature comparing database technologies [6], [7], systematic performance evaluations focusing on document-oriented Cloud OLTP workloads remain scarce. Existing studies often emphasise specific use cases [7] or fail to account for the multi-faceted nature of modern application requirements, including the need for both transactional integrity and query performance across varying load conditions [8]. Furthermore, the rapid evolution of both PostgreSQL and MongoDB technologies means that studies that compare DBMSs have a short shelf-life.

The fundamental challenge addressed in this research centres on conducting a systematic and empirical comparison of DBMS performance for document-oriented transactional workloads. We then use this methodology to specifically investigate the comparative performance characteristics of PostgreSQL and MongoDB when handling applications that require simultaneous support for transactional integrity, high availability, and scalable document management capabilities.

Our research contributes to the existing body of knowledge by presenting a case study that applies established systematic benchmarking practices to evaluating SQL and NoSQL in Cloud OLTP contexts, using PostgreSQL and MongoDB as examples for SQL and NoSQL technologies, respectively. Our empirical experiments evaluate both systems systematically by: (1) using identical hardware and network conditions in a production-like cloud environment, (2) examining performance across multiple workload compositions and scale factors, (3) analysing both throughput and latency characteristics under varying concurrent load conditions, and (4) providing practical guidance for system architects facing similar technology selection decisions.

We also aim to address some of the limitations identified in prior database comparison studies, including:

- Limited Scenario Coverage: Some studies [7] focus on narrow use cases without comprehensive workload variation, whilst in our work we perform tests along an 80-scenario matrix, hence providing broader coverage.
- Configuration Bias: Other studies [9] acknowledge that database comparison exercises do not ensure equivalent transactional and durability guarantees. We address this systematically, for example by ensuring that our workloads require a majority write concern for MongoDB.

In this context, this research provides:

1) **Benchmark Methodology**: A reproducible, cloud-based experimental approach for evaluating document-oriented database performance in cloud environments.
2) **Empirical Performance Comparison**: Evidence-based guidance for system architects and developers facing similar technology selection decisions.
3) **Empirical Performance Data**: Comprehensive performance measurements comparing PostgreSQL and MongoDB under comparable, realistic conditions.
4) **Workload-Specific Insights**: Detailed analysis of how different workload characteristics affect the relative performance of each system.

Our aim is therefore to advance beyond "which is faster" comparisons toward establishing systematic evaluation approaches that future database management comparison studies can adopt. We identify metrics, such as response time, to allow us to compare very different DBMSs, each using a vanilla configuration that is readily available in realistic setups.

The remainder of this paper is structured as follows: Section II reviews relevant literature on database management performance comparisons and OLTP workload characteristics. Section III describes our experimental methodology, including infrastructure setup and benchmark configuration. Section IV presents detailed results and analysis across different workload scenarios, and discusses the implications of our findings and provides recommendations for practical applications. Finally, Section V concludes with a summary of key contributions and directions for future research.

## II. Literature Review

### A. Choice of DBMS technology

The choice of DBMS technology significantly impacts application performance, scalability, and operational complexity. This remains a critical technical choice for the success of an IS application.

Choosing the right DBMS technology, or even the right mix of DBMS technologies, from the wide range of available options, is not a trivial exercise and it typically follows a prescribed method.

*1) Type of Workload:* First, one must identify the type of workload that the DBMS will sustain. Online transactional processing (OLTP) workloads consist of WRITE operations that modify small amounts of data, and READ queries that process a few records and retrieve the majority of the attributes available [10]. Cloud OLTP [5] is similar to OLTP, but each operation affects a single record. In contrast, Online analytical processing (OLAP) workloads typically consist of read-only queries that traverse a large amount of records, performing aggregations and retrieving a small set of attributes [10]. Workloads consisting of both transactional and analytical queries are referred to as Hybrid Transactional and Analytical Processing (HTAP) [11].

In this study, we tackle specifically the case for Cloud OLTP workloads.

*2) Data Integrity and Modeling:* Second, we must identify the DBMS features that are most pertinent to the application at hand. These may include:

1) **Data integrity**: does the DBMS need to enforce any application-specific operation pre-conditions or rules that determine whether an operation on a data element is accepted?
2) **Data modeling**: which data structures lend themselves best to both store the dataset, and satisfy the data management operations (e.g., READ and WRITE queries) in a manner that the correctness, QoS and QoD requirements of the application are optimisable?

Two of the most popular data models are the relational data model and the document data model. Codd's relational data model [12] popularised DBMSs, and stores data in tables which are matrices of rows and columns. Conversely, the document data model stores data as a series of documents identified by some unique key [13]. Relational DBMSs (RDBMSs) support the former data model, whilst Document-Oriented DBMSs support the latter. Some DBMSs have the capability to support more than one data model, and a popular data format in DBMSs that support the document data model is the JavaScript Object Notation (JSON). Relational DBMSs are typically managed via the Structured Query Language (SQL), a domain-specific language first standardised in the late eighties [14]. Conversely, non-relational DBMSs rely on other languages and are collectively referred to as No-SQL DBMSs [15]

*3) Scalability:* Another aspect of consideration is the need for horizontal scalability. Systems that require the DBMS to scale horizontally need to opt for a distributed DBMS (DDBMS) [16]. This need can be driven by several objectives, as illustrated in Figure 1. The way data is distributed across several machines for horizontal scalability can also differ, as shown in Figure 2, Figure 3, and Figure 4. The choice of data distribution strategy affects a DBMS's capability in achieving the requirements for horizontal scalability.

Traditional RDBMSs, exemplified by PostgreSQL[1], have long dominated transactional workloads due to their adoption of ACID[2] guarantees, mature optimisation techniques, and standardised SQL. However, the rise of document-oriented NoSQL databases, such as MongoDB [17], has challenged this dominance by offering flexible schema designs, horizontal scaling capabilities, and optimisations specifically tailored for document-based operations.

The technological dichotomy of RDBMSs and NoSQL databases presents particular challenges for applications managing semi-structured data that can be effectively modeled using either relational or document data models. Such applications often exhibit mixed workload characteristics, combining high-frequency read operations with periodic write-intensive tasks, demanding both transactional consistency and query performance optimisation. The decision between relational and document-oriented approaches becomes further complicated when considering operational factors such as horizontal scalability requirements, data consistency guarantees, and infrastructure complexity.

Recent advances in multi-model database capabilities have blurred traditional boundaries between relational and NoSQL systems. PostgreSQL's native JSON support and document querying capabilities enable it to handle document-oriented workloads effectively, while MongoDB has introduced features to strengthen data consistency guarantees and transactional support. This convergence necessitates empirical evaluation to understand the performance implications of each approach under realistic workload conditions.

### B. Functional DBMS Requirements for OLTP

OLTP and Cloud OLTP applications, including those handling document-based data, benefit from several features that a DBMS can offer, namely:

- **Transactional Features** [18], [19]
  - **OLTP Workload Suitability**: The system must handle workloads where data retrieval and modification operations are typically restricted to small sets of records.
  - **Data Integrity Constraints**: The system must provide mechanisms to enforce business rules and maintain data consistency, further ensuring that concurrent operations do not violate application-specific constraints.
  - **Strong Consistency Guarantees**: For critical operations, the DBMS must provide a view where concurrent data changes appear to be applied in a total order across all clients, preventing anomalies in multi-user environments.

[1]https://www.postgresql.org/, Dec 15, 2025
[2]Atomicity, Consistency, Isolation, and Durability

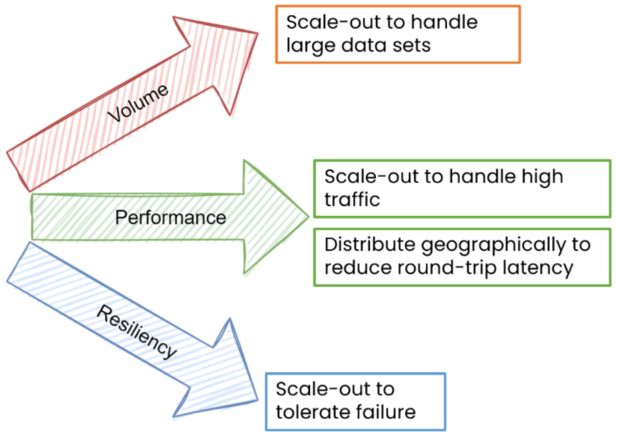
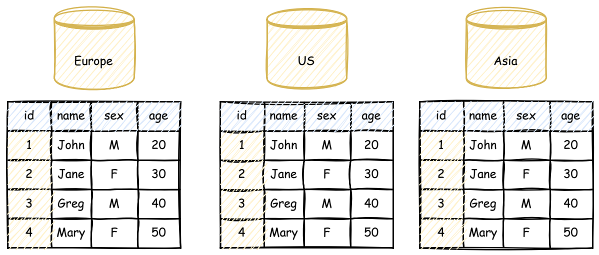

Volume
Scale-out to handle large data sets
Performance
Scale-out to handle high traffic
Distribute geographically to reduce round-trip latency
Resiliency
Scale-out to tolerate failure

Figure 1: Objectives of DBMS scalability

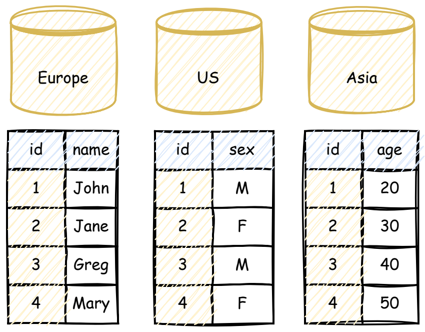


| id | name | sex | age |
|---|---|---|---|
| 1 | John | M | 20 |
| 2 | Jane | F | 30 |
| 3 | Greg | M | 40 |
| 4 | Mary | F | 50 |

| id | name | sex | age |
|---|---|---|---|
| 1 | John | M | 20 |
| 2 | Jane | F | 30 |
| 3 | Greg | M | 40 |
| 4 | Mary | F | 50 |

| id | name | sex | age |
|---|---|---|---|
| 1 | John | M | 20 |
| 2 | Jane | F | 30 |
| 3 | Greg | M | 40 |
| 4 | Mary | F | 50 |

Figure 2: Data Distribution via full Replication (Mirroring)

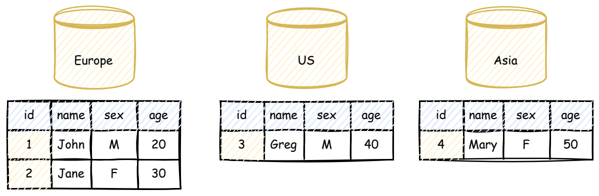


| id | name |
|---|---|
| 1 | John |
| 2 | Jane |
| 3 | Greg |
| 4 | Mary |

| id | sex |
|---|---|
| 1 | M |
| 2 | F |
| 3 | M |
| 4 | F |

| id | age |
|---|---|
| 1 | 20 |
| 2 | 30 |
| 3 | 40 |
| 4 | 50 |

Figure 3: Data Distribution via Vertical Partitioning



| id | name | sex | age |
|---|---|---|---|
| 1 | John | M | 20 |
| 2 | Jane | F | 30 |

| id | name | sex | age |
|---|---|---|---|
| 3 | Greg | M | 40 |

| id | name | sex | age |
|---|---|---|---|
| 4 | Mary | F | 50 |

Figure 4: Data Distribution via Sharding (Horizontal Partitioning)

- **Operational Features** [20]–[23]
  - **High Availability**: The system must minimise downtime and provide continuous service availability.
  - **Horizontal Scalability for Performance**: The system should support replication-based scaling to distribute read workloads across multiple nodes, particularly important for applications with varying traffic patterns.
  - **Data Model Flexibility**: The system must efficiently support either relational or document data models, allowing applications to choose the most appropriate representation for their specific use cases.
- **Performance Features** [5]
  - **Fast Read Operations**: Given that many OLTP applications exhibit read-heavy characteristics, the system must optimise for rapid data retrieval operations.
  - **Fast Write Operations**: The system must efficiently handle data modification operations without significantly impacting concurrent read performance.
  - **Mixed Workload Support**: The system should maintain consistent performance across varying ratios of read and write operations.

### C. Systematic Assessment Methodology

Although we do not consider our efforts as an exclusive exercise in benchmarking, in our methodology we adopt a strict approach and follow several objectives that are typical of benchmarking. Benchmark design is characterised by several key objectives [24], including:

1) **Relevance**: the assessment should interact with the system under test (SUT) in a manner that is realistic, and thus in-line with the typical interaction that the SUT should expect in a real-world deployment.
2) **Repeatability**: the assessment process should aim for a level of confidence that running the same assessment multiple times would yield similar results.
3) **Fairness**: the assessment should be specific to the domain at hand. An example of an unfair assessment is one that measures the performance of complex queries in RDBMS to elicit a conclusion that a distributed file system has poor performance. Fair assessments also look at multiple qualities of a system under test: for example, focusing solely on performance quality when comparing systems that are functionally different is not considered a fair assessment.
4) **Portability**: it should be possible to execute the assessment against different systems, which implies that one must carefully select the system features to examine, and find a balance between using a small subset of system features (which would render the benchmark design obsolete), and using system-specific or cutting-edge features, which may only be offered by a limited number of SUTs (reducing portability). This quality also emphasises affordability, in that a portable assessment does not necessarily require complex logic or expensive infrastructures.
5) **Understandability**: the assessment should seek to elicit meaningful metrics and the workload should be meaningful to the domain of the SUT.

Benchmarks depend on workload generators to raise requests to SUTs, and different types of workloads exist. Trace-based workloads are configured with a precise set of activities (typically extracted from monitors deployed on a live installation) that are replayed at runtime. In contrast, synthetic workloads generate artificial requests to the SUT, in a manner that the workload mix follows pre-defined probability distributions.

In general, trace-based workloads align better to the repeatability objective of benchmarking however, because they rely on execution traces from live system installations, they are exposed to several challenges including: a) traces may not be long enough to generalise benchmark results; b) it is difficult to obtain and share traces, especially due to security considerations; and c) traces may not include corner cases, preventing benchmarking from testing an SUT's behaviour in atypical situations.

Synthetic workloads can be used to overcome the challenges of trace-based workloads, and several techniques are used to improve their alignment to the benchmarking objectives. For example, a synthetic workload can be designed based on observations from live applications, from which realistic probability distributions that fit live workloads are elicited, hence aligning to the relevance benchmarking objective. Furthermore, one can increase the probability that the actual workload generated aligns to the workload distribution required by running a synthetic workload over a longer period, hence improving the aspect of repeatability. Furthermore, benchmark results are assessed based on repeated iterations to improve repeatability by, for example, minimising the impact of transient external events (e.g., transient events in a cloud infrastructure). Consequently, several studies report average values based on the throughput of three benchmark workload executions [9], [25].

Synthetic workloads are a popular choice in diverse benchmarking studies, including benchmarks for runtime verification [26], for serverless cloud computing [27], and programming frameworks [28]. Synthetic workloads are also prevalent in DBMS benchmarks. The Transaction Processing Performance Council (TPC) puts forward the specifications of several synthetic benchmarks, each modelled differently to remain relevant and representative for different domains. For example, the TPC-C benchmark [29] is widely used to assess the performance of transactional DBMSs [30]. The Yahoo! Cloud Serving Benchmark [5], or YCSB, is a synthetic workload generator, built on the basis of observations of typical web serving use cases at Yahoo!, and is used to study the performance of cloud data serving systems, such as DDBMSs [31]–[34].

## III. Implementation

### A. Requirements and Systems Analysis

This experiment addresses the following key questions:

1) How do PostgreSQL and MongoDB compare in terms of transaction throughput for document-oriented Cloud OLTP workloads across different read/write ratios?
2) What are the latency[3] characteristics of read and write operations for both systems under varying concurrent load conditions?
3) How do system metrics scale with increasing dataset sizes and concurrent user loads for both PostgreSQL and MongoDB?
4) Under what specific workload conditions does each system demonstrate superior throughput, and what factors contribute to these differences?
5) What practical considerations should guide the selection between PostgreSQL and MongoDB for applications with similar functional requirements?

## B. Scope and Limitations

This exercise focuses specifically on document-oriented workloads using the document data model capabilities of both systems. The evaluation encompasses varying workload compositions (read/write ratios), dataset sizes, and concurrent user loads using standardised benchmarking methodologies.

The experiment is constrained to specific versions of PostgreSQL and MongoDB and to a single configuration of each DBMS deployed in cloud-based Database-as-a-Service (DBaaS) environments. Various other valid configurations for each DBMS were beyond the scope of this study. The evaluation uses synthetic workloads generated by the YCSB framework [5], which may not capture all nuances of real-world application patterns. Additionally, the study focuses on performance metrics and does not extensively evaluate factors such as administrative complexity, development productivity, or long-term maintenance costs.

## C. Assessment Objectives

This study aims to address specific research objectives. The primary objective is to conduct a comprehensive empirical performance comparison between PostgreSQL and MongoDB for document-oriented Cloud OLTP workloads, evaluating their relative strengths and weaknesses across different operational scenarios. Other secondary objectives include:

1) **Throughput Analysis**: Quantify the transaction processing capabilities of both systems under varying workload compositions and concurrent user loads.
2) **Latency Characterisation**: Measure and compare the response time characteristics for both read and write operations across different system configurations.
3) **Scalability Assessment**: Evaluate how both systems perform as dataset sizes and concurrent user loads increase.
4) **Workload Sensitivity Analysis**: Determine how changes in read/write ratios affect the relative performance of each system.
5) **Practical Guidance Development**: Provide evidence-based recommendations for system architects selecting between these technologies for specific application contexts. We thus aim to carry out this exercise in a way that could prove useful and familiar to system architects faced with the question “Which DBMS should I use?”.

[3]We define *latency* as the time elapsed between when a client submits an operation to when the result of that operation arrives back at the client

## D. Methodology Overview

This empirical analysis adopts a rigorous experimental approach to evaluate the comparative performance of PostgreSQL and MongoDB for document-oriented Cloud OLTP workloads, and is carried out in the following context:

- The analysis focuses on the Document data model. This is the only data model that is natively supported by both SUTs and thus, in the spirit of fairness, is the only one considered.
- Analysis is performed using the YCSB tool, which has been used in other DBMS empirical performance analyses [35], [36]. YCSB already supports PostgreSQL and MongoDB. However, for PostgreSQL, it provided support for simulating a workload by a single client (i.e., thread). For our use case, we extended YCSB with a client-side connection pooler, and in doing so enabled the simulation of workloads via multiple clients. Our extension was also proposed on the official YCSB GitHub repository [4].
- The SUTs were deployed in DBaaS mode using Aiven for PostgreSQL and Atlas for MongoDB.
- MongoDB clients connected with the majority write concern, in an effort to approximate the durability guarantees of PostgreSQL.

## E. System Under Test Configuration

Figure 5 illustrates the SUT infrastructure setup used for this empirical performance evaluation. The configuration of the two DBMSs chosen for evaluation is given in Table I.

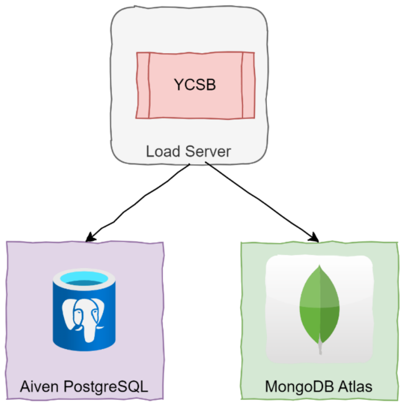


Figure 5: SUT Infrastructure Setup

Both systems were deployed in Microsoft Azure’s France Central region to ensure consistent network latency and infrastructure characteristics. The DBaaS approach was chosen to reflect realistic production deployment scenarios, aligned with our *practical guidance development* objective, while

[4]https://github.com/brianfrankcooper/YCSB/pull/1709, Oct 18, 2025

TABLE I: Database Management System Specifications

| **Specification** | **PostgreSQL** | **MongoDB** |
|---|---|---|
| Version | PostgreSQL v16.2 | MongoDB v7.0 |
| Deployment | Aiven PostgreSQL (DBaaS) | MongoDB Atlas (DBaaS) |
| Service Plan | Business-64 | M60 (low CPU optimisation) |
| Configuration | 2 servers (primary-secondary) | 3 servers (1 primary, 2 replicas) |
| Hardware | 8 vCPUs, 64 GiB RAM/server | 8 CPU cores, 64GB RAM/server |
| Storage | Premium SSD v1, 1000GB | 1024GB, 5000 IOPS (192 MB/s) |
| Connection Limit | 1000 | 32000 |
| Network Performance | – | Extremely High |
| Backup | – | Continuous Cloud Backup |

eliminating infrastructure management variables that could affect performance measurements.

The synthetic workload generation was performed using a dedicated load server with the following specifications:

- Cloud Platform: Microsoft Azure
- Machine Type: D32s_v5
- CPU: 32 vCPUs
- Memory: 128GB RAM
- Storage: 80GB SSD
- Operating System: Linux Ubuntu
- Network: Co-located in France Central region

This configuration provided sufficient resources to generate high-concurrency workloads without causing client-side bottlenecks that could skew performance measurements.

### F. Empirical Performance Evaluation Framework

The characteristics of the dataset generated by YCSB were:

- Data Model: Document-based (JSON format)
- Record Count: 1,000,000 documents for database seeding
- Document Structure: Semi-structured data typical of document-oriented applications, generated using the default YCSB data load generator (i.e., JSON documents with ten properties of 100 bytes each).
- Key Distribution: Zipfian distribution to simulate realistic access patterns

To ensure fair evaluation, the standard YCSB framework was extended with the following enhancements:

- Client-side connection pooling for PostgreSQL to enable multi-threaded workload simulation
- Optimised database drivers for both systems
- Enhanced metrics collection for latency analysis

The PostgreSQL extension addressing single-client limitations was contributed to the official YCSB repository[5], ensuring reproducibility and community benefit.

### G. Experimental Design

The experimental design evaluated system performance across multiple workload dimensions to capture realistic application scenarios:

**Read/Write Ratios:** Four distinct workload compositions were tested to represent different application characteristics:

- 100% READ / 0% WRITE: Read-only analytical queries
- 95% READ / 5% WRITE: Read-heavy with occasional updates
- 80% READ / 20% WRITE: Balanced read-write workload
- 70% READ / 30% WRITE: Write-intensive transactional workload

**Dataset Scaling:** Performance evaluation was conducted across four dataset sizes to assess scalability characteristics:

- 10,000 records: Small dataset (baseline)
- 100,000 records: Medium dataset
- 500,000 records: Large dataset
- 1,000,000 records: Maximum dataset size

**Concurrency Levels:** Five concurrent user load levels were tested to evaluate system behaviour under varying contention, defined by virtual users (vUsers):

- 60 vUsers: Low concurrency (baseline)
- 120 vUsers: Moderate concurrency
- 240 vUsers: High concurrency
- 480 vUsers: Very high concurrency
- 960 vUsers: Maximum concurrency stress test

The complete experimental design resulted in 80 unique scenarios per database system (i.e., 4 workload types × 4 dataset sizes × 5 concurrency levels). Each scenario was executed three times to ensure measurement consistency and reduce the impact of transient system variations, resulting in 240 individual test runs per system and 480 total benchmark executions.

Each benchmark run was conducted for a duration of 60 seconds to allow for system stabilisation and meaningful performance measurement collection. The total experimental execution required approximately eight hours of continuous benchmarking.

### H. Database-Specific Configurations

PostgreSQL was configured to optimise document-oriented workload performance while maintaining ACID compliance. **Data storage** uses native JSONB format for document storage and indexing. **Connection Management** uses built-in connection pooling with optimised parameters. The **Consistency Level** retains full ACID compliance with default Read Committed isolation level. **Replication** to read replicas is asynchronous.

MongoDB was also configured with production-recommended settings for transactional workloads. **Write Concern** was configured as Majority write for durability

[5]https://github.com/brianfrankcooper/YCSB/pull/1709, Oct 18 2025

guarantees. **Read Preference** is given to the Primary for consistency. **Connection Management** uses native connection pooling and multiplexing. **Replication** to the replica set requires majority acknowledgment. The majority write concern configuration was specifically chosen to approximate the durability and consistency guarantees provided by PostgreSQL, ensuring fair and meaningful comparison between systems.

We acknowledge that both DBMSs chosen for this exercise are intrinsically very different technologies, and as such it is not trivial to achieve a precise like-for-like configuration. However, we chose a configuration for each DBMS that is considered familiar to system architects and application developers, in line with our objectives.

### *I. Metrics Collection*

Performance evaluation focused on two primary metrics categories. **Throughput Metrics** include Operations per second (overall system throughput), Read operations per second, Write operations per second and Transaction completion rates. **Latency Metrics** include Average response time for read operations, Average response time for write operations, 95th percentile latency measurements and Maximum observed latencies under load. All metrics were collected using YCSB's built-in measurement framework, with additional custom instrumentation for detailed latency analysis across different concurrency levels and workload compositions.

### *J. Data Analysis Approach*

Performance analysis employed comparative statistical methods to identify significant performance differences between systems. Results were aggregated across multiple test runs to ensure statistical reliability, with percentage-based comparisons used to normalise performance differences across varying absolute throughput levels.

The analysis framework generated comprehensive performance profiles for each system across all tested scenarios, enabling identification of workload-specific performance characteristics and optimal use case recommendations.

## IV. RESULTS

The empirical performance evaluation across 80 distinct scenarios per database system reveals significant performance variations between PostgreSQL and MongoDB under different workload conditions. Figure 6 presents a summary of results as percentage differences in average throughput between MongoDB and PostgreSQL, where positive values indicate MongoDB superiority and negative values indicate PostgreSQL superiority. Figures 7, 8, 9, 10, and 11 illustrate in detail the performance profile of both SUTs under each scenario.

The empirical results demonstrate that PostgreSQL consistently outperforms MongoDB in the majority of tested scenarios, particularly excelling in mixed read-write workloads. However, MongoDB exhibits competitive advantages in specific operational contexts, notably under write-heavy workloads with maximum server saturation and read-heavy workloads under low concurrent user loads.

### *A. Throughput Analysis*

*1) Overall Performance Characteristics:* Across all tested scenarios, PostgreSQL demonstrates superior throughput performance in approximately 75% of cases, with performance advantages ranging from 20% to 40% compared to MongoDB. This performance superiority is most pronounced in scenarios involving mixed workload compositions, where PostgreSQL's optimised query execution and transaction management provide significant advantages.

The performance difference between systems varies substantially based on workload characteristics:

- **Read-Only Workloads (100% READ)**: PostgreSQL shows 15-35% better performance under medium to high concurrency levels (i.e., 240-960 virtual users)
- **Read-Heavy Workloads (95% READ/5% WRITE)**: PostgreSQL maintains 10-30% performance advantage across most scenarios
- **Balanced Workloads (80% READ/20% WRITE)**: PostgreSQL demonstrates 20-40% superior performance consistently
- **Write-Intensive Workloads (70% READ/30% WRITE)**: PostgreSQL shows 15-25% better performance, with exceptions under maximum load conditions

*2) MongoDB Performance Advantages:* MongoDB demonstrates competitive advantages in specific scenarios:

**Write-Heavy, High-Concurrency Scenarios:** MongoDB exhibits approximately 20% better throughput performance compared to PostgreSQL under write-intensive workloads (i.e., 70% READ/30% WRITE) when operating at maximum concurrent user loads (960 virtual users). This advantage likely stems from MongoDB's optimised write handling and connection management under high contention scenarios.

**Read-Heavy, Low-Concurrency Scenarios:** Under read-dominated workloads (i.e., 100% READ) with low concurrent user loads (i.e., 60-120 virtual users), MongoDB demonstrates 30-40% superior performance compared to PostgreSQL. This advantage diminishes as concurrency increases, suggesting that MongoDB's document-oriented query processing provides benefits primarily under low-contention conditions.

### *B. Latency Analysis*

*1) Read Operation Latency:* The latency analysis reveals distinct performance characteristics for read operations between the two SUTs:

**MongoDB Read Performance:** MongoDB consistently demonstrates superior read operation latency across all tested scenarios, with average improvements of approximately 5ms compared to PostgreSQL. This advantage remains relatively stable across different concurrency levels and dataset sizes, suggesting fundamental differences in document retrieval optimisation.

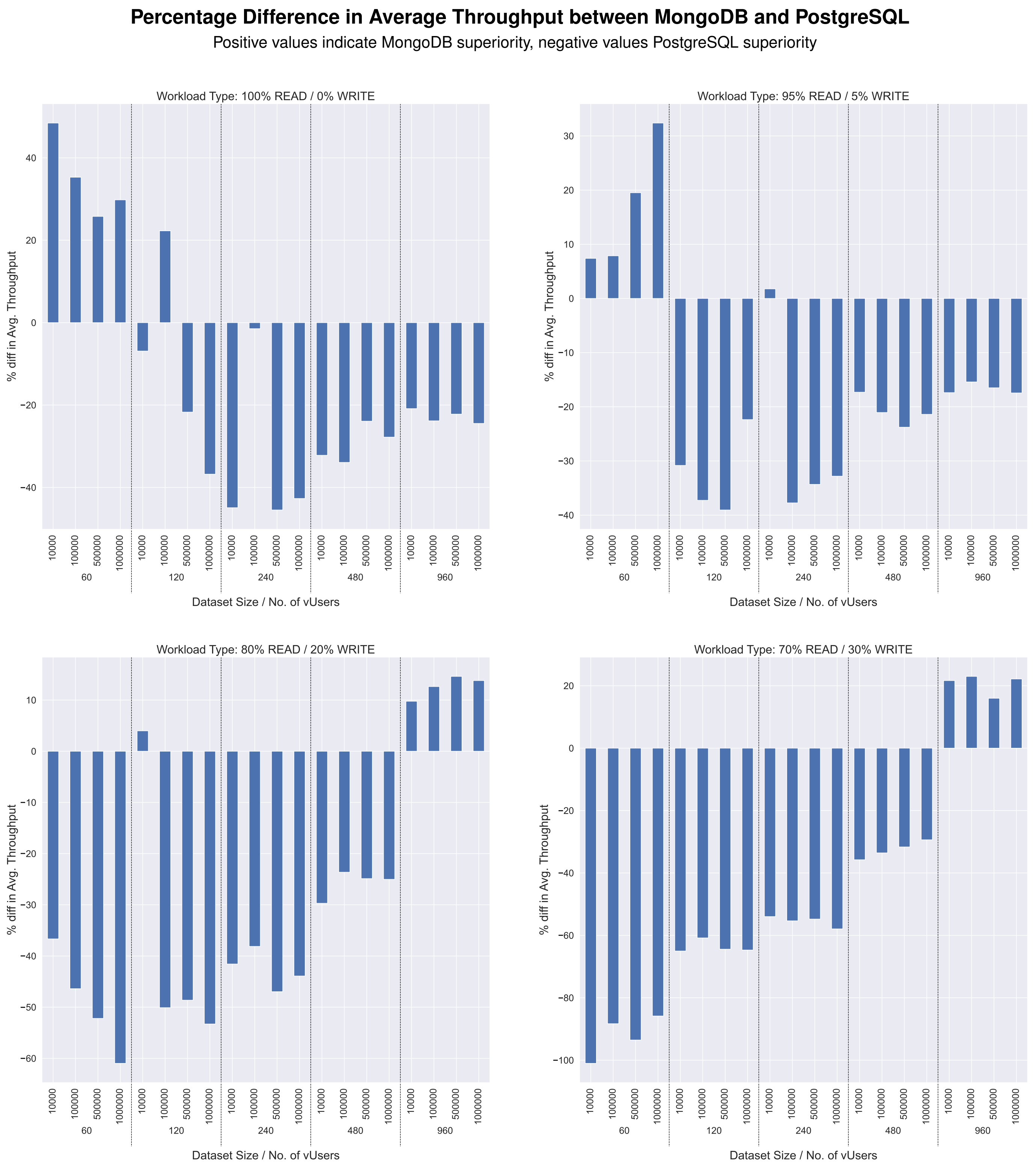


Figure 6: Summary of results as a Percentage Difference in Average Throughput between MongoDB and PostgreSQL with different workloads, dataset sizes and number of virtual users

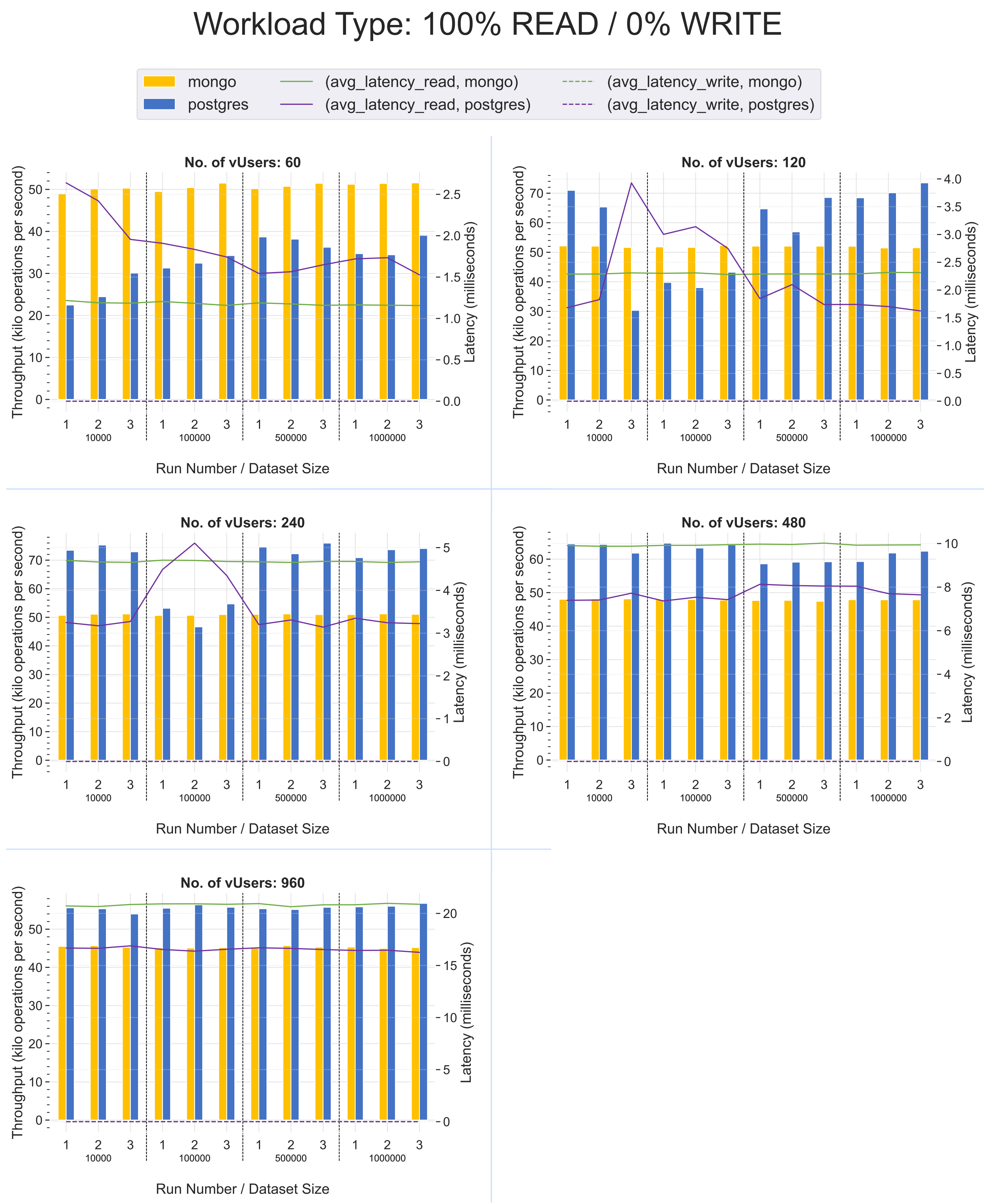


Figure 7: Detailed Throughput and Latency Results for the scenario 100% READ/ 0% WRITE

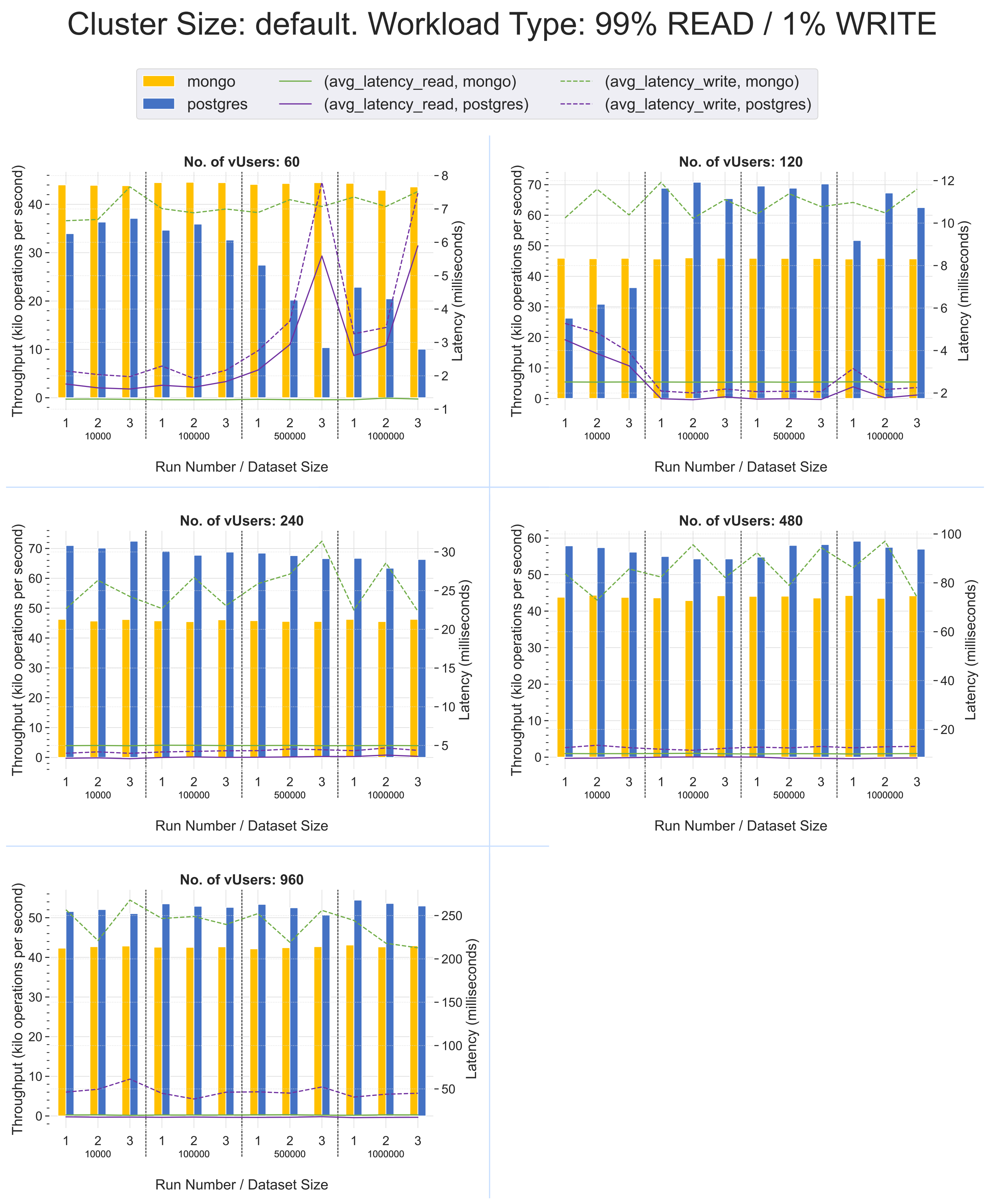


Figure 8: Detailed Throughput and Latency Results for the scenario 99% READ/ 1% WRITE

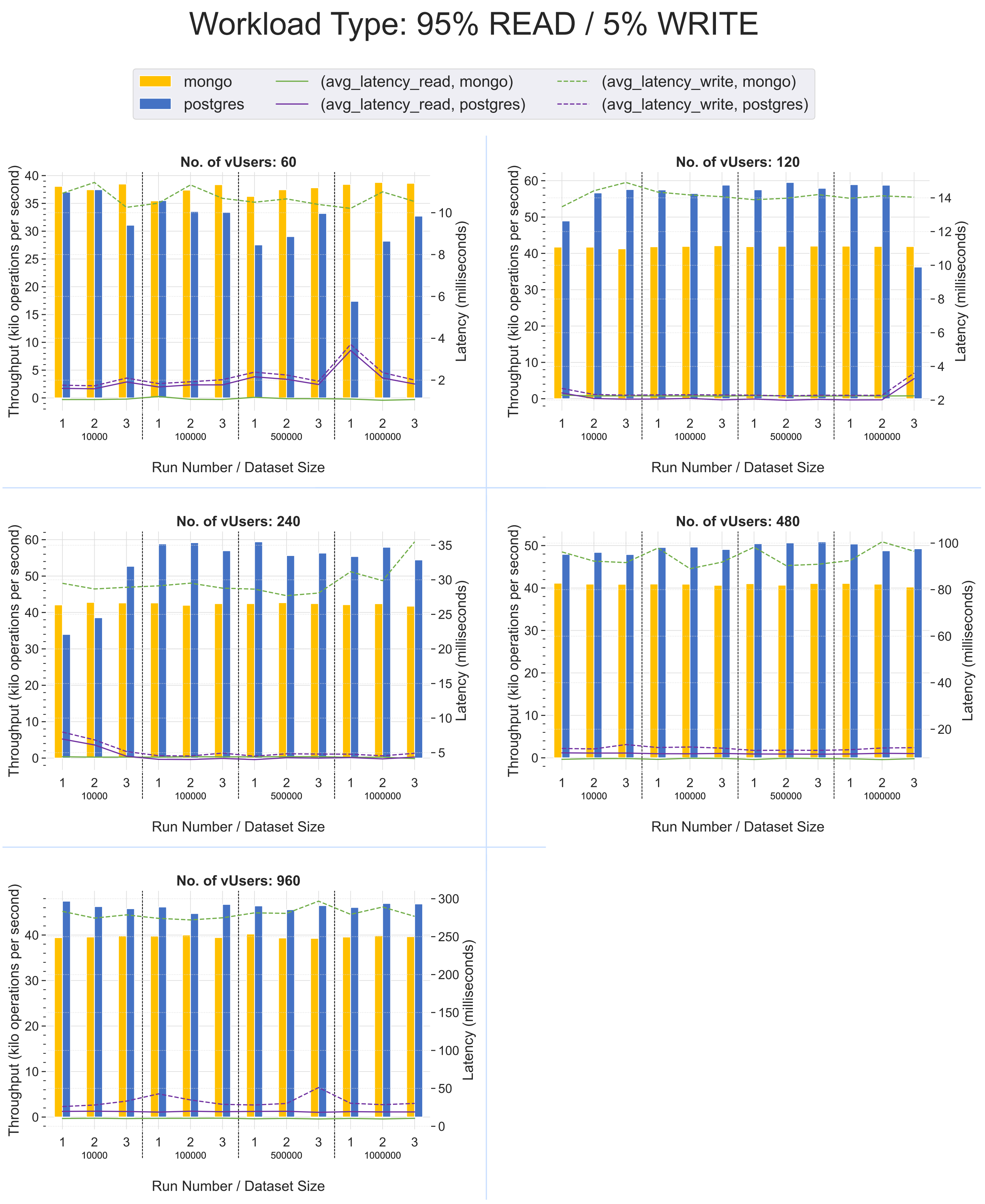


Figure 9: Detailed Throughput and Latency Results for the scenario 95% READ/ 5% WRITE

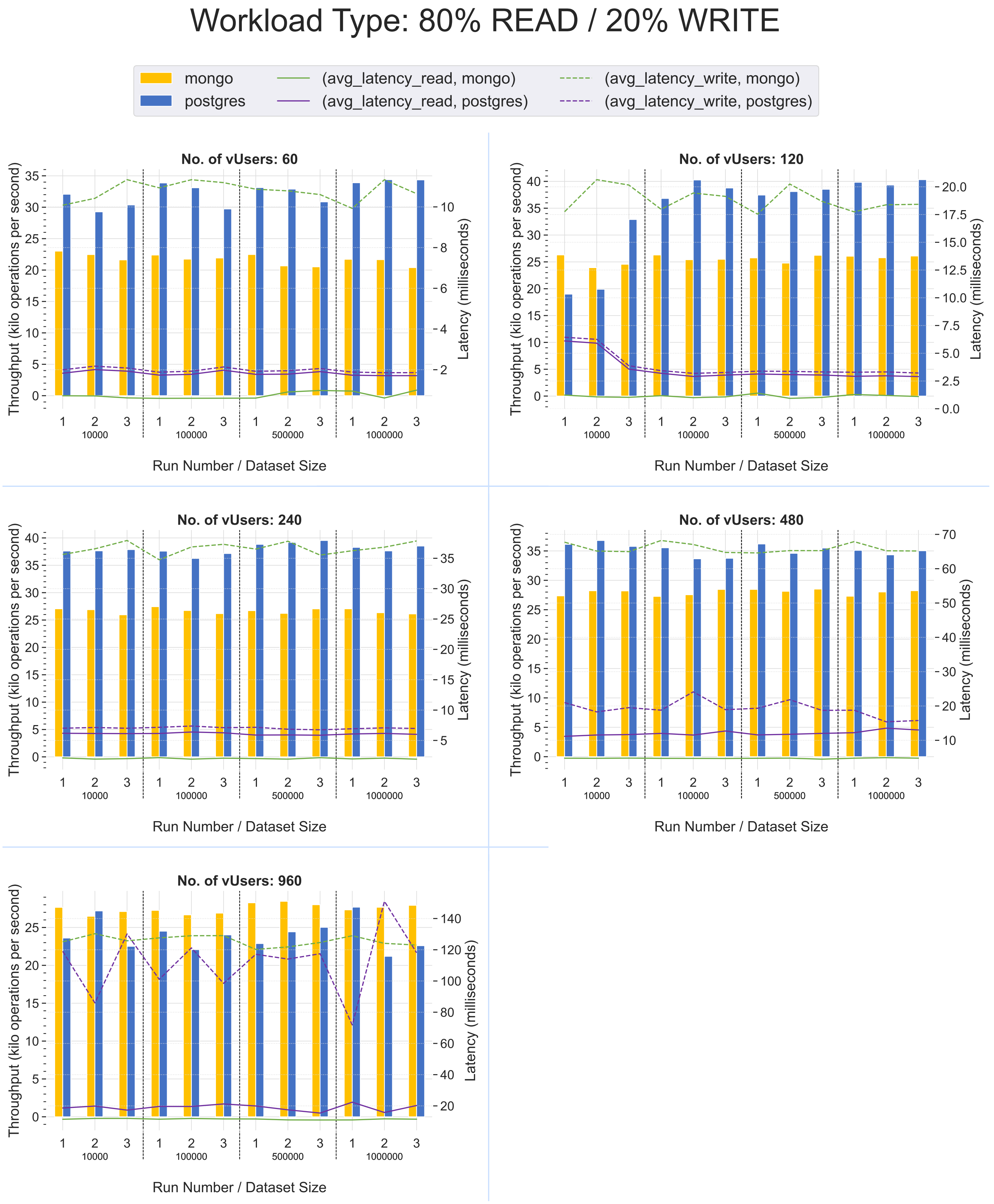


Figure 10: Detailed Throughput and Latency Results for the scenario 80% READ/ 20% WRITE

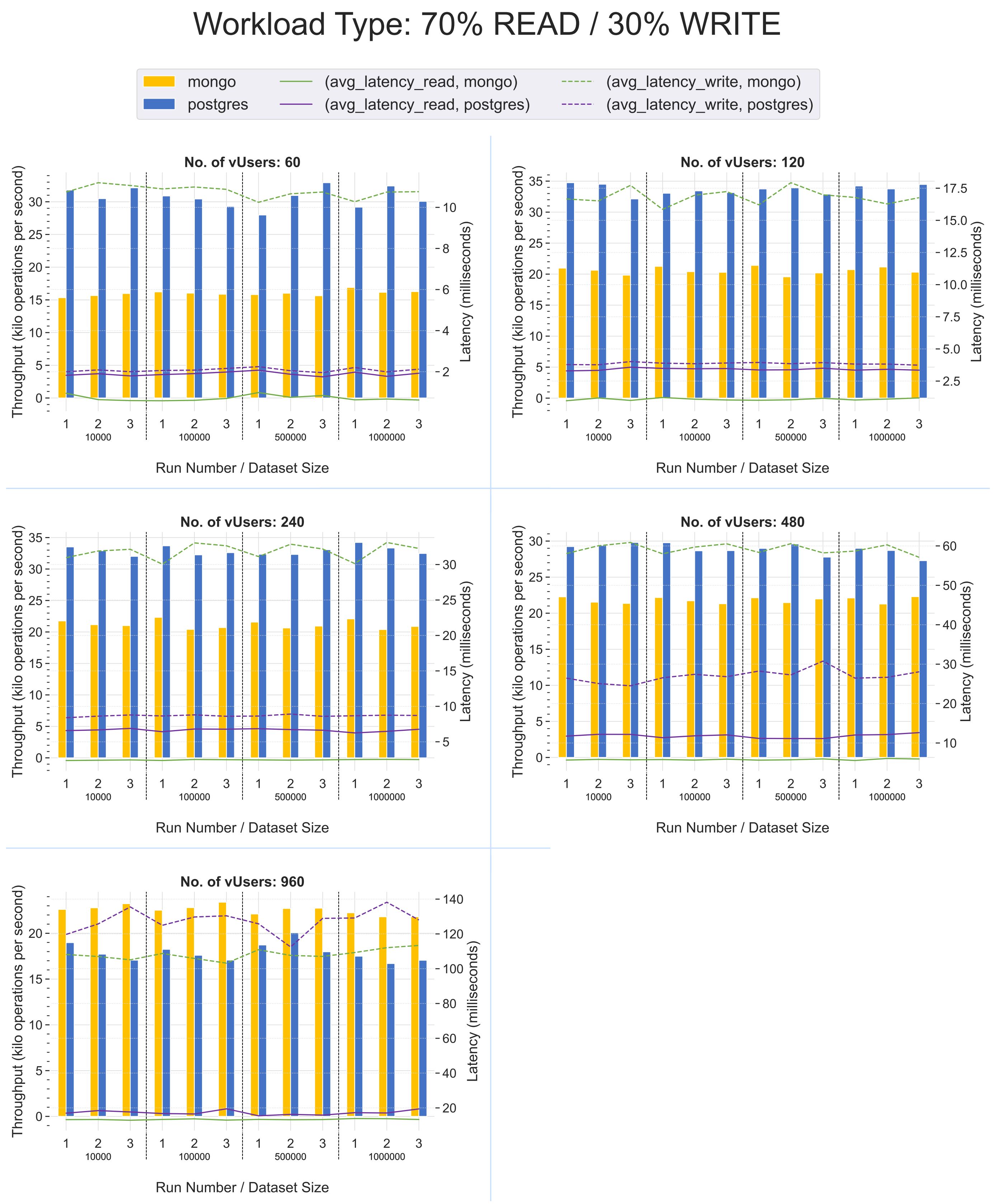


Figure 11: Detailed Throughput and Latency Results for the scenario 70% READ/ 30% WRITE

**PostgreSQL Read Performance:** Although PostgreSQL exhibits slightly higher read latencies than MongoDB, the difference remains minimal (5ms average) and does not significantly impact overall application responsiveness in most practical scenarios.

*2) Write Operation Latency:* Write operation performance reveals the most significant performance difference between the systems.

**PostgreSQL Write Performance:** PostgreSQL demonstrates substantially superior write operation performance across all tested scenarios. Write latencies remain consistently low even under high concurrent loads, maintaining sub-50ms response times in most configurations.

**MongoDB Write Performance:** MongoDB exhibits higher write operation latencies, particularly under high concurrency conditions. The most severe performance degradation occurs in write-intensive scenarios with maximum concurrent users, where MongoDB write latencies can exceed PostgreSQL by up to 250ms. For example, in the 95% READ/5% WRITE scenario with 960 virtual users, MongoDB write operations average 250ms longer than equivalent PostgreSQL operations.

## C. Scalability Analysis

*1) Dataset Size Scaling:* Both systems demonstrate reasonable scalability characteristics as dataset sizes increase from 10,000 to 1,000,000 records, whilst using the same infrastructure and DBMS configuration.

**PostgreSQL Scaling:** PostgreSQL maintains consistent performance across dataset size variations, with throughput degradation remaining below 15% as dataset size increases by two orders of magnitude. This consistent performance suggests effective query optimisation and indexing strategies for document-based operations.

**MongoDB Scaling:** MongoDB exhibits similar dataset scaling characteristics, with performance degradation comparable to PostgreSQL across different dataset sizes. However, the absolute performance remains lower than PostgreSQL in most scenarios.

*2) Concurrency Scaling:* The concurrent user load analysis reveals different scaling patterns for each system:

**PostgreSQL Concurrency Handling:** PostgreSQL demonstrates consistent performance scaling as concurrent user loads increase, maintaining stable throughput levels even at maximum concurrency (i.e., 960 virtual users). The system exhibits graceful degradation without performance cliffs or severe bottlenecks.

**MongoDB Concurrency Handling:** MongoDB shows more variable performance under increasing concurrency loads. While the system handles low to moderate concurrency effectively, performance becomes inconsistent at high concurrency levels (i.e., throughput does not increase when concurrency increases), particularly for write-intensive operations.

## D. Workload Composition Sensitivity

*1) Read/Write Ratio Impact:* The analysis reveals significant sensitivity to workload composition for both systems:

**Read-Dominant Workloads:** As read operations dominate the workload (moving from 70% to 100% read operations), MongoDB's relative performance improves, particularly under low concurrency conditions. However, PostgreSQL maintains superior absolute performance in most scenarios.

**Write-Intensive Workloads:** Increasing write operation percentages generally favour PostgreSQL due to its superior write latency characteristics. The performance gap between systems widens as write operations become more prevalent, except under maximum concurrency conditions where MongoDB shows some advantages.

## E. Statistics Reporting

All reported performance differences are based on the mean of three independent benchmark runs per scenario, improving reporting reliability. The consistent patterns observed across multiple test runs and scenario variations provide confidence in the robustness of the observed performance characteristics.

The most significant findings, including PostgreSQL's 20-40% throughput advantages and MongoDB's 250ms write latency penalties under high concurrency, were consistently observed across all repetitions, indicating systematic rather than random performance differences.

## F. Performance Thresholds

Both database systems demonstrated the capability to handle substantial transaction loads:

- **Maximum Observed Throughput**: Both systems achieved over 20,000 transactions per second under optimal conditions
- **Sustained Performance**: Both systems maintained over 15,000 transactions per second across most scenarios
- **Baseline Performance**: Even under adverse conditions, both systems sustained minimum throughput levels exceeding 10,000 transactions per second

These performance levels significantly exceed typical application requirements and demonstrate the suitability of both systems for high-throughput Cloud OLTP applications.

## G. Summary of Key Findings

The comprehensive performance evaluation yields the following primary observations:

1) **PostgreSQL General Superiority**: PostgreSQL demonstrates superior performance in approximately 75% of tested scenarios, with advantages ranging from 20-40%
2) **MongoDB Niche Advantages**: MongoDB shows competitive performance in read-heavy, low-concurrency scenarios (30-40% advantage) and write-heavy, high-concurrency scenarios (20% advantage)
3) **Latency Trade-offs**: MongoDB offers superior read latency (5ms improvement) while PostgreSQL provides substantially better write latency (up to 250ms improvement under high load)
4) **Scalability Characteristics**: Both systems demonstrate reasonable scaling behaviour across dataset sizes and concurrency levels

5) **High Absolute Performance**: Both systems exceed 20,000 transactions per second, indicating suitability for demanding Cloud OLTP applications

## V. Conclusion

This study identified the intricacies of delivering the QoS and QoD required by users of an IS sustaining a transactional workload, under the characteristics of Cloud OLTP [5].

We discussed several of the many characteristics of a DBMS that have an important bearing on the choice of product for a particular use-case. We then walked through the use-case considered in our problem domain, and briefly described how and why MongoDB and PostgreSQL are a good fit. Lastly, we detailed a thorough performance comparison of a representative setup of these two DBMSs, using a workload suitable for the document data model. Results for this comparison exercise show that PostgreSQL outperforms MongoDB in the general case, but MongoDB performs faster in other cases.

This exercise brings forth several observations, including that:

- An optimal choice of DBMS technology depends on a thorough understanding of the problem domain at hand, ensuring that the DBMS delivers the necessary functions to support the dataset, the operations that will be performed on it, and the system and business requirements that need to be satisfied.
- NoSQL DBMSs are not necessarily faster than SQL-based DBMSs in all cases. Our empirical analysis has shown that both MongoDB and PostgreSQL can meet stringent QoS and QoD requirements.
- The data model should not be the only deciding factor as to whether to use an SQL-based or NoSQL DBMS: many SQL-based DBMSs support multiple data models, including the document data model used by NoSQL DBMSs.
- Deploying a distributed DBMS requires careful thought as to which level of data consistency and durability guarantees are needed and expected from the DBMS. For example, MongoDB does not guarantee that cross-document changes occurring in the master node are replicated in the same order to read-only replicas[6], whilst PostgreSQL's replication methodology based on the write-ahead log (WAL) does provide these guarantees[7].

Nonetheless, the rigorous approach of this effort is not considered to constitute a thorough analysis of this non-trivial topic. Notably, the comparison was done on a single, representative setup of PostgreSQL and MongoDB available at the time of writing. Similar exercises can be carried out on several other setups, and we identify five areas that represent limitations of this study and where future research efforts can be directed:

[6]https://www.mongodb.com/docs/manual/core/replica-set-sync/#multithreaded-replication, accessed Nov 9, 2025

[7]https://www.postgresql.org/docs/current/protocol-replication.html, accessed Nov 9, 2025

1) *Configuration Scope:* This study examined one representative configuration per system. Future work should explore the impact of different consistency levels, storage engines, and optimisation parameters, e.g., different configurations of READ and WRITE concerns for MongoDB, the impact of READs handled by slave nodes and WRITEs handled by master nodes and weaker data consistency levels such as Eventual Consistency and Causal Consistency [37]–[39].
2) *Workload Patterns:* YCSB workloads, while standard, may not capture all real-world access patterns. Industry-specific benchmarks (e.g., e-Commerce, IoT telemetry) would introduce workload and dataset variety (e.g., varying amounts of data records, varying document sizes) and thus broaden applicability.
3) *Version Evolution:* DBMSs evolve rapidly, so performance comparisons should be conducted periodically to track performance evolution across versions.
4) *Hardware Diversity:* Cloud-based evaluation provides consistency but may not reflect performance on specialised hardware configurations.
5) *Operational Complexity:* Performance is only one dimension of database selection. Future framework extensions should incorporate operational metrics, development productivity, and total cost of ownership.

## References

[1] Y. Lee, A. N. Chen, and V. Ilie, "Can online wait be managed? The effect of filler interfaces and presentation modes on perceived waiting time online," *MIS Quarterly*, pp. 365–394, 2012.

[2] S. Papastavrou, P. K. Chrysanthis, and G. Samaras, "Performance vs. freshness in web database applications," *World wide web*, vol. 17, no. 5, pp. 969–995, 2014.

[3] C. Humby, "Data is the new oil," *Proc. ANA Sr. Marketer's Summit. Evanston, IL, USA*, vol. 1, 2006.

[4] M. Xiaolei, L. Sen, and Y. Xiaofei, "Traffic Data Management Technology in ITS," *Intelligent Road Transport Systems*, p. 97, 2022.

[5] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, "Benchmarking cloud serving systems with YCSB," in *Proceedings of the 1st ACM Symposium on Cloud Computing*, ACM, 2010, pp. 143–154.

[6] M. Salahuddin, S. Majeed, S. Hira, and G. Mumtaz, "A Systematic Literature Review on Performance Evaluation of SQL and NoSQL Database Architectures," *Journal of Computing & Biomedical Informatics*, vol. 7, no. 02, 2024.

[7] A. Makris, K. Tserpes, G. Spiliopoulos, D. Zissis, and D. Anagnostopoulos, "MongoDB Vs PostgreSQL: A comparative study on performance aspects," *GeoInformatica*, vol. 25, no. 2, pp. 243–268, 2021.

[8] T. Taipalus, "Database management system performance comparisons: A systematic literature review," *Journal of Systems and Software*, vol. 208, p. 111 872, 2024.

[9] J. Klein, I. Gorton, N. Ernst, P. Donohoe, K. Pham, and C. Matser, "Performance evaluation of NoSQL databases: a case study," in *Proceedings of the 1st Workshop on Performance Analysis of Big Data Systems*, 2015, pp. 5–10.

[10] M. Bach and A. Werner, "Hybrid column/row-oriented DBMS," in *Man–Machine Interactions 4: 4th International Conference on Man–Machine Interactions, ICMMI 2015 Kocierz Pass, Poland, October 6–9, 2015*, Springer, 2015, pp. 697–707.

[11] C. Camilleri, J. G. Vella, and V. Nezval, "HTAP With Reactive Streaming ETL," *Journal of Cases on Information Technology (JCIT)*, vol. 23, no. 4, pp. 1–19, 2021.

[12] E. F. Codd, "A relational model of data for large shared data banks," *Communications of the ACM*, vol. 13, no. 6, pp. 377–387, 1970.

[13] H. Hashem and D. Ranc, "Evaluating NoSQL document oriented data model," in *IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, IEEE, 2016, pp. 51–56.

[14] *ISO 9075:1987 - information processing systems — database language — SQL*, https://www.iso.org/standard/16661.html, Accessed: 21-May-2024, 1987.

[15] C. Strozzi, "NoSQL: A relational database management system," *Lainattu*, vol. 5, p. 2014, 1998.

[16] C. Camilleri, J. G. Vella, and V. Nezval, "D-Thespis: A Distributed Actor-Based Causally Consistent DBMS," in *Transactions on Large-Scale Data-and Knowledge-Centered Systems LIII*, Springer, 2023, pp. 126–165.

[17] DoubleClick, ShopWiki, and GiltGroupe, *Mongodb*, https://www.mongodb.org, https://www.mongodb.org, 2007.

[18] R. Elmasri and S. Navathe, *Fundamentals of Database Systems*, 7th ed. Pearson/Addison Wesley, 2017.

[19] A. Thomson, T. Diamond, S.-C. Weng, K. Ren, P. Shao, and D. J. Abadi, "Fast distributed transactions and strongly consistent replication for OLTP database systems," *ACM Transactions on Database Systems (TODS)*, vol. 39, no. 2, pp. 1–39, 2014.

[20] K. Domdouzis, P. Lake, and P. Crowther, *Concise guide to databases: A practical introduction*. Springer, 2021.

[21] P. Lake and P. Crowther, "Database Availability," in *Concise Guide to Databases: A Practical Introduction*, Springer, 2013, pp. 221–239.

[22] Z. H. Liu and D. Gawlick, "Management of Flexible Schema Data in RDBMSs-Opportunities and Limitations for NoSQL," in *CIDR*, 2015.

[23] C. Camilleri, J. G. Vella, and V. Nezval, "Thespis: Causally-consistent OLTP," in *2021 16th Conference on Computer Science and Intelligence Systems (FedCSIS)*, IEEE, 2021, pp. 261–269.

[24] D. Bermbach, E. Wittern, and S. Tai, *Cloud service benchmarking*. Springer, 2017.

[25] S. Chowdhury and W. Golab, "A scalable recoverable skip list for persistent memory," in *Proceedings of the 33rd ACM Symposium on Parallelism in Algorithms and Architectures*, 2021, pp. 426–428.

[26] L. Aceto, D. P. Attard, A. Francalanza, and A. Ingólfsdóttir, "On benchmarking for concurrent runtime verification," in *International Conference on Fundamental Approaches to Software Engineering*, Springer International Publishing Cham, 2021, pp. 3–23.

[27] M. Copik, G. Kwasniewski, M. Besta, M. Podstawski, and T. Hoefler, "Sebs: A serverless benchmark suite for function-as-a-service computing," in *Proceedings of the 22nd International Middleware Conference*, 2021, pp. 64–78.

[28] C. Camilleri, J. G. Vella, and V. Nezval, "Actor model frameworks: An empirical performance analysis," in *International Conference on Information Systems and Management Science*, Springer, 2022, pp. 461–472.

[29] F. Raab, "TPC-C - The Standard Benchmark for Online Transaction Processing (OLTP)," in *The Benchmark Handbook for Database and Transaction Systems (2nd Edition)*, Morgan Kaufmann Publishers Inc, 1993.

[30] N. Crooks, M. Burke, E. Cecchetti, S. Harel, R. Agarwal, and L. Alvisi, "Obladi: Oblivious serializable transactions in the cloud," in *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, 2018, pp. 727–743.

[31] S. A. Mehdi, C. Littley, N. Crooks, L. Alvisi, N. Bronson, and W. Lloyd, "Not Causal! Scalable Causal Consistency with No Slowdown Cascades," in *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, 2017, pp. 453–468.

[32] K. Shudo and T. Yaguchi, "Causal consistency for distributed data stores and applications as they are," in *IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, IEEE, vol. 1, 2016, pp. 602–607.

[33] M. Zawirski, "Dependable eventual consistency with replicated data types," Ph.D. dissertation, Universite Pierre et Marie Curie, 2015.

[34] C. Camilleri, J. G. Vella, and V. Nezval, "Horizontally Scalable Implementation of a Distributed DBMS Delivering Causal Consistency via the Actor Model," *Electronics*, vol. 13, no. 17, p. 3367, 2024.

[35] S. Ferreira, J. Mendonça, B. Nogueira, W. Tiengo, and E. Andrade, "Benchmarking Consistency Levels of Cloud-Distributed NoSQL Databases Using YCSB," *IEEE Access*, 2025.

[36] N. B. Seghier and O. Kazar, "Performance benchmarking and comparison of NoSQL databases: Redis vs MongoDB vs Cassandra using YCSB tool," in *International Conference on Recent Advances in Mathematics and Informatics (ICRAMI)*, IEEE, 2021, pp. 1–6.

[37] C. Camilleri, J. G. Vella, and V. Nezval, "Thespis: Actor-Based Causal Consistency," in *2017 28th International Workshop on Database and Expert Systems Applications (DEXA)*, IEEE, Aug. 2017, pp. 42–46. DOI: 10.1109/DEXA.2017.25.

[38] C. Camilleri, J. G. Vella, and V. Nezval, "ThespisTRX: Causally-Consistent Read Transactions," *International Journal of Information Technology and Web Engineering (IJITWE)*, vol. 15, no. 1, pp. 1–16, 2020.

[39] C. Camilleri, J. G. Vella, and V. Nezval, "ThespisDIIP: Distributed Integrity Invariant Preservation," in *Database and Expert Systems Applications*, M. Elloumi, M. Granitzer, A. Hameurlain, C. Seifert, B. Stein, A. M. Tjoa, and R. Wagner, Eds., Cham: Springer International Publishing, 2018, pp. 21–37, ISBN: 978-3-319-99133-7.

# Investigation of Multiple Cognitive Biases in Military Contexts

Mark A. Livingston, Prithviraj Dasgupta, Jonathan W. Decker, and John Kliem
Information and Decision Science Branch
Naval Research Laboratory
Washington, DC, USA
e-mail: {mark.a.livingston18, prithviraj.dasgupta, jonathan.w.decker4, john.kliem3}.civ@us.navy.mil

*Abstract*—**We consider the problem of detecting cognitive biases in problems domains that are relevant to military personnel and roles they may have. In particular, we determined that anchoring bias, zero-risk bias, attraction effect, and compromise effect were relevant to military domains. In a user study, we hoped to elicit these biases and determine whether co-occurrence existed. We elicited anchoring bias in a time-extended task, but had limited success eliciting the other biases in small, disconnected scenarios. We did not observe co-occurrence of any of these four biases. We sought, but did not observe, whether visual presentation aids of text scenarios affected the presence of bias. We note some effects of user-identified strategies on biases.**

***Keywords- cognitive biases; anchoring bias; zero-risk bias; attraction effect; compromise effect; user study***

## I. Introduction

Decision-makers often rely on heuristic strategies, perhaps even without realizing it. These cognitive biases, or unstated bases for decisions, often lead to poor choices, including in military contexts [1], where poor decisions may lead to unnecessary loss of life. Cognitive biases in human decision-making while solving a problem are known to affect the outcome [2][3]. These biases usually degrade the outcome's value to the decision maker and to others that are affected by the problem's outcome. Researchers have proposed several techniques to detect biases.

Our main goal in this work was to develop scenarios, many based on military contexts, that would elicit hypothesized cognitive biases and determine what we could observe that may enable prediction of them. A second goal was to determine if these biases (when they occur) co-occur in individuals. Thus, we also collected information that we hoped would give predictive value for the existence or co-existence of our selected biases. Further, we hypothesize that visual presentation of information may mitigate bias, so we present our scenarios in text only and in text with illustrations of data presented in the text. On these last two questions, we hoped to make new research contributions.

The remainder of this paper is organized as follows. In Section II, we review literature in order to select biases to investigate and scenarios to further our investigation. We also describe differences our work introduces. In Section III, we describe the user study we used to gather data. In Section IV, we will present some data filtering we needed in order to conduct analysis. Our data analysis appears in Sections V and VI, reflecting the variety of forms of analysis we needed. Discussion appears in Section VII. We draw conclusions and recommend ideas for future work in Section VIII.

## II. Selection of Biases for Investigation

Dimara et al. [2] proposed a task-based taxonomy of 154 cognitive biases. We relied heavily on their taxonomy (mainly as expressed in their Table 2) to decide what biases we would try to elicit. Their list of tasks includes estimation, decision, hypothesis assessment, causal attribution, recall, opinion reporting, and other. We focused on the decision task, because we felt that was most relevant to the military domain that is our focus. They also used an intuitively-developed set of sub-categories as a second level in their taxonomy. This level consisted of *association* (cognition is biased by connections between items), *baseline* (cognition is biased by comparison with a baseline), *inertia* (cognition is biased by the prospect of changing the current state), *outcome* (cognition is biased by how well something fits a desired outcome), and *self perspective* (cognition is biased by a self-oriented view point).

Our initial idea was to examine biases made in the context of a strategy game, thinking this would be a good proxy for military tasks. This caused us to select one task from Dimara et al.'s [2] Estimation category, in the baseline sub-category: the anchoring effect. We felt that an initial solution (shown in a tutorial phase) to a simple game would elicit the effect. However, in order to isolate the biases from each other and be able to better control their elicitation, we opted for writing scenario sets, separately from the game, to elicit the other biases. We wanted something that we believed could be elicited and detected in a straightforward manner, but that would still be representative of decisions made in a variety of military contexts. This led us to select three other biases: the attraction effect (in the Decision/baseline portion of the taxonomy), the compromise effect (also in the Decision/baseline portion), and the zero-risk bias (in the Decision/association portion). The following subsections summarize research on these biases. In Section III, we describe how we created scenarios to (attempt to) elicit each of these biases.

### *A. Anchoring Bias*

Anchoring bias [3] causes humans to rely heavily on an initial piece of information, called an *anchor*. Because of this, humans tend to overlook information that would lead to better choices in subsequent decisions, and, instead, gravitate towards choices that align with the anchor. Initial research on analyzing anchoring biases focused on single-point decision problems. The main experimental design used for anchoring bias in such single-point decisions is the following: first, a decision maker is exposed to the anchor, about the likely outcome of a decision. Then, the decision maker is asked to make the same or a very similar decision. Anchoring bias is claimed to affect the latter decision if the latter decision's outcome is similar to the initial decision outcome. A canonical example is to anchor the decision maker to a price, e.g., 100 for a certain piece of clothing. Subsequently, the decision maker is shown a similar piece of clothing that is priced well below (or well above) 100, without revealing the price, and asked its worth. If the decision maker says that the clothing is worth around 100, it indicates that they are anchored to the initial price of 100.

Researchers [4][5] have reported the presence of anchoring bias in decision making for time-extended tasks (reviews of books and college applications). However, in these research studies, while making the decision for the current task the decision maker had access to the features of the current task, in addition to their experience from past decisions on similar tasks stored in their memories. In contrast, we ask 'If access to the current task's features while making the decision for the task were to be taken away, and the decision maker had to rely solely on experiences from memory from similar tasks to make decisions, is anchoring bias still present?' This question does not seem to have been investigated well in the literature.

These research settings are complementary to the research in this paper. The two main differences between our work and these are, first, we do not reveal the current problem's features (e.g., current book or college application under review) to the decision maker and the decision maker has to rely only on past task features and decisions from memory to make the current task's decision. Other slight distinctions are that these techniques use offline data that was not generated specifically for the bias studies and there was limited information about the background of the decision maker. On the other hand, the subjects in our study are people that were familiar with computer-game playing and decision-making in scenarios similar to our game. In addition, we report on a study that would have detected co-occurrence of other biases alongside an anchoring bias.

### *B. Attraction Effect*

Simonson [6] defines the *attraction effect* in a situation in which you have two alternatives and two dimensions that are important to your selection (Fig. 1). These two alternatives (Options 1 and 2 in Fig. 1) create a trade-off between the two dimensions of selection. In addition, a third choice (Option 3 in Fig. 1) has similar values to one of the first two alternatives, but is clearly weaker. This may imply weaker in both dimensions, or clearly weaker in one dimension

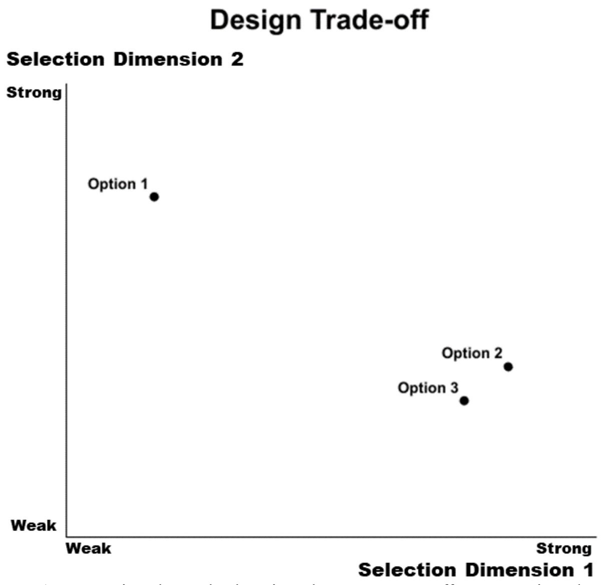


Figure 1. A notional graph showing the *attraction effect* occurring due to the relative placement of options along two dimensions of selection. Option 1 is sometimes known as the *target*; it is generally the best option or a competitor to the selection designer's choice. Option 3 should be seen as clearly inferior to Option 2, but even when Option 1 is objectively better than Option 2, the attraction effect leads a consumer to choose Option 2, because the inferior Option 3 attracts the user's attention.

while equal or even (very) slightly stronger in the other dimension [7][8]. A notional graph of the first form (weaker in both dimensions) appears in Fig. 1; Simonson [6] gives forms in which Option 3 is equal to or even slightly stronger than Option 2 in Selection Dimension 1. We adopt all three forms, and implemented some of each case, depending on whether we thought the form in which the inferior option was weaker would be believable at all. Graphs for these scenarios were much like Fig. 1, with notional labels, not specific values for axis labels.

### *C. Compromise Effect*

Simonson [6] also defines the *compromise effect* in a similar situation: you again have two alternatives and two dimensions that are important to your selection. These two alternatives (Options 1 and 2 in Fig. 2) create a trade-off between the two dimensions of selection. Again, there is an additional third choice (Option 3 in Fig. 2), but this time it is very near the mean values of the first two options in each dimension of selection. Hence, it can be seen as a good *compromise* between the two *extreme* alternatives. (These terms will be used to refer to choices in the analysis.)

Kivetz et al. [8] note that the compromise may be slightly better, exactly, or slightly worse than the average of the two extremes. (Respectively, Option 3 would be right of, on, or left of the line connecting Options 1 and 2, in Fig. 2.). We adopted the model of the compromise being slightly worse, because we felt it would more readily elicit the effect. In such a constructed scenario, no one should (rationally) choose the compromise without accepting an overall inferior choice. Again, the graph axis labels were relative terms; precise values were never used (as depicted in Fig. 2).

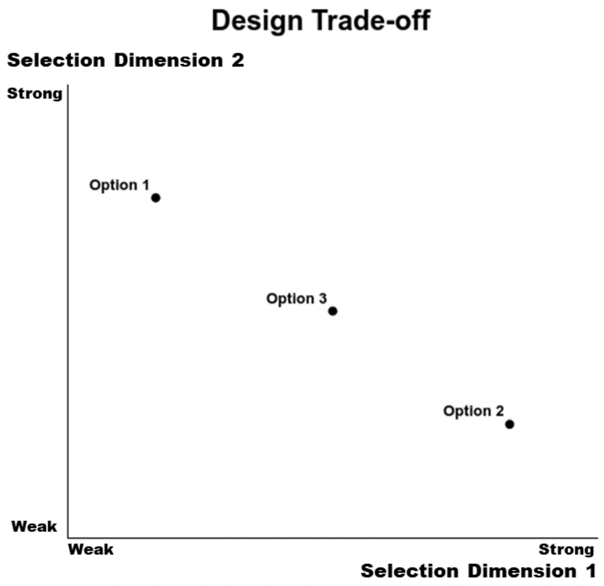


Figure 2. A notional graph showing the *compromise effect* occurring due to the relative placement of options along two dimensions of selection. Options 1 and 2 are seen in some sense as extreme, whereas Option 3 should be seen as good compromise. This leads a consumer to choose Option 3, even if it may not be an exact trade-off (i.e., it would not be on the line connecting Options 1 and 2).

### D. Zero-risk Bias

Baron et al. [9] define *zero-risk bias* as showing a preference for reducing a portion of risk to zero. They do this in the context of how to allocate resources to cleaning up environmentally contaminated sites. They presented three options for cleaning up both sites at varying levels. The authors defined zero-risk bias as expressing that the option that included a reduction to zero risk for one site (out of two) was better. In another version, the zero-risk option actually reduced the cancer cases by fewer incidences; zero-risk bias was defined as choosing this inferior option. This definition of zero-risk bias matches the earlier one of Viscusi et al. [10]. Their choices involved health risk from an existing product versus a new household product (which was presented as real but was only for purposes of the scenario). They found that consumers were willing to accept greater overall "cost" (by whatever metric was defined in the scenario) when the risk in one sub-part of the choice was zero. We followed this model in creating questions we hoped would elicit zero-risk bias.

## III. Data Collection

Under an IRB-approved protocol, we conducted data collection in collaboration with the Naval Aerospace Medical Institute (Pensacola, FL, USA). Various U.S. Navy and Marine personnel volunteered for our study during their free time. Volunteers completed the following steps, per our IRB-approved protocol: (1) informed consent, (2) demographics questionnaire, (3) Cognitive Reflection Test (CRT) [11], (4) Rational-Experiential Inventory (REI) [12], (5) Terrain Orientation Task [13], and then (6--9) four tests designed to elicit the biases described above. The order of these last four sections was determined using a 4x4 Latin square retrieved from an online Latin square generator [14]. This Latin square was counterbalanced for first-order sequence effects [15]. We elected to include the CRT and REI because Sleboda and Sokolowska [16] found each to be predictive of aspects of decision making. We wanted to test whether the Terrain Orientation Task would be predictive of any ability shown on the search-and-destroy tank game, described above.

Our pool of 90 volunteers was skewed to male (69, versus 20 female, with one declining to answer) and college-age or post-college age: 49 were ages 18-22, 34 were ages 23-27, and just seven were age 28 or older. Education level was asked via giving their highest academic degree; 32 said a high school diploma, three said an Associate's degree, 43 said a Bachelor's degree, and 12 said a Master's degree. All were fluent in English (the language of the text portions of the study); most were native speakers, but six identified another language as native, with 2-29 years of speaking English among those, one was native in English and a second language, and five declined to answer. We do not believe language was a barrier, as all participants were members of the U.S. military and thus communicate regularly in English.

### A. Details of the Anchoring Bias Data Collection

Computer-based games have been employed in education and cognitive analysis [17] as an enabler for humans to perform learning or decision-making tasks. Following this, we implemented a game for detecting anchoring bias in a sequential decision-making task. A game player must move a game piece in a grid-based 2D environment. At any point in the game, the player can see only a portion of the game board revealed via a circular viewport centered around the game piece's current location (Fig. 3, top; the red cluster of dots is the game piece). The environment contains objects called tanks that are placed in a cluster around a certain location in the environment. Fig. 3 (bottom) shows the tanks on the game board with the region outside the viewport grayed out for legibility. A tank can be removed or cleared by the player by pressing a key when the game-piece is in the vicinity of the tank. There is also an exit at a fixed location in the environment (elliptical pad on the right edge in Fig. 3, bottom). The exit can be seen only when it is in the player's viewport, but its location is known to the player from the start of the game. The player has two objectives: first, detect and clear all the tanks in the environment, second, after clearing all the tanks, exit the environment.

Due to the limited size of the viewport, a player cannot know beforehand where the tanks are located inside the environment. Consequently, they have to search the environment by moving around the game-piece. Once the tanks are visible inside the viewport, they can move the game-piece to each tank's vicinity, clear the tanks, and finally move to the exit. The game piece could be moved in only the four cardinal directions, Up, Down, Left, or Right. The game board was discretized into a grid-like environment for the purpose of tracking the game-piece's location. Fig. 3 (top) shows a screen capture of the game; Fig. 3 (bottom) shows the full game board (in faded colors) for illustration.

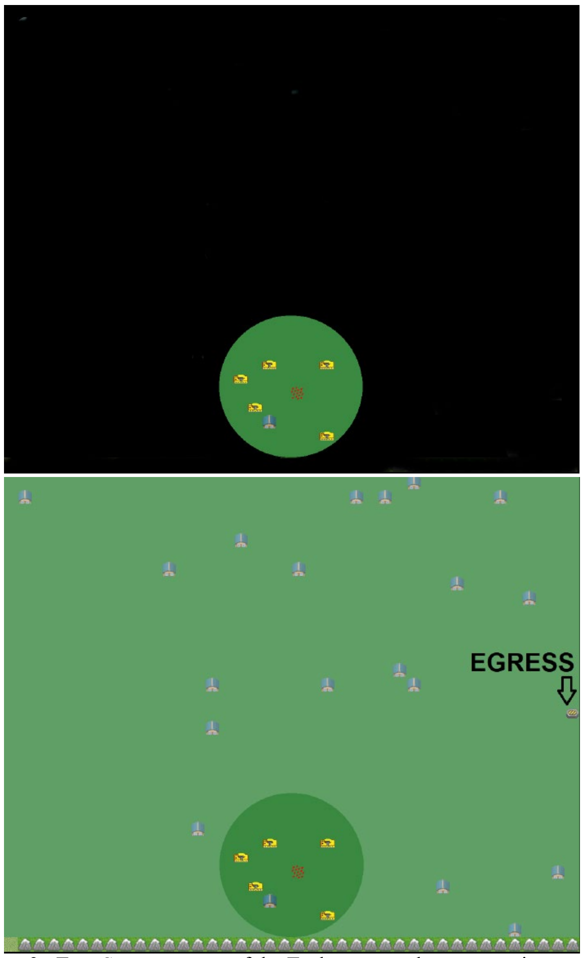


Figure 3. Top: Screen capture of the Tank game as the user saw it, a mostly black field with a viewport centered on the player's current location; the red cluster of dots at the middle of the viewport is the player's game-piece. Bottom: Tank game with grayed map outside viewport (for illustration).

We partitioned the environment into six equal-size cells (three division horizontally and two vertically); the game piece always began in the lower-left cell. The game had two phases. During the anchoring phase, all tanks were placed in a randomly-selected cell other than the lower-left cell. The player then played the game twice. The game would then silently (i.e., unbeknownst to the player) switch to the evaluation phase, where the player's movements would (or would not) indicate anchoring bias.

## B. Details of the Attraction Effect Data Collection

We wrote a mostly custom set of scenarios for our users, beginning with Simonson's consumer scenarios [6] as a basis. We drew inspiration for some scenarios from other studies of the attraction effect [7][18]. We introduced graphs that summarized certain aspects of each scenario; these were shown in the second half of the study for each type of bias. The graphs for the attraction effect clearly showed the closeness of the attractor and the decoy, whereas the target was near the opposite corner of the graph. Since the order of scenarios was determined by a Latin square, all scenarios had a graph available for us to show, and the question counter determined when to show graphs. So, all scenarios were presented with and without graphs across our participants.

## C. Details of the Compromise Effect Data Collection

We again wrote a custom set of scenarios for our users, based largely on Simonson's scenarios [6], but also drawing inspiration from other studies of consumer choice [8][19] and other studies of decision-making [20][21]. As with the attraction effect, we introduced graphs that summarized the compromise being made, showing the not-quite-linear relationship of the three choices, with the one in the middle deviating slightly from this relationship in the direction of the slightly worse according to Kivetz et al.'s [8] description. Again, a Latin square determined the order of scenarios, and the question counter showed graphs in the second half of this portion of the study.

## D. Details of the Zero-risk Bias Data Collection

As with the attraction effect and compromise effect, we wrote custom scenarios for our users, drawing from previous studies of zero-risk bias [9][22][23] or decision-making [20][24]. We again created graphs which used clustered (pairs of) bars to illustrate the choice to be made. Participants needed to sum the length of bars in a cluster in order to compare to the single bar for the zero-risk option. We chose not to use a stacked bar, in part because they are generally more confusing to readers [25], and in part because we felt that the stacking would make the inferiority of the zero-risk option too obvious. (The stacked bars would take the place of having to sum the component results.) As with the attraction and compromise effects, the order of scenarios was determined by Latin square and the graphs shown once the question counter reached the second half of this portion of the study.

## E. Further Details of the Text-based Data Collections

As noted above, each of the tests for the attraction effect, compromise effect, and zero-risk bias were divided in halves. The first half in each showed a text version of the scenario and response options. The second half had this text (scenario and response options) as well as a graph that illustrated what we viewed as the critical data on which participants would want to make their decision for that scenario. At the end of each half, participants were asked if they had any strategy on that section; portions that displayed graphs prompted specifically to indicate whether the participant felt the graph was helpful in that section. On this free text response, we assessed the sentiment towards the graph. The response was categorized as *positive* if the participant indicated using the graph or finding it helpful. The response was categorized as *negative* if the participant indicated ignoring the graph or finding it confusing or otherwise unhelpful. Responses that did not mention the graph at all were categorized as *neutral*. Sentiment will be used as an independent variable in our analyses of these three biases.

We further evaluated these free text responses to determine if the participants indicated a particular strategy to select their response to the scenarios in that portion of the study (nine-question blocks). We identified keywords in the response, and then we coalesced these keywords into five categories; we then added another category for responses that explicitly mentioned keywords from multiple of the other categories. This process yielded the following assessment of strategies (with descriptions of responses that fit into them):

- *None*: Participants expressed clearly that they had "no strategy" or that they "didn't use a strategy"
- *Intuition*: Participants said they "went with their gut reaction" or chose "what felt right"
- *Feature*: Participants said they chose based on a particular feature (that changed with each scenario); examples include least loss of life, lowest value of equipment lost, and lowest cost of devices.
- *Efficient*: Participants said they were aiming for efficiency or the best use of resources
- *Balance*: Participants said they tried to "balance" the competing interests or chose a "middle" option
- *Mixed*: Participants explicitly used multiple words or phrases of the types cited in the preceding descriptions

This yielded an independent variable, Strategy, on which we will report (in Section VI) analyses conducted.

## IV. Data Filtering

Before conducting data analysis, we needed to develop methods to determine which data trials indicated the presence of each bias. We also noted some issues with data that caused us to discard some data trials as unreliable for bias detection. The tank game and the text-based scenarios required separate procedures. These are detailed in this section.

### A. Method for Detection of Anchoring Bias

We partition the environments into six equal-size cells (three horizontal divisions and two vertical divisions). The initial position of the player's game piece was always the lower-left cell. The tanks were placed in a randomly-chosen cell other than the lower-left cell. The game proceeded in two phases. In the anchoring phase, the cell containing the tanks did not change, and the player played five iterations of the game. In the evaluation phase, a new cell was again chosen randomly for the tanks from the four remaining cells (not the initial cell for the player and not the previous cell for the tanks). The player did not know the game switched to the evaluation phase, but simply played two more iterations of the game. For detecting anchoring bias, we check whether, during an evaluation run, the player visited the location where the tanks were during the anchoring runs before exploring other regions of the map. Recall that the map of the game board outside the viewport is not visible to the player while playing the game. So, the only reason for a player to go towards the anchoring location would be due to anchoring bias induced by the location retained in their memory during anchoring runs. If the trajectory during an evaluation phase includes visits into cells that contain the previous position of the tanks, then we considered the player to exhibit anchoring bias. Further details of this are available in a previous paper [1].

From the 74 players that played our game for two game sets each, we collected 148 data instances. Each instance was comprised of five anchoring runs followed by two evaluation runs. These data instances were analyzed for detecting anchoring bias. While analyzing, we found that some of the data instances had to be discarded owing to an oversight in the placement of the anchor. If the location of the tanks during the evaluation run was in-between or en-route from the start location to the location of tanks during the anchoring runs, then it was not possible to determine if the player was anchored or not. We discarded 69 of the 148 data points, leaving 79 valid data points.

### B. Assessing the Responses to Text-based Stimuli

Since three of our question sets were designed to elicit cognitive biases via scenarios presented (primarily or completely) through prose, we applied a filter based on the reading speed implied by the question word count and the response time. Reading speed has been studied for well over 100 years; however, there still seems to be some concerns raised in the literature about the accuracy of the estimates of reading speed. Brysbaert [26] reviewed 190 studies, dating back to 1898. He concluded that for adults reading silently in English (the language of our study), an average reading speed is 238 words per minute (wpm) for non-fiction and 260 wpm for fiction. Noting the existence of "reliable individual differences," he gives ranges of 175-300 wpm (non-fiction) and 200-320 wpm (fiction). He further noted general agreement that college-age young adults have the highest reading speed. (As noted in Section III, this age group was over half of our participant pool.) In addition, we note that our subjects were reading texts that were markedly below their grade level; our texts were rated with the Flesch-Kincaid Grade Level [27] as being eleventh grade level (approximately age 17 in the U.S.) or lower. Intuitively, this could increase the reading speed, although we lack a good estimate for this increase. Furthermore, it is possible that participants did not read all answers choices (perhaps choosing the first or second that they read), making it hard to estimate reading speed for a full question-and-answer set. Finally, our scenarios and answer choices were generally short; one scenario was 126 words, whereas the remaining 53 (across all bias types) were 35-96 words. Answer choices were 8-66 words. This makes reading speed estimates somewhat sensitive to the short length of the passages. In order to be extremely conservative in disqualifying our participants, we assert a maximum reasonable speed of 1000 wpm. Trials above this reading speed were removed from the analysis; we note that a stricter limit of 640 wpm did not substantially change the results. We also removed trials that were "orphaned" by this filtering, in that very few trials from that participant for a certain condition (e.g., graphs present, or scenario type) remained. Keeping such trials would have made the analysis too sensitive to a small sample.

In summary for these three tests, from 3960 data trials; the wpm filter left 3294 trials for analysis. There were 64 participants who completed the attraction effect scenarios, 61 who completed the compromise effect scenarios, and 58 who completed the zero-risk bias scenarios.

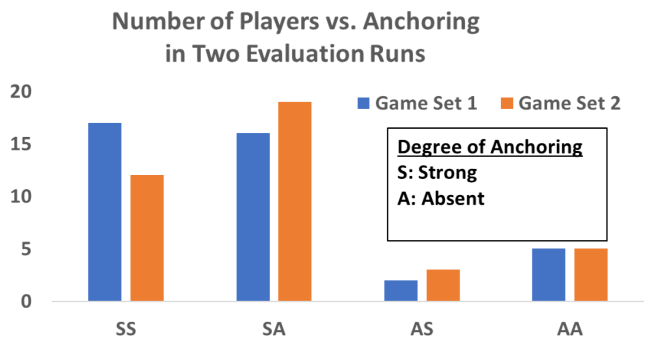


Figure 4. Bar chart showing the number of players (y-axis) that have Strong or no (Absent) anchoring (x-axis) in our two game sets, each consisting of five anchoring runs and two evaluation runs.

## V. Analysis of Anchoring Bias

Because the anchoring bias portion of the study requires a very different method of analysis, we focus this section on analysis of anchoring bias. We had a few research hypotheses and goals, which we present in subsections.

### A. *Exhibiting Anchoring Bias*

*Hypothesis 1*: Participants would exhibit the anchoring bias after five iterations of the tank game.

We detect anchoring bias when the trajectory data from either the first or both evaluation runs meet the criteria above (Section IV.A.). The results show (Fig. 4) evidence of anchoring bias. Out of the 79 data instances, 64 data instances (81%) showed that the player had been anchored (SS and SA in Fig. 4) either in both or only in the first evaluation runs. Across the two game sets, there was very little variation (6%) in the number of subjects displaying anchoring bias. This indicates a strong propensity for anchoring bias among the subjects.

### B. *Duration of Anchoring Bias*

*Hypothesis 2*: When anchoring bias was present, it would last through both evaluation runs.

We determined the number of data instances that showed strong anchoring in the first evaluation run versus those that showed strong anchoring in both evaluation runs (SA versus SS in Fig. 4). We found that in 35 instances players showed that the effect of anchoring waned between the first and second evaluation runs, while the anchoring remained strong between the two evaluation runs for 29 instances. These values indicate that there is small but non-negligible support that the effect of anchoring bias diminishes if the player gets information that contradicts the anchor.

We found that in the first game set, 16 players showed anchoring only in the first evaluation run and 17 showed anchoring in both evaluation runs. In the second game set, these numbers became 19 and 12, respectively. The decrease in strong anchoring in both evaluation runs between the first and second game sets (from 17 to 12), and simultaneous increase in subjects that showed anchoring only in the first evaluation run (from 16 to 19) points further in the direction that, as the player sees more information contradicting the anchor, the effect of anchoring diminishes. Players may have been more fatigued at the start the second set of evaluation runs, after playing 12 runs (five anchoring runs in each of two game sets plus two evaluation runs in first game set) of the game. Conventionally, fatigue would lead to the human brain making shortcuts via heuristics and strengthening the anchoring bias. However, we saw diminishing anchoring bias across game sets. This seems to indicate that the disappointment of not finding the tanks at the anchoring location weakens the anchoring bias and motivates the player to explore in a more objective, less biased manner.

### C. *Building a Model for Prediction of Anchoring Bias*

*Hypothesis 3:* A bias prediction model would enable us to predict exhibition of the bias during evaluation runs from a propensity toward bias exhibited in anchoring runs.

To investigate this hypothesis, we used a bias prediction model based on the work of Jesteadt et al. [28]. We summarize our previous discussion [25] of this model. The model is a linear combination of the stimulus from the current task perception, the stimulus from the task in the previous time-step, and the outcome of the decision in the previous time-step. We mask the current task perception, eliminating one factor. We consider the trajectory length up to viewing the first tank in the viewport as the stimulus from that anchor. This yields a linear model in which each anchor's influence during evaluation is the $J_{eval} = \alpha + \Sigma\ \beta\ J_{anc,i}$, where the summation is over i=1..5 for the five anchoring runs. We used linear regression with least squares [29] to solve this equation. If the slope of the regression line was less than zero, then we say that the participant had propensity toward anchoring bias.

We compare this bias prediction model with the detection. For the first evaluation run (Fig. 5, parts (a) and (c)), the model was generally accurate (true positive plus false negative of 80% and 77%, respectively). Unsurprisingly, the prediction accuracy of the model diminishes considerably to 52% and 37%, respectively, in the two game sets (Fig. 5, parts (b) and (d)). It appears that the exposure to a different location of tanks than the anchoring runs in the first evaluation run reduced the player's reliance on the anchor to search for the tanks during the second evaluation run. We note that none of these results reach the threshold of statistical significance through Fisher's Exact Test. In our best result (Fig. 5(a)), there was statistically no association between the model prediction and the exhibition of bias (p=0.204). We attribute the disconnect between the apparently high percentage of accuracy and the failure to achieve statistical significance to the low sample size; we again lament the need to remove data, as discussed in Section IV.A. We note that with 100 participants (whose data were not invalidated as described in Section IV.A.) and the percentages we observed on the first evaluation run (Fig. 5(a)), Fisher's Exact Test would yield statistical significance.

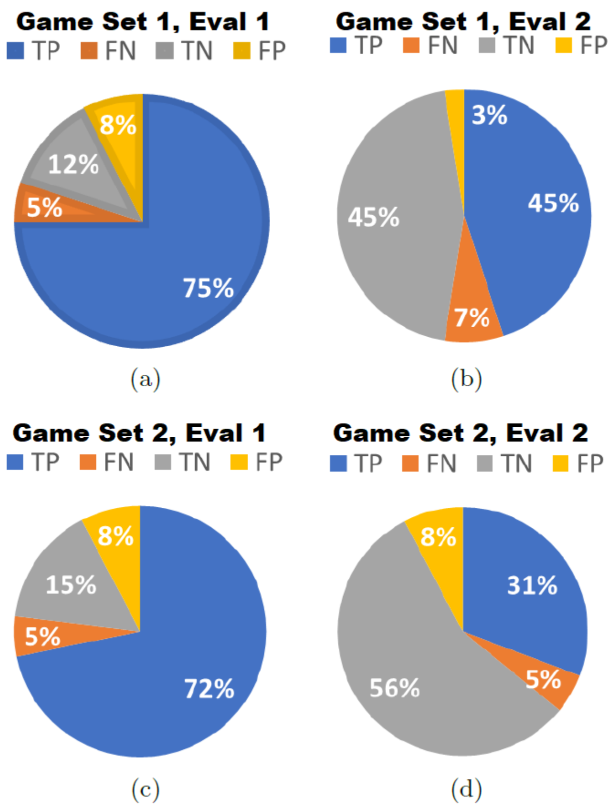


Figure 5. Effect of anchoring bias propensity during anchoring runs on decision in evaluation runs, for game sets 1 (40 trajectories) and 2 (39 trajectories). T/F denote anchoring during anchoring runs as true or false; P/N denote detection of anchoring during evaluation as positive or negative. So, TP and FN are accurate predictions, whereas TP and FN are inaccurate.

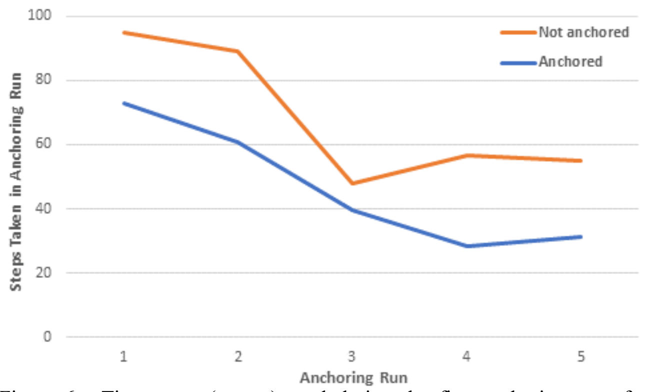


Figure 6. Time steps (moves) used during the five anchoring runs for participants who were later judged to be anchored (blue) and those who were judged to be not anchored (orange). The separation of these two graphs and the significant differences reported lead us to hypothesize that this could be a way to detect anchoring bias in real-time.

Players played the two sets of the game back-to-back without any break. An immediately relevant question is whether the model predicts the anchoring in the second iteration of the game, after having seen the anchor no longer be reliable in the evaluation runs of the first iteration of the game. The answer (Fig. 5(c)) appears to be promising, although this result also does not have statistical support from Fisher's Exact Test; it would appear to require approximately 150 (valid) participants at the percentages indicated in Fig. 5(c) to achieve this threshold. Still, such a result would correspond to findings in other sequential decision-making [5], where the anchoring effect diminished as the decision maker was exposed to more information from successive decision problems that were contrary to the features of the problem in the positive decision instance.

Overall, our findings of the anchoring bias prediction model indicate that a more robust prediction model, based on a larger data set, would be worth investigating for longer-term prediction of anchoring bias effects.

### *D. Potential for Real-time Detection of Anchoring Bias*

We had one other long-term goal for which we could not form a hypothesis. We hoped to identify a method which would indicate in real-time whether a participant appeared to be getting anchored, rather than relying on a post-hoc assessment of whether the participant was anchored. It appears that the number of time steps (moves) used in our game is a potential real-time indicator of whether a participant is getting anchored. We observed a significant correlation between the count of anchoring runs and the number of time steps (moves) taken in the game until the tanks were within the viewport. This was true for both those judged post-hoc to have been anchored – Pearson R=-0.98, t(3)=-7.99, $p < 0.005$ – and those judged to be not anchored – Pearson R=-0.88, t(3)=-3.29, $p<0.047$. But these two observations were different. Analysis of variance (ANOVA) showed that, for the first iteration of the game, those who were not anchored were significantly slower than those who were – F(1,38)=5.793, $p<0.022$, generalized effect size η=0.132. This difference disappeared for the second iteration of the game – F(1,37)=0.393, $p>0.534$. Looking at the graph of the moves taken (Fig. 6), we can see the separation, which leads us to hypothesize that this may be a way to detect anchoring bias in real time. Further data would have to be gathered to validate this hypothesis.

## VI. Analysis of Biases through Text Scenarios

The attraction effect, compromise effect, and zero-risk bias were hoped to be elicited through text-based scenarios. The analysis of these three biases follows a similar pattern, and we present these analyses in the following subsections.

As a preliminary result, we noted a strong correlation – Pearson R=0.85, t(16)=6.336, $p<0.001$ – between the trial number (1-18) and reading speed (Fig. 7). This correlation measured reading speed under the assumption that participants read the scenario and all three answer choices completely. This measure was averaged across all three decision-making tasks, which "folds over" the trial number three times, if participants completed all three question sets. Some of this could be attributed to using the graphs as a shortcut, which some participants indicated that they were doing. This could also indicate that participants were getting fatigued, since they "read" faster and faster as they progressed through the trials. Note that the last three trials (and at least four of the last five) are faster than the speed that the literature on reading comprehension indicates is likely for a reader to be able to read for comprehension (see

TABLE I: DISTRIBUTION OF RESPONSES TO THREE SCENARIO TYPES

| Attraction Effect | | Compromise Effect | | Zero-risk Bias | |
|---|---|---|---|---|---|
| Attractor | 401 | Compromise | 384 | Zero-risk | 365 |
| Decoy | 373 | Extreme | 714 | Balanced risk | 679 |
| Target | 378 | | | | |

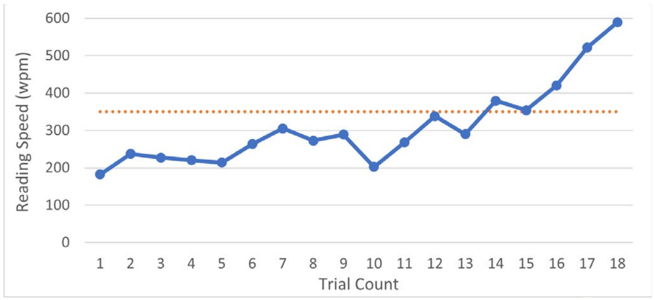


Figure 7. Measured reading speed versus trial count. We saw a strong correlation (Pearson R=0.846, t(16)=6.336, p<0.001) between trial number and reading speed. Trials 10 through 18 showed graphs, which may have influenced "reading speed" by enabling a shortcut to reading prose for the information needed to make a decision.

Section IV.B.). Perhaps participants were not reading all the answer choices, or they were just scanning. Some of this may be attributed to most of the scenarios being at a reading grade level well below the participants' respective abilities (based on the self-reported education levels, reported in Section III).

Overall, we hypothesized a skewed distribution of the responses to the three types of scenarios (for the three biases discussed below). We believed this would indicate the presence of the bias. As shown in Table I, we got a nearly balanced set of responses to each of the three question types. Thus, we cannot conclude that the bias was elicited by our scenarios. Therefore, our analysis focuses on whether there were certain features of the scenarios or of the participants that may be associated with eliciting the bias in a subset of the data. We note that not all participants completed all sections of the study, so the degrees of freedom in the ANOVA measures below are not the same as in the test above for the anchoring bias, nor do they match each other.

### A. *Detection of Attraction Effect*

Following [16], we measured the correlation between the REI scores (both the rational and experiential, as well as the ability and engagement sub-scales within each score) and the selection of the unpreferred options. We did not find a significant correlation between any REI score (the two main scales or any of the four sub-scales) and the rate of selecting unpreferred responses. We also measured the CRT score for each participant and measured the correlation of these scores against the rate of selection of the unpreferred options. Again, we found no significant results for the attraction effect. We had hoped that we would not only elicit the biases described above, but that we could help mitigate these by the presence of the graphs. However, there was no main effect of the presence of the graphs in the second half of the set of questions for the attraction effect.

We wrote scenarios of multiple types that reflected many roles members of the military might encounter in their duties. This encompasses mundane issues like purchasing non-military equipment, purchasing military equipment, and even more weighty matters such as aspects of military strategy. We noted a main effect – $F(3,189)=9.465$, $p<0.001$, $\eta=0.030$ – of this cost measure on the response time. However, we note that the generalized effect size $\eta$ was very small. The questions were ordered according to a Latin square using the question ID; this led to some variation in the relative placement of the type of cost measure that created a potential confound of this effect. In addition, some of this effect may reflect the general behavior of participants to get faster as the question count rose. Finally, we categorized scenarios into multiple types, inflating the degrees of freedom and potentially the effect. Therefore, we note this effect, but do not yet consider it to be a reliable result.

We had hoped to elicit the biases, and also mitigate these by the presence of the graphs illustrating the data. Unfortunately, there was no main effect of the presence of the graphs in the second half of the study on the rate of selecting the attractor or decoy – $F(1,63)=0.689$, $p=0.409$. We also did not observe a main effect of the presence of the graphs on response time – $F(1,63)=0.977$, $p=0.326$. Foreshadowing the contrast with the compromise effect and zero-risk bias (which both show this main effect), we note that the attraction effect scenarios tended to have a lower word count, which could have confounded the "shortcut" of using the graph and not reading the answer choices. There was a main effect of graph sentiment on the rate of selecting the various responses to the attraction effect – $F(2,61)=3.423$, $p<0.039$, $\eta=0.101$. Participants whose sentiment about the graphs appeared to us to be negative selected unpreferred options on the attraction effect (attractor or decoy) 61.1% of the time. Participants whose sentiment appeared neutral selected unpreferred options 66.7% of the time. Participants whose sentiment appeared positive selected unpreferred options 73.5% of the time. Follow-up t-tests indicate that all differences between these values are statistically significant. (For the smallest difference, negative to neutral, $t(47)=3.236$, $p<0.003$.) It appears that those participants with positive sentiment toward the graphs were led to exhibit the bias with *greater* frequency than those who did not use (or actively avoided) the graphs. In retrospect, perhaps eye tracking would have been advisable, so that we could know exactly what portions of the graph those who said they used it or liked were reading in order to make their decision. That might have given us greater insight to this result.

The Strategy we inferred based on the free text responses had a significant main effect on their rate of selecting the unpreferred option on the attraction effect portion of the study – $F(5,84)=3.228$, $p<0.011$, $\eta=0.161$ (Fig. 8). Using post-hoc t-tests, we determined that there were essentially two groups of three strategies. The strategies (see Section III.E.) *None*, *Intuition*, and *Mixed* were not significantly different from each other, and the remaining strategies of *Balance*, *Efficient*, and *Feature* were not different from each other (though some differences with the second set showed a

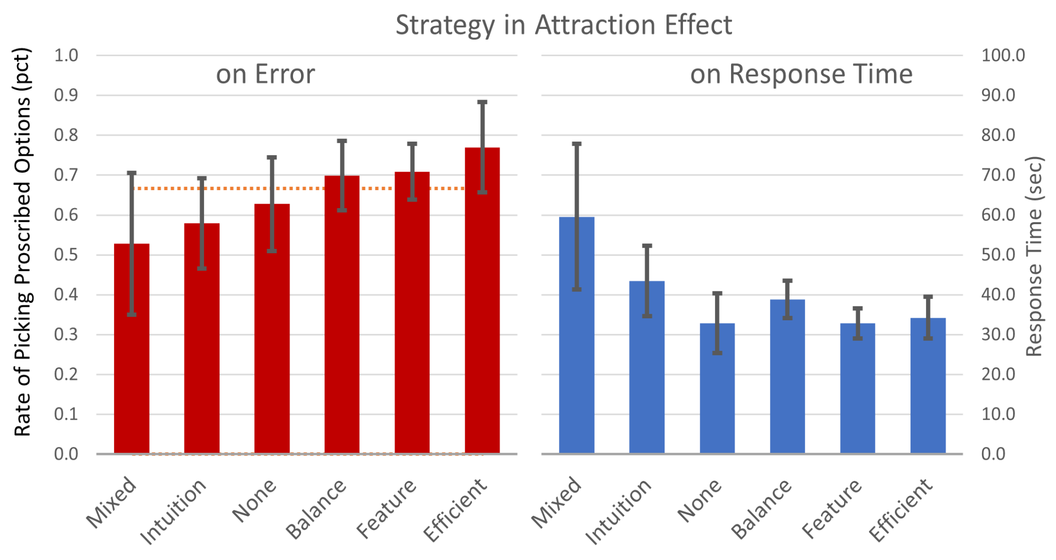


Figure 8. This pair of graphs shows the effects of strategy in the attraction effect portion of the study. The strategy was assessed by the research team based on the free text responses of participants. The assessed values showed a main effect of strategy on the rate of selecting the unpreferred option (left, red bars). The orange line represents random selection; as noted in the text, the three strategies with the least error were statistically different from the three with the most error. We also noted a trend towards difference in response time based on strategy for the attraction effect scenarios (right, blue bars), largely due to the slow responses using the *Mixed* strategy.

trend in the t-test). But all strategies in the first set led to significantly different performance than strategies in the second set. The first column of Table II shows that the usage of these strategies was not equally distributed amongst our participants on the attraction effect scenarios.

There was also a trend for the inferred strategy to lead to differences in response time for the attraction effect scenarios – F(5,84)=1.990, p<0.089, η=0.106. However, the grouping is quite different. The *Mixed* strategy was notably slower than all the others, with the *Intuition* strategy being slightly slower than the remaining four. While the graphs do not mirror each other, we do note that the *Mixed* strategy had the lowest rate of selecting the unpreferred option, took the longest time, and was the least-often used. We did not find an interaction between graph sentiment and inferred strategy.

There was statistically no association between anchoring bias and the attraction effect. There was no association between the attraction effect and the compromise effect, nor between the attraction effect and the zero-risk bias (p>0.409 for all associations).

## B. Detection of Compromise Effect

As with the attraction effect, we measured the correlation between REI scores and CRT scores. Again, we did not find a significant correlation between REI score (or any subscale) and choosing the compromise option. Similarly, we did not find a correlation with CRT scores and selection of the compromise option. We had hoped that we would not only elicit the biases described above, but that we could help mitigate these by the presence of the graphs. However, there was no main effect of the presence of the graphs in the second half of the set of questions for the compromise effect.

Regarding the type of scenario, we once again observed a main effect of the type of cost measure on response time – F(1,60)=27.970, p<0.001, η=0.032. As with the attraction

TABLE II. FREQUENCY DISTRIBUTION OF STRATEGIES IN SCENARIOS

| **Strategy** | **Attraction Effect** | **Compromise Effect** | **Zero-risk Bias** |
|---|---|---|---|
| None | 17 | 13 | 12 |
| Intuition | 19 | 14 | 12 |
| Feature | 42 | 33 | 51 |
| Efficient | 14 | 8 | 6 |
| Balance | 28 | 41 | 29 |
| Mixed | 8 | 13 | 6 |

effect, there was uncontrolled variation in the ordering (we did not counterbalance the order of questions by type), and participants generally got faster, which may confound this effect. With the compromise effect, we had only two scenario types, but even without this potential inflation, we consider it a weak (small η) and, for now, unreliable result.

Again, as we did with the attraction effect, we had hoped that the presence of the graphs would reduce the rate of selecting the compromise option. Unfortunately, we did not observe a main effect – F(1,60)=2.000, p=0.162. However, we did observe a main effect of the presence of the graphs on response time – F(1,60)=23.943, p<0.001, η=0.029. We see a statistically significant but small effect, and like the previous main effect of scenario type on response time, because the graphs were always the second half of each portion of the study, this effect could be due to faster ingest of information through the graph, or may be confounded with the general pattern of participants getting faster with the increasing number of data trials to complete.

There was no main effect of graph sentiment on the distribution of choices among the options – F(2,58)=1.081, p>0.453. Neither was there a main effect on the response time – F(2,58)=1.500, p>0.231. There was no main effect of strategy on the rate of selecting the compromise option – F(6,94)=1.501, p>0.186 – or on the response time – F(6,94)=0.778, p>0.589. However, examining the distribution of the inferred use of each strategy (middle column of Table II) and the definition of the *Balance* strategy (Section III.E.), we see that the most commonly named strategy was choosing a "balance" between the features or the option that was "in the middle." This explicitly expresses a bias for the compromise option, and is the strongest evidence we have that at least some participants made their choice on the basis of this bias. However, this strategy was used on approximately one-third of the data trials (33.6%), so while it indicates that some users chose on the basis of this bias, it does not constitute a majority strategy, and it does not appear to have led to significantly better performance. We did not find an interaction between graph sentiment and inferred strategy.

There was no statistical association between anchoring bias and the compromise effect. Nor was there an association between the compromise effect and zero-risk bias (p>0.748 for both associations). We observed a trend toward statistical association between the inferred strategy of *Balance* (defined in Section III.E. as participants saying they tried to balance the competing interests or features, or that they chose the middle option), p<0.058. So, it appears quite possible that participants were accurate in assessing their own strategy, at least with the Balance strategy.

### C. *Detection of Zero-risk Bias*

As with the two previous biases, we measured the correlation between the REI scores and the rate of zero-risk bias, as well as the CRT scores and the rate of the zero-risk bias. In contrast to the previous results, we did see a significant correlation between the CRT score and the rate of selecting the zero-risk option, t(56)=2.5425, p=0.014. This would appear to extend the prior results, since zero-risk bias was not a part of the previous work [16]. We get a similar result evaluating this association with Fisher's Exact Test, yielding an association with p<0.022. A significant portion of the incorrect responses we saw to the CRT were the intuitive-but-incorrect responses [11]. Therefore, it should not be a surprise that this effect also manifests itself in a significant (negative) correlation between the number of the intuitive (but incorrect) responses our participants gave and their rate of selecting the zero-risk option, t(56)=2.954, p<0.005. The zero-risk option can be seen as an intuitively best choice, even though deeper analysis shows it is inferior.

Again, in parallel to the previous two bias types, we noted a main effect of scenario type on response time – F(4,228)=7.665, p<0.001, η=0.041. All the caveats described for the attraction and compromise effects apply to the zero-risk bias portion of the study (uncontrolled variation in ordering of scenario type, general behavior of participants to get faster, large degrees of freedom due to multiple type categories). Again, the effect is not yet considered reliable.

Turning to the presence of the graphs, we did not observe a main effect on the rate of selecting the zero-risk option – F(1,57)=0.040, p=0.841. But we observed a main effect on the response time – F(1,57)=9.325, p<0.021, η=0.021. As with the compromise effect, this main effect is potentially confounded with the general tendency for participants to get faster as the number of data trials increased (since the graphs were always the second half of this section of the study).

We found that graph sentiment had a main effect on the response time in the zero-risk portion of the study – F(2,55)=5.557, p<0.007, η=0.168. Participants with positive sentiment were fastest (58.3 sec), followed by participants who expressed no sentiment about the graphs (66.1 sec). Participants who said they found the graph unhelpful or said they did not use it were slowest, at 94.2 sec. All these pairwise differences are significant with p<0.001. As with the result for the attraction effect, we lament the missed opportunity to have recorded eye movements and gain greater insight into this result. There was no main effect on the selection of responses to the scenarios intended to elicit the zero-risk bias – F(2,55)=1.476, p>0.710.

There was a trend for these strategies to lead to different rates of selecting the unpreferred option on the zero-risk bias portion of the study – F(5,76)=2.188, p<0.065, η=0.126. There was a trend for the various strategies to lead to different response times. Since we chose (as described above) to make the zero-risk option objectively worse (i.e., a greater total loss), the *Feature* strategy should have helped identify this aspect of the scenario's data and thus have been successful at avoiding the unpreferred option. But participants had to look for the minimal loss summed over the two portions of the outcome. Using post-hoc t-tests, we found that this strategy had a statistically lower rate than only the *Balance* strategy. There was a trend for the *Feature* strategy to be better than the *Efficient* strategy. There was main effect of strategy on the response time – F(5,76)=1.77, p>0.128. We did not find an interaction between graph sentiment and inferred strategy.

There was statistically no association between anchoring bias and zero-risk bias (p>0.745). Nor was there association between inferred strategy and the zero-risk bias (p>0.406).

## VII. Discussion

We investigated using game-playing and decision-making exercises to induce cognitive biases, with limited success. We were able to induce anchoring bias. However, for a small fraction of the players (1 out of 74 instances in set 1 and 3 out of 74 instances in set 2), we found that they initially showed influence of the anchor during the first few anchoring runs, but in subsequent anchoring runs and in the evaluation run, the anchoring effect went away. They started exploring the map instead of heading to the anchor location. Fig. 9 shows an example where the first two anchoring runs (top) show anchoring but the other anchoring runs (bottom) do not. This de-anchoring was more pronounced in set 2. Perhaps the evaluation runs in game set 1 reduced the reliance of the player on the anchor during game set 2 even after they found it, prompting general exploration again.

The movement of game-piece in our computer-based game for anchoring bias was controlled by keyboard arrow keys; thus, it was limited to the four cardinal directions. This resulted in players using long horizontal or vertical tracks to

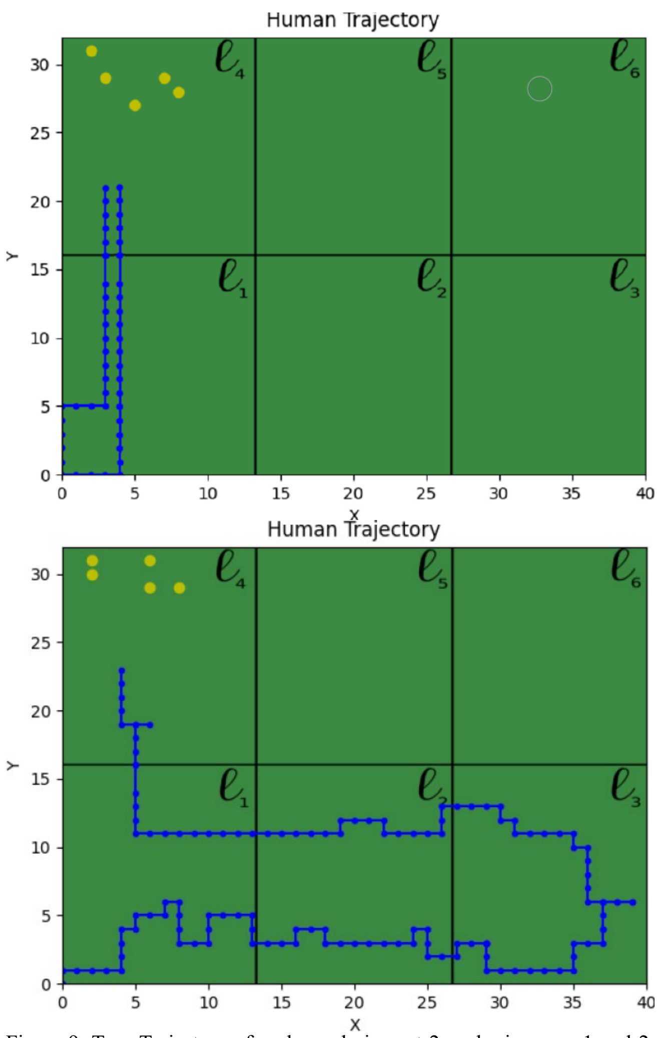


Figure 9. Top: Trajectory of a player during set 2 anchoring runs 1 and 2. Bottom: Trajectory during anchoring runs 3-5 (bottom). These images also illustrate the six cells of the game board to detect anchoring bias.

explore the environment. The number of key presses made by players in the game was not recorded and there is a possibility that some players purposefully reduced the number of keystrokes by holding keys to continue in the same direction for long periods. This could also have stemmed from psychological factors like motivation, interest, and engagement with the game and overall experiment.

Anchoring bias, as we have used the term, intersects with other biases. Sequential bias deals with repetitive decision outcomes in sequential (not necessarily time-extended) tasks. Experiential bias considers the reliance of humans on experience from past decision outcomes on the current decision. It would be interesting to analyze our results with appropriate theoretical models for these biases, to understand overlap, similarity, and divergence between these biases.

What causes humans to depend on anchors for making decisions? The conventionally accepted theory is the human brain is inclined to make shortcuts via heuristics [2] due to boredom, motivation, repetitiveness and other factors. In contrast, the selective accessibility model [30] proposed an alternative theory that the brain made information related to the anchor more readily accessible to its decision process. The difference is subtle but consequential, as the former attributes the cause of anchoring bias to the internal working of the brain's decision-making process while the latter attributes it to the information presented to the brain's decision-making process. A deeper understanding, fortified with appropriate mathematical models for these two theories, would help with a clearer understanding of anchoring bias.

## VIII. Conclusion and Future Work

We elicited anchoring bias with a time-extended task. We were statistically unable to elicit the attraction effect, compromise effect, or zero-risk bias. We have some evidence that the zero-risk bias and compromise and attraction effects exist due to selection of options that should not have been selected at all and due to statements from the participants in which they indicate the compromise effect. However, we had limited success in identifying factors that contribute to the bias. The stated strategy and the use of graphs seems to have influenced the attraction effect, and the stated strategy also influenced the zero-risk bias. Further research would be needed to determine how one might systematically elicit (and therefore, how to systematically mitigate) these biases. As noted above, we recommend that future research on the potential for graphs to mitigate bias should incorporate eye tracking to determine the extent of participants' use of graphs and which options they considered the longest. We recommend counterbalancing the usage of graphs to investigate the reliability of our results on the response time in the presence of graphs for both the compromise effect and zero-risk bias.

One could explore links between learning style, problem-solving strategies, and these biases. Perhaps the biggest difference between the text-based scenarios (with or without graphs) is the level of engagement participants had with the exercises. Our initial design concept included interleaving the tank game with responding to decision-making scenarios at stopping points in the game. This design felt contrived; thus, we worried that our data would suffer. Clearly, there is room for improvement on our design. We repeat the lament of our tank game design and the placement of anchors that caused us to discard many data trials because the placement of the tanks rendered it impossible to determine if the path was due to anchoring or not.

We note one potential avenue to identify bias in real-time. The anchoring effect appears potentially predicted by the time spent in the anchoring runs. The stated strategy of the participant matched the exhibition of the compromise effect. In what appears to be a new result in the literature, we found a statistical effect of the responses to the CRT and the exhibition of the zero-risk bias. We believe these observations hold promise for future research. Study designs, with larger participant pools, that explicitly investigate these relationships could potentially validate our results and would represent logical next steps for the detection and prediction of each of these biases.

## Acknowledgment

The authors thank Zhuming Ai, Bryan Brandt, Joseph Coyne, Kayla Russell, and Kaylin Strong. The authors also thank the editors and reviewers for their many contributions to the improvement of this manuscript. The Office of Naval Research funded our work via the NRL Base Program.

## References

[1] P. Dasgupta, J. Kliem, M.A. Livingston, J. Decker, “Inducing and Detecting Anchoring Bias via Game-Play in Time-extended Decision-Making Tasks,” IARIA Annual Congress on Frontiers in Science, Technology, Services, and Applications, July 2025

[2] E. Dimara, S. Franconeri, C. Plaisant, A. Bezerianos, P. Dragicevic, “A Task-Based Taxonomy of Cognitive Biases for Information Visualization,” IEEE Transactions on Visualization and Computer Graphics, vol. 26, no. 2, pp. 1413-1432, 2020

[3] A. Tversky and D. Kahneman, “Judgment under Uncertainty: Heuristics and Biases,” Science, vol. 185, no. 4157, pp. 1124-1131, 1974

[4] P.T. Sukumar, R. Metoyer, S. He, “Making a Pecan Pie: Understanding and Supporting The Holistic Review Process in Admissions,” Proceedings of the ACM Conference on Human-Computer Interaction, Volume 2 CSCW, Article no. 169, 2018

[5] J.M. Echterhoff, M. Yarmand, J. McAuley, “AI-Moderated Decision-Making: Capturing and Balancing Anchoring Bias in Sequential Decision Tasks,” Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems, Article no. 161, 2022

[6] I. Simonson, “Choice Based on Reasons: The Case of Attraction and Compromise Effects,” J. of Consumer Research, vol. 16, no. 2, pp. 158-174, 1989

[7] J. Huber, J.W. Payne, C. Puto, “Adding Asymmetrically Dominated Alternatives: Violations of Regularity and the Similarity Hypothesis,” J. of Consumer Research, vol. 9, no. 1, pp. 90-98, 1982

[8] R. Kivetz, O. Netzer, V. Srinivasan, “Extending Compromise Effect Models to Complex Buying Situations and Other Context Effects,” J. of Marketing Research, vol. XLI, pp. 262-268, Aug 1989

[9] J. Baron, R. Gowda, H. Kunreuther, “Attitudes Toward Managing Hazardous Waste: What Should Be Cleaned Up and Who Should Pay for It?,” Risk Analysis, vol. 13, no. 2, pp. 183-192, 1993

[10] W.K. Viscusi, W.A. Magat, J, Huber, “An Investigation of the Rationality of Consumer Valuations of Multiple Health Risks,” RAND Journal of Economics, vol. 18, no. 4, pp. 465-479, Winter 1987

[11] S. Frederick, “Cognitive Reflection and Decision Making,” J. of Economic Perspective, vol. 19, no. 4, pp. 25-42, Fall 2005.

[12] R. Pacini and S. Epstein, “The Relation of Rational and Experiential Information Processing Styles to Personality, Basic Beliefs, and the Ratio-Bias Phenomenon,” J. of Personality and Social Psychology, vol. 76, no. 6, pp. 972-987, 1999

[13] J.T. Coyne et al., “Evaluation of New Measures of Spatial Ability and Attention Control for Selection of Naval Flight Students,” Human Factors in Design, Engineering, and Computing (Proceedings of AHFE Intl.), vol. 159, pp. 2006-2016, 2024

[14] D. Masson, “Balanced Latin Square Generator,” Online at https://damienmasson.com/tools/latin_square, Last accessed 11 Oct. 2025

[15] J.V. Bradley, “Complete Counterbalancing of Immediate Sequential Effects in a Latin Square Design,” J. of the American Statistical Association, vol. 53, no. 282, pp. 525-528, 1958

[16] P. Sleboda and J. Sokolowska, “Measurements of Rationality: Individual Differences in Information Processing, the Transitivity of Preferences and Decision Strategies,” Frontiers in Psychology, vol. 8, pp. 1844, 2017

[17] M.W.B. Zhang, J.B. Ying, G. Song, R.C.M. Ho, “A Review of Gamification Approaches in Commercial Cognitive Bias Modification Gaming Applications,” Technology and Health Care, vol. 26, no. 6, pp. 933-944, 2018

[18] S.A. Malkoc, W. Hedgcock, S. Hoeffler, “Between a rock and a hard place: The failure of the attraction effect among unattractive alternatives,” J. of Consumer Psychology, vol. 23, no. 3, pp. 317-329, 2013

[19] R. Dhar, S.M. Nowlis, S.J. Sherman, “Trying Hard or Hardly Trying: An Analysis of Context Effects in Choice,” J. of Consumer Psychology, vol. 9, no. 4, pp. 189-200, 2000

[20] A. Mintz, N. Geva, S.B. Redd, A. Carnes, “The Effect of Dynamiac and Static Choice Sets on Political Decision Making: An Analysis Using the Decision Board Platform,” American Political Science Review, vol. 91, no. 3, pp. 553-566, Sep. 1997

[21] L.K. Marusich, N. Buchler, J.Z. Bakdash, “Human Limits to Cognitive Information Fusion in a Military Decision-Making Task,” 19th Intl. Command and Control Research and Technology Symposium, June 2014

[22] E. Schneider, B. Streicher, E. Lermer, R. Sachs, D. Frey, “Measuring the Zero-Risk Bias: Methodological Artefact or Decision-Making Strategy?” Zeitschrift für Psychologie (J. of Psychology, Germany), vol. 225, no. 1, pp. 31-44, 2017

[23] D.J. Mezzio, V.B. Nguyen, A. Kiselica, K. O’Day, “Evaluating the Presence of Cognitive Biases in Health Care Decision Making: A Survey of U.S. Formulary Decision Makers,” J, of Managed Car & Specialty Pharmacy, vol. 24, no. 11, pp. 1173-1183, Nov. 2018

[24] L. Dodd and J. Moffat and J. Smith, “Discontinuity in decision-making when objectives conflict: a military command decision case study,” J. of the Operational Research Society, vol. 57, no. 6, pp. 643-654, 2006

[25] C. Nobre, K. Zhu, E. Mörth, H. Pfister, J. Beyer, “Reading Between the Pixels: Investigating the Barriers to Visualization Literacy,” Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems, May 2024

[26] M. Brysbaert, “"How Many Words do we Read per Minute? A Review and Meta-analysis of Reading Rate,” J. of Memory and Language, vol. 109, Dec. 2019

[27] J.P. Kincaid, R.P. Fishburne Jr., R.L. Rogers, B.S. Chisson, “Derivation of New Readability Formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel,” Research Branch Report 8-75, Naval Air Station Memphis, Feb. 1975

[28] W. Jesteadt, R.D. Luce, D.M. Green, “Sequential Effects in Judgments of Loudness,” J. of Experimental Psychology: Human Perception and Performance, vol. 3, no. 1, pp. 92-104, 1977

[29] Y. Nievergelt, “Total Least Squares: State-of-the-Art Regression in Numerical Analysis,” SIAM Review, vol. 36, no. 2, pp. 258-264, 1994

[30] T. Mussweiler and F. Strack, “Comparing is Believing: A Selective Accessibility Model of Judgmental Anchoring,” European Review of Social Psychology, vol. 10, no. 1, pp. 135-167, 1999

# VR-PM: Immersive Process Mining and Process Modeling with BPMN and Petri Nets in Virtual Reality

Roy Oberhauser[0000-0002-7606-8226]
Computer Science Dept.
Aalen University
Aalen, Germany
e-mail: roy.oberhauser@hs-aalen.de

*Abstract* - **Automation of business and industrial processes are increasing essential for and affect a larger set of stakeholders. Process Mining (PM) relies on execution logs to provide process-centric analysis data, which can support process stakeholders with evidence-based insights and decision support. Yet PM has typically been the purview of specialists, while analysis, insights, and models have not been readily accessible to a larger set of affected stakeholders. This paper contributes VR-PM, an immersive solution concept towards visualizing and interacting with process models (BPMN and Petri Nets), process variants, process conformance checking, and PM data in Virtual Reality (VR). Our realization shows its feasibility, and a case-based evaluation provides insights into its capabilities.**

***Keywords - process mining; process analysis; virtual reality; business process management; process conformance checking; process discovery; Petri Nets; BPMN; process modeling.***

## I. Introduction

This article extends our paper on VR-ProcessMine [1], adding immersive Petri Net models, Business Process Model and Notation (BPMN) models, process discovery, an alternative process variant visualization, and conformance checking capabilities in VR.

The digital transformation sweeping through society affects businesses and organizations, resulting in an increased emphasis on business agility and automation that affects a larger set of stakeholders. Business Processes (BPs) (or workflows) are one significant automation area, as evidenced by the Business Process Management (BPM) market, which is forecast to grow from \$20B in 2024 to \$60B by 2030 [2]. Each execution of such a process leaves a digital footprint of process-related events and the timepoint of execution, typically contained in various log files across the various IT systems (business, manufacturing, etc.) involved in an enterprise. BPs are a way for ordering the activities involved in and executed in an enterprise, be they automated, semi-automated, or human-driven, and thus BPM is where much of the value generated by an enterprise is achieved.

Repeatable processes are fundamental for describing and understanding how businesses and organizations operate, and are utilized in IT, manufacturing, Industry 4.0, etc. Increasing digitalization and automation necessitate evidence-based comprehension and analysis of the processes involved, including their variations, anomalies, performance, and evolution. Process Mining (PM) [3] is a sub-field of data science specifically focused on analyzing event data generated when (business) processes are executed [4]. Because PM relies on event logs of actual process executions, it is evidence-based (or fact-based). This analysis can provide essential insights for understanding and optimizing (business) process execution. When referring to processes we assume BPs to be a subset of the more abstract term and will use both terms interchangeably. One process variant represents a set of process instances that resulted in the same sequence of events.

The Process Mining Manifesto [5] describes six guiding principles and eleven challenges. Our contribution intends, in particular, to support these principles:

- *GP1: Event Data Should Be Treated as First-Class Citizens*,
- *GP4: Events Should Be Related to Model Elements*, and
- *GP6: Process Mining Should Be a Continuous Process*.

Furthermore, the contribution seeks to address these challenges in particular:

- *C10: Improving Usability for Non-experts*, and
- *C11: Improving Understandability for Non-experts*, are a primary motivation for our work.
- A secondary effect that is enabled is to support *C9: Combining Process Mining with other Types of Analysis*.

In general, visualization remains a challenge when dealing with large data sets that involve relations and different variation sets. As data and processes become more relevant to the digital enterprise and stakeholders more digitally savvy, it is all the more relevant and challenging to integrate non-expert enterprise stakeholders in process analysis. By leveraging the immersive capabilities of Virtual Reality (VR), BP analysis can be made more accessible to a wider set of stakeholders, such that not just process modeling specialists, but also those directly involved in executing a BP or observing an automated BP can view und gain insights to various issues regarding a BP of interest, including the combination with other relevant enterprise models.

In prior work, we developed VR-BPMN [6] to visualize business processes in VR based on the BPMN notation. Our VR-EA [7] contributed a VR solution for visualizing, navigating, annotating, and interacting with ArchiMate Enterprise Architecture (EA) models. VR-EAT [8] presented our VR-based solution for visualizing dynamically-generated EA diagrams from EA tools.

This paper contributes VR-PM, an immersive solution concept towards visualizing and interacting with process models as BPMN [9] and Petri Nets [10][11], process variants, process conformance checking, and PM data in VR. Our prototype realization shows its feasibility, and a case-based evaluation provides insights into its capabilities for addressing the aforementioned challenges. Relative to our initial VR-ProcessMine [1], it offers enhanced capabilities by integrating the visualization of process event logs, Petri Nets, BPMN models, process conformance checking, an alternative process variant visualization, and process discovery, which can generate either a Petri Net or a BPMN model.

This paper is structured as follows: Section II discusses related work. In Section III, the solution is described. Section IV provides details about the realization. The evaluation is described in Section V and is followed by a conclusion.

## II. Related Work

The systematic literature review by Milani et al. [12] develops a PM practitioner framework for capturing business questions that PM use cases can answer. While VR and visualization are not mentioned, model discovery (with BPMN and Petri Nets cited as examples) and conformance checking are. Vom Brocke et al. [13] propose a PM research framework that also defines process participants, process stakeholders, and external partners, who are affected by and have an interest in process analyses that span technical, individual, group, organizational, and ecosystem levels. Indeed, our VR-PM solution can support the various roles involved in processes and PM, including behavioral visibility, discovery, variant analysis, and conformance, and enhanced integration with increasing datafication to comprehend and utilize the available information.

As to work regarding PM visualization, the systematic literature review on visualization in PM by Eckhard et al. [14] cites no work involving VR nor any BPMN visualization in conjunction with PM. It explicitly mentions that BPMN and Petri Net PM visualization are under-represented, and conformance-checking PM visualization is under-researched. They conclude that Directly Follows Graph (DFG) is the most commonly used PM visualization. 3DCR [15] is a 3D immersive tool that uses an industrial declarative process modeling language Dynamic Condition Response Graphs (DCR graphs), offering activity-based 3D process animations in the game view using domain-specific representations (sickness registration process). Work involving PM with VR include Vogel and Thomas [16], which shows groundwork and an architecture concept, yet no prototype is described nor are VR screenshots provided. Corea and Delfmann [17] describe a 3D tool for geospatial visualization of paths based on a DFG generated from the PM tool PM4Py. Other work combining PM with VR is often specialized to processes in certain sectors, such as training for factory or manufacturing, logistics, safety, or education and learning, or the health sector. For instance, Roldán et al. [18] describe a complex assembly training system for Industry 4.0 operators.

Open source PM tools include the ProM Framework [19], Apromore [20], and PM4Py [21]; commercial options include products from over 35 vendors, including Celonis, Disco, UiPath, ARIS, and PAFnow. These tools typically provide a 2D user interface with some being Web-based interfaces (e.g., Celonis, UiPath, Apromore), whereas our contribution offers a VR-based solution that can be more engaging and accessible for collaboration with non-experts.

## III. Solution

Our solution for PM visualization incorporates VR. VR provides an unlimited space for visualizing a complex set of events, models, and processes and their interrelationships simultaneously in a spatial structure. This provides an immersive, intuitive experience for digital process model visualization and analysis in a 3D space viewable from different perspectives that supports sharing and collaboration among stakeholders. As to benefits of an immersive VR experience vs. 2D for an analysis task, [22] investigated a software analysis task that used a Famix metamodel of Apache Tomcat source code dependencies in a force-directed graph. They found that VR does not significantly decrease comprehension and analysis time nor significantly improve correctness (although fewer errors were made). While interaction time was less efficient, VR improved the UX (user experience), being more motivating, less demanding, more inventive/innovative, and more clearly structured.

To provide context for our solution, our generalized solution concept for VR-PM is shown in Figure 1. VR-PM extends VR-ProcessMine capabilities and leverages our generalized VR Modeling Framework (VR-MF) [7] solution concept. It consists of a VR-based domain-independent hypermodeling framework, which addresses four primary aspects that require special attention when modeling in VR: visualization, navigation, interaction, and data retrieval. Our prior solutions in the EA and BP space, VR-EA [7] provides specialized direct support and mapping for EA models in VR, including both ArchiMate as well as BPMN via VR-BPMN [6]. VR-EAT [8] extends this further with integration of EA tools for accessing dynamically generated diagrams and models from an EA tool in VR. VR-EA+TCK extends these capabilities by integrating further enterprise knowledge, information, and content repositories such as a Knowledge Management System (KMS) and/or an Enterprise Content Management System (ECMS). Our other VR solutions address Software Engineering (SE) and Systems Engineering (SysE) areas.

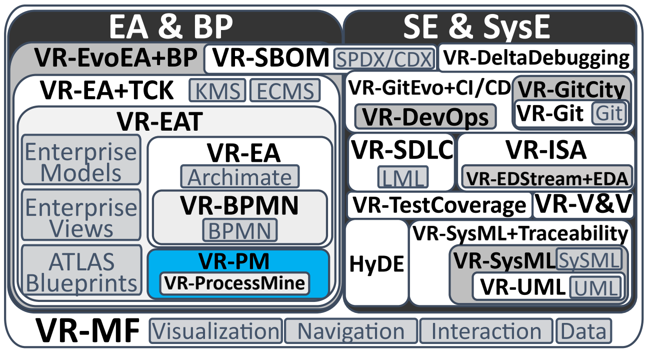


Figure 1. The VR-PM solution concept (blue) in relation to our prior VR solution concepts.

In order to support PM and visual analysis, the VR-PM solution should exhibit the following capabilities:

- *Log file import*: event logs in different event log formats can be imported and processed;

- *Multiple simultaneous analyses*: different event log types (versions or models) can be loaded in order to compare them directly;
- *3D visualization*: elements should be depicted in 3D to support an immersive observation experience;
- *Free element placement*: an individual analysis should be movable in VR space so that they can be compared in locality with another analysis of interest;
- *Hide/show analyses*: to minimize visual clutter, analyses can be hidden and then seen again;
- *Variant depiction*: variants can be analyzed separately as Directly-Follows Graph (DFG) depictions;
- *Colored hot spots*: events are colored to indicate their relative frequency;
- *Log event model*: for reference, show the events before their transformation to other models; and
- *Both formal and practical process models:* offer Petri Nets as a highly formal and rigorous process model for in-depth analysis, as well as a BPMN model that can be practically used and readily understood by novice stakeholders.

### A. Visualization in VR

In order to differentiate process variants (depending on the analysis being done), our visualization concept for VR depicts each of these on separate vertical plates standing on a common hyperplane representing a single process. This permits any plate to be selected, moved, and compared with others of interest. Furthermore, since the number of process variants can be very large, it leverages the unlimited space in VR, allowing the hyperplane to depict many process variants at once. All process variants are initially equally spaced on the hyperplane and can be compared with each other.

### B. Navigation in VR

The immersion afforded by VR requires addressing how to navigate the space while reducing the likelihood of potential VR sickness symptoms. Thus, two navigation modes are included in the solution: the default uses gliding controls, enabling users to fly through the VR space and view objects from any angle they wish. Alternatively, teleporting permits a user to select a destination and be instantly placed there (i.e., by instantly moving the camera to that position); while this can be disconcerting, it may reduce the susceptibility to VR sickness for those prone to it that can occur when moving through a virtual space.

### C. Interaction in VR

Since interaction with VR elements has not yet become standardized or intuitive, in our VR concept, user-element interaction is handled primarily via the VR controllers and a virtual tablet. Our VR-Tablet provides detailed context-specific element information, and can provide a virtual keyboard for text entry fields (via laser pointer key selection) when needed. An affordance in the form of an anchor (sphere) is provided on a corner of any hyperplane or model, which if selected can be used to reduce visual clutter by collapsing (hide) or expanding (show) that object, or the anchor can be used to place the object elsewhere.

## IV. Realization

Our solution prototype is partitioned into the Data Hub, our backend for PM and data processing, and the Unity front end responsible for VR visualization (see Figure 2). The Data Hub, based on Python, prepares datasets for visualization. The python library pm4py (Process Mining for Python) [21] is used to convert the imported log files into data objects and data frames. REST APIs are used to access the backend pm4py functionality from the frontend Unity VR environment, which is implemented in C#.

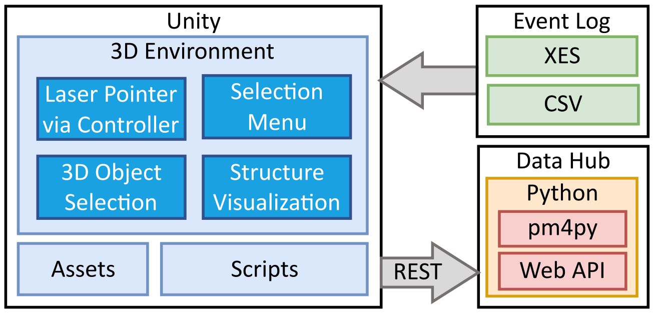


Figure 2. VR-PM logical architecture.

Via our VR-Tablet concept, various configuration options and inputs are supported, as shown in Figure 3.

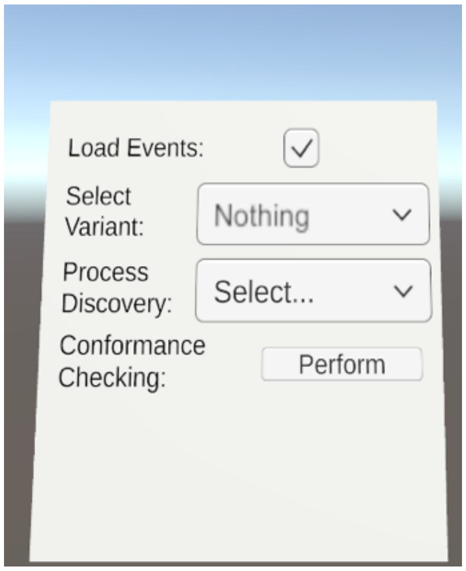


Figure 3. VR-Tablet showing event option, variant selection, process discovery, and conformance checking options.

### A. Event Logs

An event log is a hierarchical collection of traces. Each trace is a sequence of events that belong to a single case. An event log is required to extract any process variants, which vary in the process paths taken or the activity order. eXtensible Event Stream (XES) [23] or CSV event formats are supported. If no event log exists, but a process model as a BPMN or Petri Net, then via simulation pm4py can generate an event log of various executions.

Apromore [24] was also used to generate and test CSV file support, whereby the simulation offered the ability to specify probability selections pro gateway and included cost.

## *B. Process Discovery*

If an event log exists but no process model, then a process model (Petri Net and BPMN) can be generated via process discovery using PM4Py. However, if a process model such as BPMN exists and no event log, then an event log can be created by process simulation by selecting a Petri Net and then loading the event log in the VR-Tablet, then implicitly a simulation is run based on the model. If an event log already exists, then it is not overwritten. By default, 100 process executions are simulated using PM4Py to generate an event log.

Given an event log, process discovery using PM4Py can generate a Petri Net or a BPMN model based on the data, invoked as shown in Listing 1. Petri Net generation uses the heuristic miner algorithm, while BPMN uses the pm4py inductive miner algorithm. If only a BPMN model is provided as input, then it is converted to a Petri Net. Further detail on the algorithms is provided in the PM4Py documentation.

```
import pm4py

petri_net = pm4py.discover_petri_net_alpha(event_log)
bpmn = pm4py.discover_bpmn_inductive(event_log)
```

Listing 1. Code snippet showing process discovery impementation.

## *C. Process Conformance Checking*

Given a process model as a Petri Net and an event log, pm4py's process conformance via alignment is utilized, as shown in Listing 2. A replay fitness (i.e. alignment) metric score is returned by pm4py, whereby a max value of 1 indicates the behavior contained in the event log is fully conformant to the process model, whereas 0 indicates no conformance whatsoever, and anything in-between partial conformance. Further details are provided in the pm4py documentation.

```
from pm4py.algo.conformance.alignments.decomposed import
    algorithm
from pm4py.algo.evaluation.replay_fitness.algorithm import
    evaluate, Variants

conf = algorithm.apply(event_log, net, im, fm)

alignment_score = evaluate(conf,
                           variant=Variants.ALIGNMENT_BASED)
```

Listing 2. Code snippet showing process discovery impementation.

## *D. Process Variant Plates*

To support process variant analysis, a DFG algorithm provides a summary of all process event transitions and variants and how often each process variant was executed. Figure 4. shows a DFG-based process map visualization result using pm4py. A node represents an event. A graph consists of a set of transitions between a set of nodes.

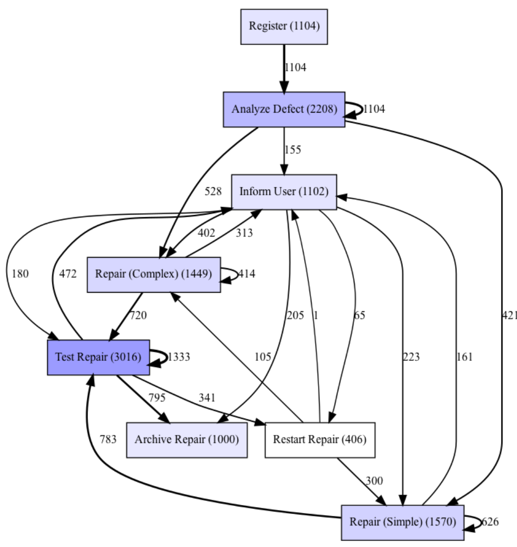


Figure 4. Example DFG-based process map result from pm4py.

Listing 3 shows the result of the parsed dataset providing the number of transitions occurring between two events.

```
Counter({
    ('Test Repair', 'Test Repair'): 1333,
    ('Register', 'Analyze Defect'): 1104,
    ('Analyze Defect', 'Analyze Defect'): 1104,
    ('Test Repair', 'Archive Repair'): 795,
    ('Repair (Simple)', 'Test Repair'): 783,
    ('Repair (Complex)', 'Test Repair'): 720,
    ('Analyze Defect', 'Repair (Complex)'): 528,
    ...
})
```

Listing 3. Snippet of a parsed DFG dataset.

For process variants, data is converted to a dictionary, whereby all duplicates are removed so that each node exists only once in the graph. The recursive list of fan-in relations (nodes reaching this node) together with their occurrence frequency provides the basis for a weighted directed graph as shown in Listing 4. Aggregating the total occurrences across all incoming transitions of a node (event) provides a total frequency of that event across all process instances.

```
{
  "Register": {
    "start":
  },

  "Analyze Defect": {
    "Register": 1104,
    "Analyze Defect": 1104
  },

  "Repair (Complex)": {
    "Analyze Defect": 528,
    "Repair (Complex)": 414,
    "Inform User": 402,
    "Restart Repair": 105
  },
    ...
}
```

Listing 4. Recursive list of fan-in relations.

## V. Evaluation

The evaluation of our VR solution concept is based on the design science method and principles [25], in particular a viable artifact, problem relevance, and design evaluation (utility, quality, efficacy). Our case study focuses on addressing the aforementioned challenges identified in the PM Manifesto [5], which are: Improving Usability for Non-experts, Improving Understandability for Non-experts, and Combining Process Mining with other Types of Analysis. The case scenarios are:

- Event Log Visualization and Comprehension
- Process Discovery and Simulation Scenario
- Process Conformance Checking Scenario
- Detailed Process Variant Analysis Scenario
- Combining PM with other Types of Analysis

### A. Event Log Visualization and Comprehension

Event logs contain data in formats that are not readily comprehensible for stakeholders. The event log is visually depicted in the form of a 3D nexus rather than text-based data, improving understandability and comprehensibility, as shown in Figure 5. A 3D nexus enables larger graphs to not take up as much space and can be immersively explored without requiring large navigational distances. Events are depicted as spheres (white, green for start, and red for end), directional edges as pipes (darker indicates the source or from side) representing transitions to indicate the sequence direction.

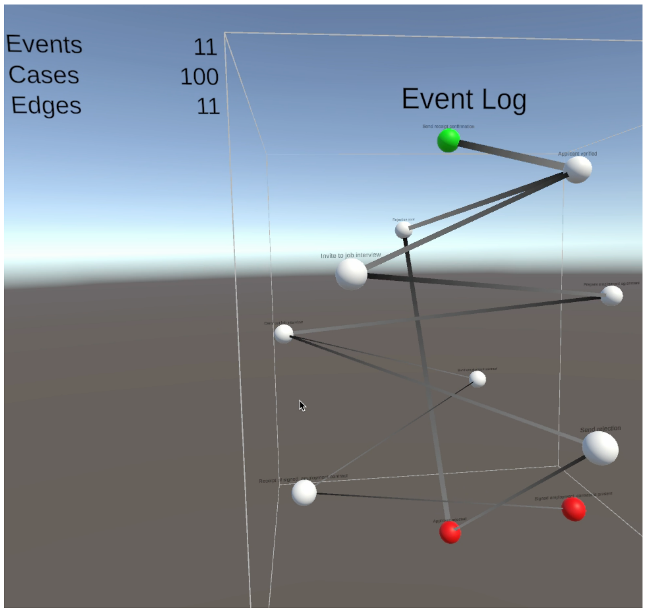


Figure 5. Event log visualized in a nexus form with metrics on top left.

Selecting an event depicts detailed information about it, as shown in Figure 6. Examples of relevant process information include cost information or task duration information, if available in the event log it can be depicted.

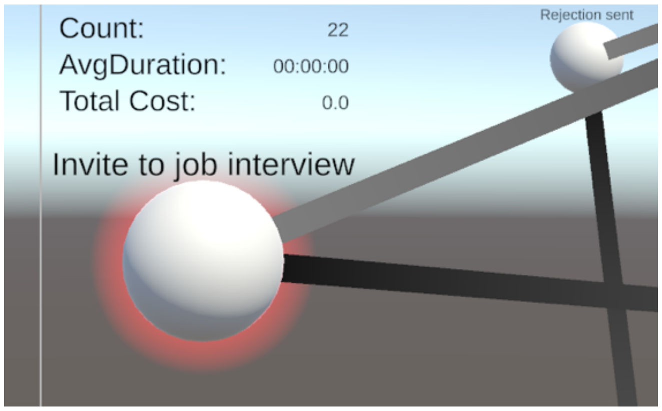


Figure 6. Selected event in event log nexus depicts detailed information related to the event.

### B. Process Discovery and Simulation Scenario

Process discovery generates a process model based on event data. To demonstrate process simulation, a BPMN model of a Hiring Process for ACME AG [26] was used, shown in Figure 15. This is depicted in VR-PM in 3D in Figure 16. A Petri Net generated by PM4Py can also be depicted in VR-PM, as shown in Figure 17.

To support basic process variant analysis, the event log can be used to analyze variants within the BPMN model, as shown in Figure 18. Here, activities for any selected variant are highlighted. Only one selected variant is highlighted at a time in BPMN, limiting comparison but supporting simplicity. For more detailed variant analysis, separate variant plate visualization is offered as described later.

To support usability, any process model elements (Petri Net or BPMN) can be repositioned, as shown for the highlighted element in Figure 7 and Figure 8.

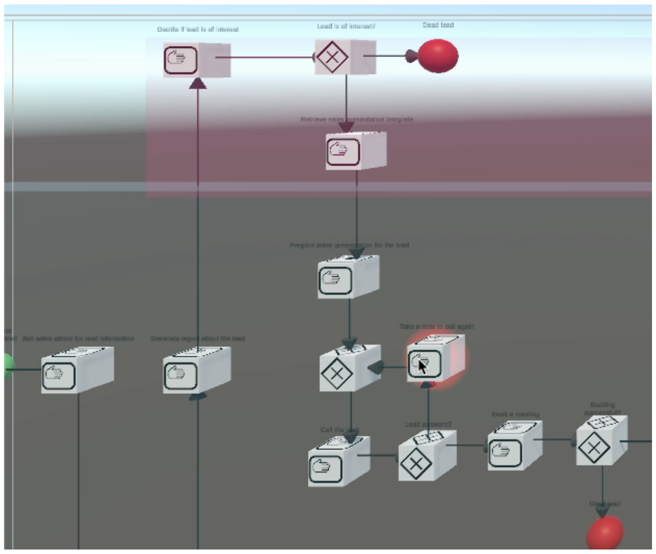


Figure 7. Selection of BPMN process model elements to reposition from default layout.

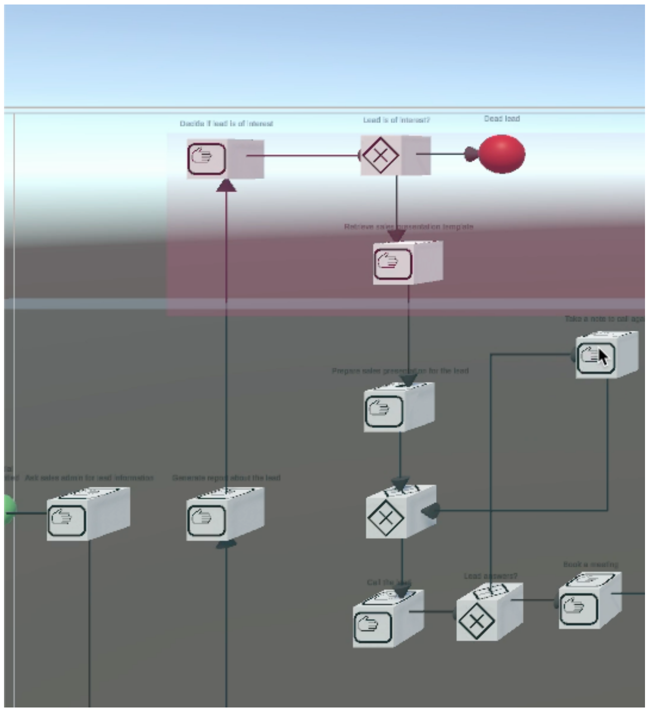

Decide if lead is of interest
Lead is of interest?
Dead lead
Retrieve sales presentation template
Take a note to call aga
Prepare sales presentation for the lead
Ask sales admin for lead information
Generate report about the lead
Call the lead
Lead answers?
Book a meeting

Figure 8. BPMN process model element repositioned.

Process simulation generates traces, which can vary in their event sequences, as shown in Listing 5 versus Listing 6. These result in process variants, each of which contains one or more process instances that followed the same sequence. Selecting a variant in the VR-Tablet result in its path being highlighted in the process model, as was shown in Figure 18. Since the generated BPMN does not necessarily convey all the data available in the event log, a more detailed variant analysis using panes with DFGs is offered, which is described later.

```
<?xml version="1.0" encoding="utf-8" ?>
<log xes.version="1849-2016" xes.features="nested-attributes" xmlns="http://www.
xes-standard.org/">
  <trace>
    <string key="concept:name" value="0" />
    <event>
      <string key="concept:name" value="Ask sales admin for lead information" />
      <date key="time:timestamp" value="1970-04-26T18:46:40+00:00" />
      <string key="case:concept:name" value="0" />
    </event>
    <event>
      <string key="concept:name" value="Check company's financial statements" />
      <date key="time:timestamp" value="1970-04-26T18:46:41+00:00" />
      <string key="case:concept:name" value="0" />
    </event>
    <event>
      <string key="concept:name" value="Check company's web site" />
      <date key="time:timestamp" value="1970-04-26T18:46:42+00:00" />
      <string key="case:concept:name" value="0" />
    </event>
    <event>
      <string key="concept:name" value="Generate report about the lead" />
      <date key="time:timestamp" value="1970-04-26T18:46:43+00:00" />
      <string key="case:concept:name" value="0" />
    </event>
    <event>
      <string key="concept:name" value="Decide if lead is of interest" />
      <date key="time:timestamp" value="1970-04-26T18:46:44+00:00" />
      <string key="case:concept:name" value="0" />
    </event>
  </trace>
```

Listing 5. Example trace snippet from a generated XES event log.

```
<trace>
  <string key="concept:name" value="52" />
  <event>
    <string key="concept:name" value="Ask sales admin for lead information" />
    <date key="time:timestamp" value="1970-04-26T18:54:40+00:00" />
    <string key="case:concept:name" value="52" />
  </event>
  <event>
    <string key="concept:name" value="Check company's financial statements" />
    <date key="time:timestamp" value="1970-04-26T18:54:41+00:00" />
    <string key="case:concept:name" value="52" />
  </event>
  <event>
    <string key="concept:name" value="Check company's web site" />
    <date key="time:timestamp" value="1970-04-26T18:54:42+00:00" />
    <string key="case:concept:name" value="52" />
  </event>
  <event>
    <string key="concept:name" value="Generate report about the lead" />
    <date key="time:timestamp" value="1970-04-26T18:54:43+00:00" />
    <string key="case:concept:name" value="52" />
  </event>
  <event>
    <string key="concept:name" value="Decide if lead is of interest" />
    <date key="time:timestamp" value="1970-04-26T18:54:44+00:00" />
    <string key="case:concept:name" value="52" />
  </event>
  <event>
    <string key="concept:name" value="Retrieve sales presentation template" />
    <date key="time:timestamp" value="1970-04-26T18:54:45+00:00" />
    <string key="case:concept:name" value="52" />
  </event>
  <event>
    <string key="concept:name" value="Prepare sales presentation for the lead" />
    <date key="time:timestamp" value="1970-04-26T18:54:46+00:00" />
    <string key="case:concept:name" value="52" />
  </event>
  <event>
    <string key="concept:name" value="Call the lead" />
    <date key="time:timestamp" value="1970-04-26T18:54:47+00:00" />
    <string key="case:concept:name" value="52" />
  </event>
  <event>
    <string key="concept:name" value="Book a meeting" />
    <date key="time:timestamp" value="1970-04-26T18:54:48+00:00" />
    <string key="case:concept:name" value="52" />
  </event>
  <event>
    <string key="concept:name" value="Do the meeting" />
    <date key="time:timestamp" value="1970-04-26T18:54:49+00:00" />
    <string key="case:concept:name" value="52" />
  </event>
</trace>
```

Listing 6. Further trace snippet sequence from a generated XES event log.

To demonstrate scalability and immersion in VR, a larger BPMN insurance company model [25], containing 27 activities and 13 gateways, was evaluated, shown in Figure 19. This was loaded into VR-PM as shown in Figure 20.

### C. *Process Conformance Checking Scenario*

Process conformance checking can be initiated by the button on the VR-Tablet, as shown in Figure 21. The results are displayed at the top of the VR-Tablet, showing the percentage of fitting traces, the average fitness, and the log fitness. This figure also shows multiple model representations (event log nexus, BPMN, and Petri Net) simultaneously, which may be helpful when analyzing conformance issues.

### D. *Detailed Process Variant Analysis Scenario*

For detailed DFG-based process variant analysis, the dataset consisted of randomly generated process variants based on a software defect repair process (a snippet is shown in Listing 7). A process variant represents multiple process instances, whereby any process instance exhibits the same sequence (or node transition) order.

```
{
"variant": "Register,Analyze Defect,Analyze Defect,Inform User,
Repair (Complex),Repair (Complex),Test Repair,Test Repair,Archive
Repair",
"case:concept:name": 78
},
{
"variant": "Register,Analyze Defect,Analyze Defect,Repair (Simple),
Inform User,Repair (Simple),Test Repair,Test Repair,Archive
Repair",
"case:concept:name": 75
},
{
"variant": "Register,Analyze Defect,Analyze Defect,Repair (Simple),
Repair (Simple),Test Repair,Test Repair,Inform User,Archive
Repair",
"case:concept:name": 67
},
{
"variant": "Register,Analyze Defect,Analyze Defect,Repair
(Complex),Repair (Complex),Test Repair,Inform User,Test Repair,
Archive Repair",
"case:concept:name": 64
},
{
"variant": "Register,Analyze Defect,Analyze Defect,Repair (Simple),
Repair (Simple),Test Repair,Inform User,Test Repair,Archive
Repair",
"case:concept:name": 41
},
{
"variant": "Register,Analyze Defect,Analyze Defect,Repair (Simple),
Repair (Simple),Test Repair,Test Repair,Restart Repair,Repair
(Simple),Inform User,Repair (Simple),Test Repair,Test Repair,
Archive Repair",
"case:concept:name": 29
},
{
"variant": "Register,Analyze Defect,Analyze Defect,Inform User,
Repair (Simple),Repair (Simple),Test Repair,Test Repair,Archive
Repair",
"case:concept:name": 28
},
{
"variant": "Register,Analyze Defect,Analyze Defect,Repair (Simple),
Repair (Simple),Test Repair,Test Repair,Inform User,Restart Repair,
Repair (Simple),Repair (Simple),Test Repair,Test Repair,Archive
Repair",
"case:concept:name": 21
},
```

Listing 7. Dataset snippet of randomized process variants (in JSON).

For variant analysis, the initial variant plate shows a DFG with entire set of nodes and transition frequency; the plates behind it depict the different process variants, as shown in Figure 9. Each process variant is rendered on a separate 2D plate stacked horizontally (as depth), thus the third dimension allows the user to readily perceive the number of process variants.

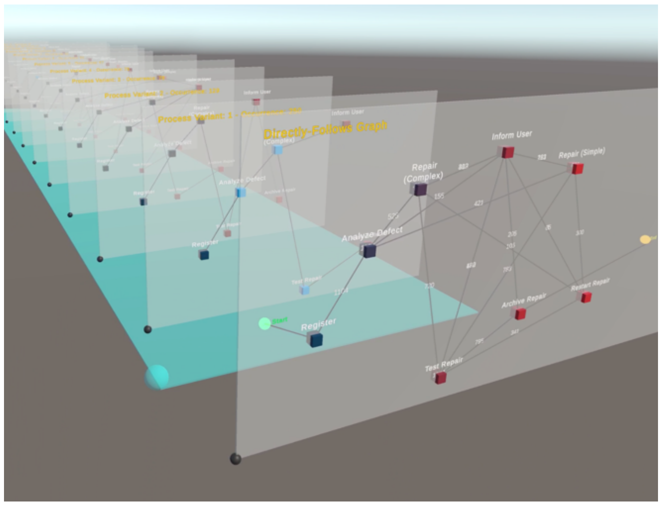


Figure 9. Initial plate shows DFG with entire set of nodes and transition frequency [1]. Each plate thereafter represents one process variant.

The variants are assigned a default ID with their occurrence frequency indicated on top, as seen in Figure 10.

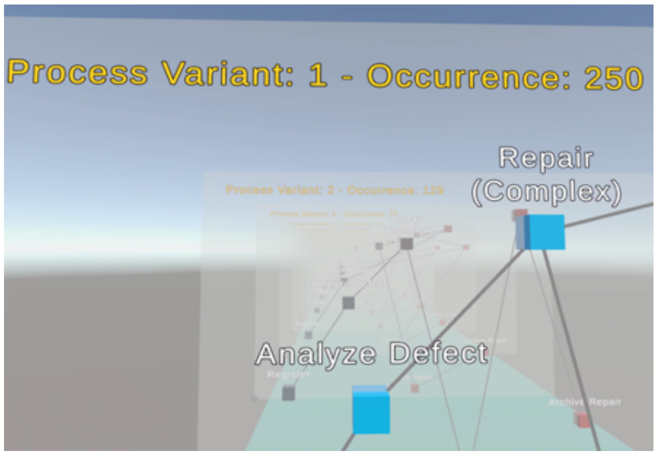

Process Variant: 1 - Occurrence: 250
Repair (Complex)
Analyze Defect
Process Variant: 2 - Occurrence: 120

Figure 10. Partial process variant showing occurence frequency [1].

The total number of input transitions to a node represent the total number of times that event occurred. Thus, the higher this number, the higher the frequency. To represent this visually, a ten-step color scale was used to map the frequency between low activity (blue) and high activity (red), analogous to mapping temperature, as shown in Figure 11. This can be used to quickly identify frequently occurring events in a process and help focus analysis and potential optimizations.

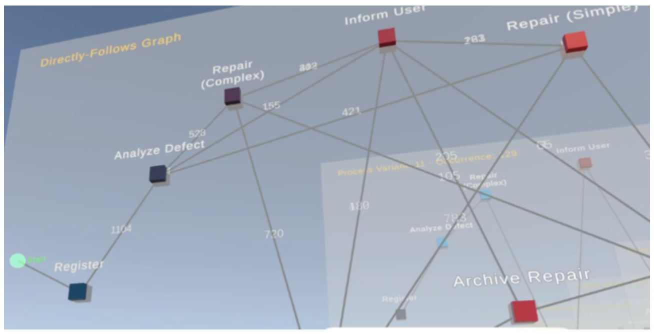


Figure 11. Node color and edges in a DFG, from VR-ProcessMine [1].

As plates on a hyperplane, the relation to the process model is supported by the hyperplane, yet each variant can be considered separately and compared with others by moving them with the affordance in the bottom left corner, as shown in Figure 12. One can also collapse any that are irrelevant to the analysis to reduce visual clutter.

A large set of different process models, and their variants, can be depicted simultaneously, supporting larger cross-process analysis scenarios as seen in Figure 13.

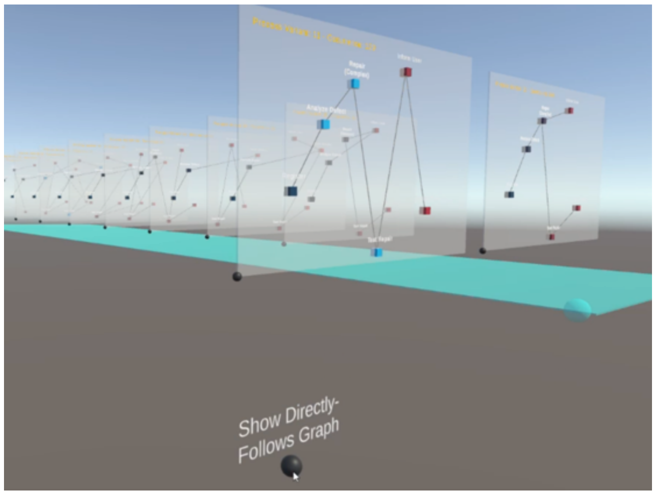


Figure 12. Anchor control for variant comparison or collapsing/expanding [1].

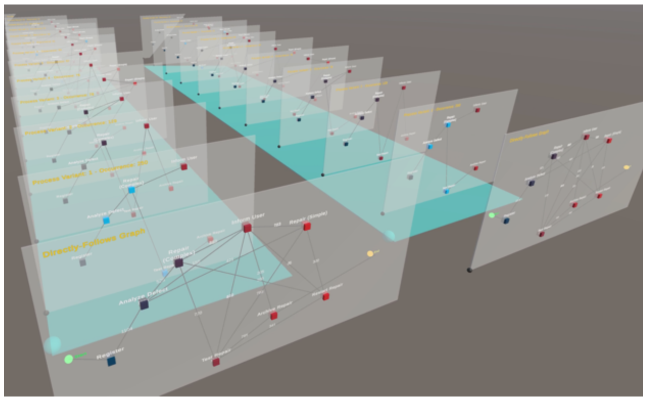


Figure 13. Different processes can be displayed via different hyperplanes [1].

## E. Combining PM with other Types of Analysis

PM analysis can provide significant operational insights, yet additional enterprise knowledge or data may also be relevant. Towards addressing this challenge, one advantage of VR's unlimited space is its ability to represent multiple heterogenous models simultaneously (Figure 21), which non-experts can immersively explore. Further heterogeneous hypermodeling capabilities in VR are exemplified with our VR-EAT [8], which is based on diagrams generated by Atlas and transformed in VR, and with our VR-EA [7], offering VR-based ArchiMate diagrams, as shown in Figure 14. Thus, VR-PM models can be placed side-by-side with other enterprise models in VR, making deeper cross-model and cross-domain analysis readily accessible to stakeholders, while enabling operational insights to guide such analyses.

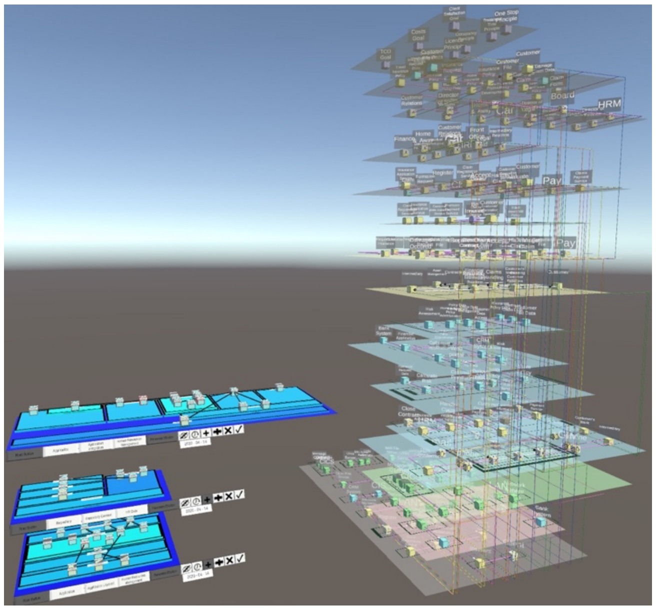


Figure 14. VR-EAT [8] enterprise hypermodeling and analysis: Atlas-based diagrams (left) and VR-EA [7] using ArchiMate-based diagrams (right).

## F. Discussion

Towards addressing the aforementioned three PM challenges, our solution concept seeks to improve the usability and understandability for non-experts and enabling the combination with other analyses. It does this by providing an immersive VR experience addressing visualization, navigation, interaction, and data retrieval. This was shown via common PM case scenarios:

- Event Log Visualization and Comprehension
- Process Discovery and Simulation Scenario
- Process Conformance Checking Scenario
- Detailed Process Variant Analysis Scenario
- Combining PM with other Types of Analysis

On the basis of these PM support scenarios, the solution can further PM stakeholder engagement and collaboration, such that they can intuitively interact with and comprehend process models and evidence-based process event data, and readily incorporate these in their daily work routines.

## VI. Conclusion

The increasing digitalization in enterprises and organizations implies that the business, operational, and industrial processes executed will increasingly also become digitally accessible, offering a significant opportunity for further uptake and engagement of stakeholders, such as enterprise citizens. While current PM tools and techniques can provide valuable insights for optimizing (business) processes, these benefits can be hindered if insights are not readily accessible to a larger (non-expert) stakeholder set, such as those directly involved in performing these processes.

VR-PM contributes an immersive solution concept for visualizing and interacting with process models (BPMN and Petri Nets), process variants, process conformance checking, and PM data in VR. Our realization shows its feasibility, and the case-based evaluation provides insights into its capabilities towards addressing certain challenges described in the Process Mining Manifesto, in particular improving usability, understandability, and the potential to combine PM with other types of analysis.

Future work includes offering native editing capabilities to the process models, the depiction of message flows, more comprehensive PM analyses, deeper integration with our enterprise hypermodeling VR-EA-TCK and VR-BPMN solution concept, automatic filtering of process variants by a node or transition of interest, enhanced collaboration support, and a comprehensive empirical study.

## Acknowledgment

The authors would like to thank Maurizio Jenak, Laura Berroth, and Michael Scholl for their assistance with the design, implementation, figures, and evaluation.

## References

[1] R. Oberhauser, “VR-ProcessMine: Immersive Process Mining Visualization and Analysis in Virtual Reality,” In: Proceedings of the Fourteenth International Conference on Information, Process, and Knowledge Management (eKNOW 2022), IARIA, 2022, pp. 75-80.

[2] Grand View Research, Inc., “Business Process Management Market (2025 - 2030).” [Online]. Available from: https://www.grandviewresearch.com/industry-analysis/business-process-management-bpm-market 2025.11.28

[3] W. M. van der Aalst, “Process mining: a 360 degree overview,” In: Process mining handbook, Cham: Springer International Publishing, 2022, pp. 3-34.

[4] W. M. van der Aalst. Process Mining - Data Science in Action, Second Edition. Springer, 2016.

[5] W. V. D. Aalst, et al., "Process Mining Manifesto," In: International Conference on Business Process Management. Springer, Berlin, Heidelberg, 2011, pp. 169-194.

[6] R. Oberhauser, C. Pogolski, and A. Matic, "VR-BPMN: Visualizing BPMN models in Virtual Reality," In: Shishkov, B. (ed.) BMSD 2018. LNBIP, vol. 319, Springer, Cham, 2018, pp. 83–97. https://doi.org/10.1007/978-3-319-94214-8_6

[7] R. Oberhauser and C. Pogolski, "VR-EA: Virtual Reality Visualization of Enterprise Architecture Models with ArchiMate and BPMN," In: Shishkov, B. (ed.) BMSD 2019. LNBIP, vol. 356, Springer, Cham, 2019, pp. 170–187. https://doi.org/10.1007/978-3-030-24854-3_11

[8] R. Oberhauser, P. Sousa, and F. Michel, "VR-EAT: Visualization of Enterprise Architecture Tool Diagrams in Virtual Reality," In: Business Modeling and Software Design. BMSD 2020. LNBIP, vol 391, Springer, Cham, 2020, pp. 221-239. https://doi.org/10.1007/978-3-030-52306-0_14

[9] OMG, "Business Process Model and Notation (BPMN) Version 2.0.2," 2014.

[10] J.L. Peterson, "Petri nets," ACM Computing Surveys (CSUR), 9(3), 1977, pp. 223-252.

[11] W. Reisig. Understanding Petri Nets. Springer-Verlag Berlin, 2016. doi: 10.1007/978-3-642-33278-4.

[12] F. Milani, K. Lashkevich, F.M. Maggi, and C. Di Francescomarino, “Process mining: a guide for practitioners,” In: International conference on research challenges in information science, Springer, Cham, 2022, pp. 265-282.

[13] J. Vom Brocke, M. Jans, J. Mendling, and H.A. Reijers, „A five-level framework for research on process mining,” Business & Information Systems Engineering, 63(5), pp. 483-490, 2021.

[14] F. Eckhard, H. Wittges, and S. Rinderle-Ma, “Exploring the Role of Visualization in Process Mining: A Systematic Literature Review.” In: International Conference on Business Information Systems, Springer, Cham, 2025, pp. 165-178.

[15] J.C. Villalobos, S.J. Jensen, and H.A López-Acosta, “3DCR: A Tool for Immersive Process Mining,” In: 6th International Conference on Process Mining. CEUR-WS, 2024.

[16] J. Vogel and O. Thomas, "Towards a virtual reality-based process elicitation system," 40 Years EMISA 2019. LNI P-304, Gesellschaft für Informatik e.V., 2020, pp. 91-104, ISBN 978-3-88579-698-5

[17] C. Corea and P. Delfmann, “Geo-aware process mining,” In: Proc.Best Dissertation Award, Doctoral Consortium, and Demo. & Resources Forum at BPM, 2024, pp. 101-105.

[18] J. J. Roldán, E. Crespo, A. Martín-Barrio, E. Peña-Tapia, and A. Barrientos, "A training system for Industry 4.0 operators in complex assemblies based on virtual reality and process mining," Robotics and computer-integrated manufacturing, 59, 2019, pp. 305-316.

[19] B. F. Van Dongen, A. K. A. de Medeiros, H. Verbeek, A. Weijters, and W. van der Aalst, “The ProM Framework: A new era in Process Mining Tool Support,” In: International Conference on Application and Theory of Petri Nets, Springer, 2005, pp. 444–454.

[20] M. La Rosa et al., "APROMORE: An advanced process model repository," Expert Systems with Applications, 38(6), 2011, pp. 7029-7040.

[21] A. Berti, S. van Zelst, and D. Schuster, "PM4Py: A process mining library for Python," Software Impacts, 17, p. 100556, 2023. doi: 10.1016/j.simpa.2023.100556

[22] R. Müller, P. Kovacs, J. Schilbach, and D. Zeckzer, "How to master challenges in experimental evaluation of 2D versus 3D software visualizations," In: 2014 IEEE VIS International Workshop on 3Dvis (3Dvis), IEEE, 2014, pp. 33-36.

[23] IEEE, "IEEE Standard for eXtensible Event Stream (XES) for Achieving Interoperability in Event Logs and Event Streams," in IEEE Std 1849-2023, 2023.

[24] Apromore. [Online]. Available from https://academic-cloud.apromore.org 2025.11.28

[25] A.R. Hevner, S.T. March, J. Park, and S. Ram, "Design science in information systems research," MIS Quarterly, 28(1), 2004, pp. 75-105.

[26] D. Sola, C. Warmuth, B. Schäfer, P. Badakhshan, J.-R. Rehse, and T. Kampik, “SAP Signavio Academic Models: A Large Process Model Dataset,” in Process Mining Workshops - ICPM 2022 International Workshops, October 23-28, 2022, Revised Selected Papers, 2022, vol. 468, pp. 453–465. doi: 10.1007/978-3-031-27815-0_33.

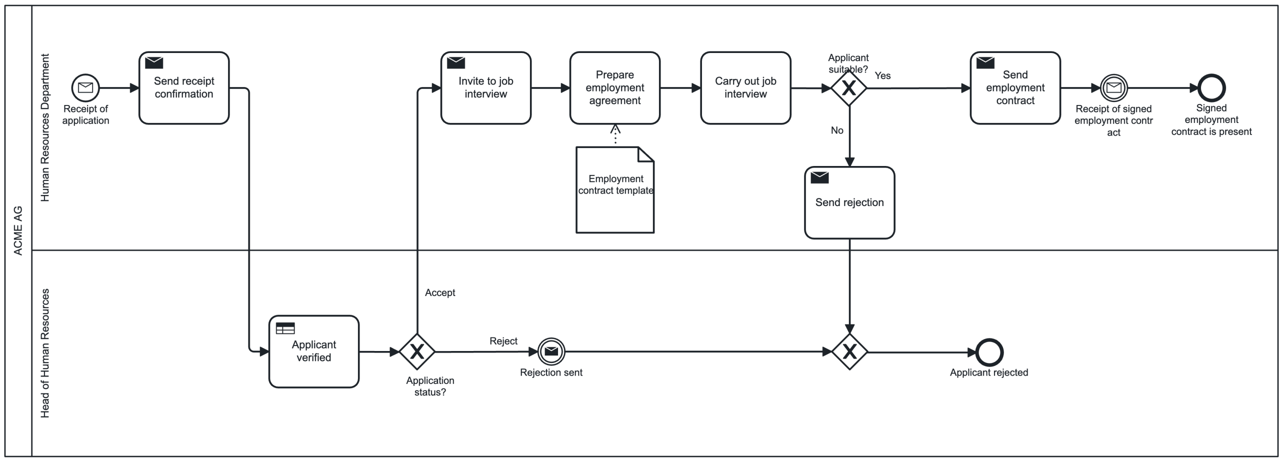


Figure 15. A BPMN model diagram of a hiring process for ACME AG [26].

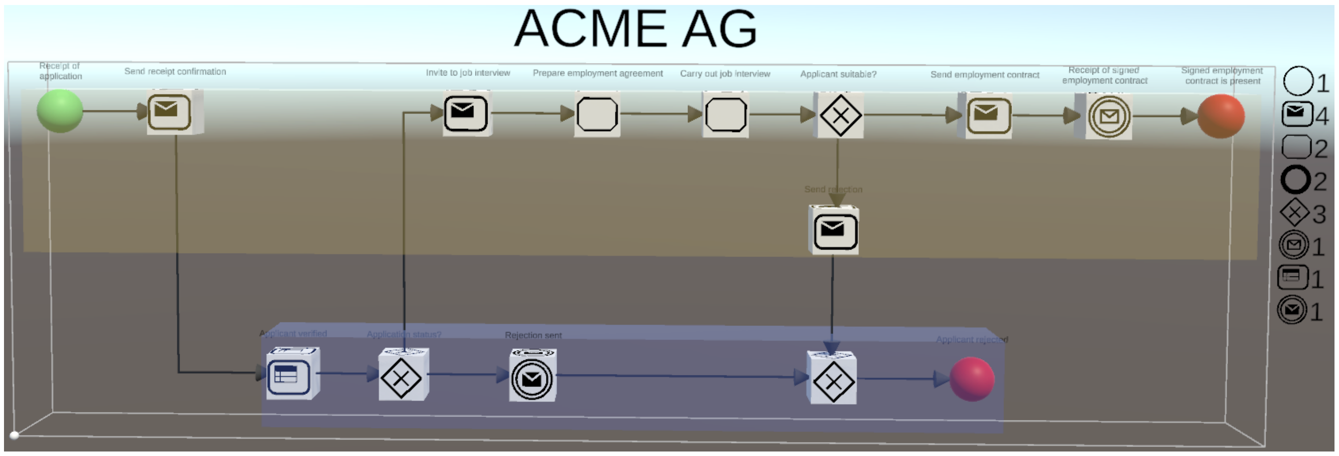


Figure 16. A BPMN model (ACME AG) depicted in 3D in VR-PM, showing count metrics on the legend on the right.

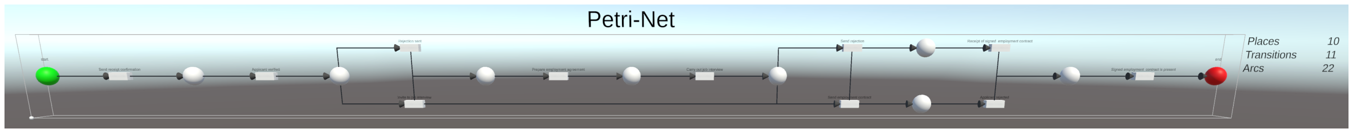


Figure 17. Petri Net 3D visualization in VR-PM.

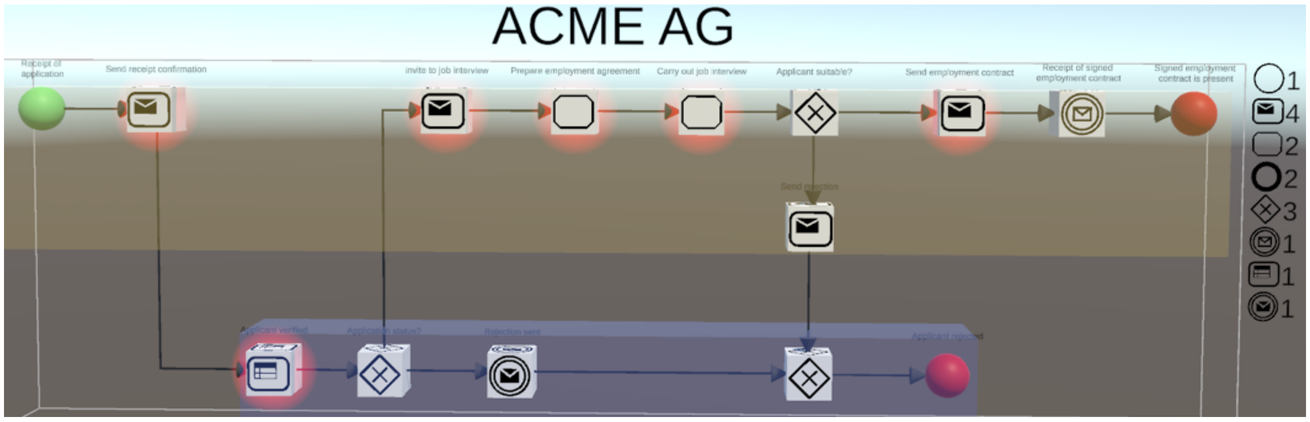


Figure 18. VR-PM variant selection within the BPMN model highlights its corresponding activities.

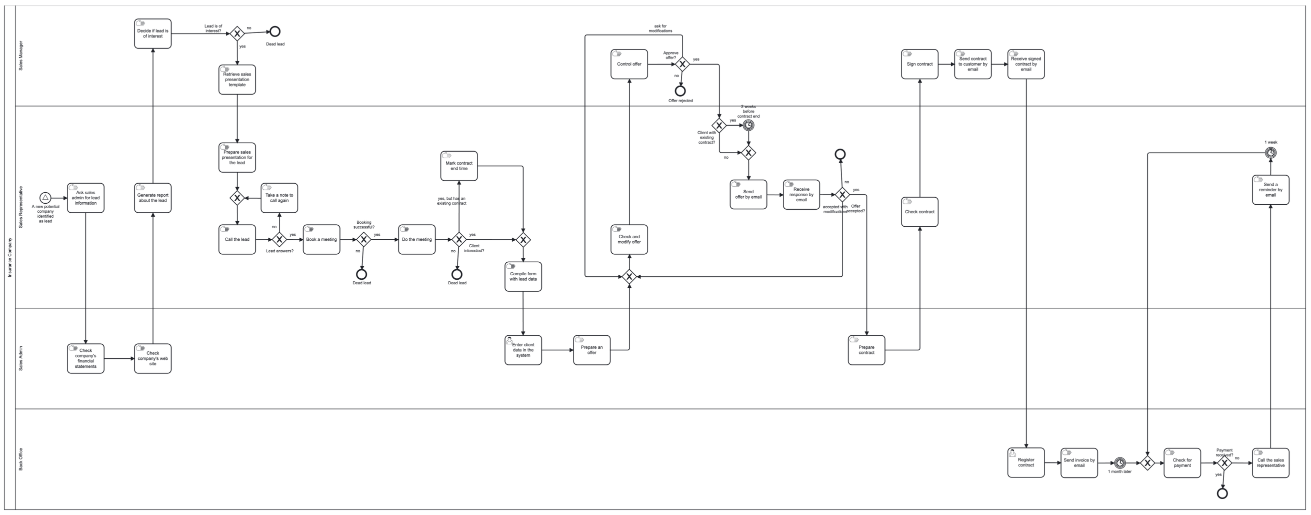

Insurance Company
Sales Manager
Sales Representative
Sales Admin
Back Office
Decide if lead is of interest
Lead is of interest?
no
yes
Dead lead
Retrieve sales presentation template
Prepare sales presentation for the lead
A new potential company identified as lead
Ask sales admin for lead information
Generate report about the lead
Take a note to call again
Call the lead
Lead answers?
Book a meeting
Booking successful?
Do the meeting
Mark contract end time
yes, but has an existing contract
Client interested?
Compile form with lead data
Check company's financial statements
Check company's web site
Enter client data in the system
Prepare an offer
ask for modifications
Control offer
Approve offer?
Offer rejected
Check and modify offer
Client with existing contract?
2 weeks before contract end
Send offer by email
Receive response by email
accepted with modifications
Offer accepted?
Prepare contract
Check contract
Sign contract
Send contract to customer by email
Receive signed contract by email
1 week
Send a reminder by email
Register contract
Send invoice by email
1 month later
Check for payment
Payment received?
Call the sales representative

Figure 19. Large BPMN model diagram (Insurance Company) [26].

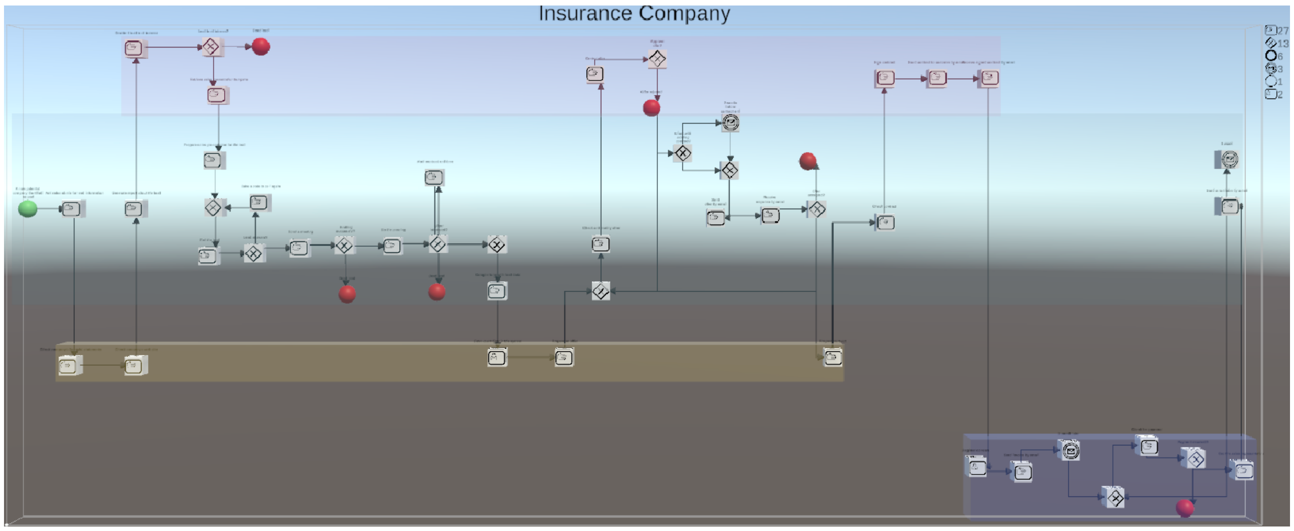
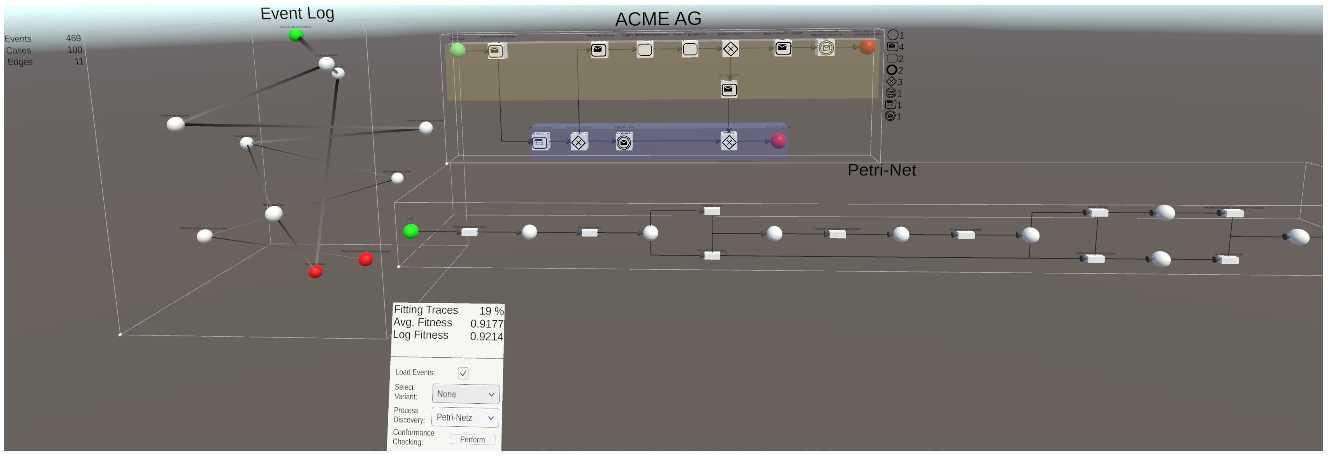


Figure 20. The large BPMN model (Insurance Company) depicted in 3D in VR-PM.



Events 469
Cases 100
Edges 11

Fitting Traces 19 %
Avg. Fitness 0.9177
Log Fitness 0.9214

Load Events: ✓
Select Variant: None
Process Discovery: Petri-Netz
Conformance Checking: Perform

Figure 21. VR-PM showing multiple model representations simultaneously (event log nexus left, BPMN top, Petri Net bottom) with compliance metrics.