TASK Quarterly 30 (3) 2026

Automatic extraction of the process of changes to legal acts

Julia Żęgota*

Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Narutowicza 11/12, 80-233 Gdańsk, Poland

*s184707@student.pg.edu.pl

https://doi.org/10.34808/tq2026/30.3/c

Abstract

In an era of frequent legislative changes and a growing volume of published laws, automating amendment-related processes is essential for effective analysis and interpretation of legal documents. Manually tracking and describing relationships between new and amending acts in the Journal of Laws of the Republic of Poland is time-consuming and prone to error. These tasks can be supported through metadata extraction tools and process mining algorithms. This work presents a system for automatic extraction and visualization of amendment processes. Parsing tools were applied to obtain and process the texts of laws, transforming the extracted metadata into event logs and a Directly-Follows Graph (DFG) for process mining. A web-based application provides graphical representation of the data. The resulting tool automatically generates a graph illustrating relationships between acts and their evolution over time. A pilot evaluation with legal professionals and non-expert users provides preliminary evidence of improved efficiency and readability in identifying changes and visualizing legislative paths, though confirmation with a larger and more representative sample remains necessary. The tool offers practical support for lawyers and analysts working with historical and current legislative processes.

Keywords:

Process Mining, Process Discovery, Legal acts

1. Introduction

Modern rule-of-law states rely on a dynamically evolving system of legal acts to support the functioning of public institutions and social regulation. Legislative amendments occur frequently and exert significant influence on society. Their development can be monitored through the Journal of Laws of the Republic of Poland, the primary source of information on binding law. However, analyzing and tracking these changes remains time-consuming and error-prone when conducted manually, especially since amendments are published as PDF files that hinder automated processing.

As digitalization progresses, legal acts within European Union countries have become widely accessible and systematically organized in official repositories, enabling more effective automation of legislative data collection and processing, as well as the development of tools for analyzing and visualizing legislative changes. This article addresses the problem of automatically extracting amendment processes from legal acts using process mining techniques and presenting them graphically. The aim is to improve access to legal information and enhance transparency in the evolution of the legal system.

The motivation for this work arises from the increasing need to automate and simplify access to information on amendments in Polish legal acts, making them clearer and more accessible to both professionals and non-experts. A central challenge is the PDF format used for publication, which lacks structure and complicates the extraction of essential elements such as articles, sections, modifications, and references, thus requiring advanced text-processing methods.

Existing solutions, such as the Sejm API, provide programmatic access to metadata about acts published in the Journal of Laws. However, the metadata available through the API is not sufficient to fully reconstruct the detailed structure of legislative amendments, particularly the specific articles and provisions affected by each amendment [1]. Moreover, metadata that is manually maintained may be subject to inconsistencies and omissions, especially when describing detailed amendment information. These limitations hinder effective comparative analysis and the development of tools for tracking legislative histories.

Another notable gap is the absence of intuitive visualizations of amendment processes. Users currently lack tools that clearly depict the evolution of a legal act—for example, through graphs or timelines. Manual comparison of multiple versions is laborious and susceptible to mistakes, and no existing solution effectively integrates data-processing techniques with an accessible graphical interface.

In response to these challenges, the system proposed in this work combines PDF data extraction, structural transformation (in this case, into event logs) and visualization in the form of a graph, addressing a clear gap in existing approaches. The system enhances automation and efficiency in processing legislative information while contributing to broader accessibility of the law by simplifying its analysis and interpretation.

The objective of the study is to develop a complete, automated tool supporting the analysis of legislative changes—from publication, through formal extraction, to user-friendly visualization—facilitating a deeper understanding of the structure and dynamics of the Polish legal system.

The Introduction section presents the aim and justification of the work, describes the structure of legal acts in the Polish Journal of Laws and presents related studies across many other countries. The Methods section describes the key components of the approach, including metadata extraction, event logs, process mining, and graphical visualization, together with their implementation in the proposed system. The Results section outlines the outcomes of the work, the use cases, and the tests conducted, along with a presentation of the obtained results. The article concludes with Conclusions, which provides the main conclusions and outlines directions for future work.

1.1. Polish Journal of Laws and the Structure of Legal Acts

In the Polish legal system, binding normative acts are published in two official journals: the Journal of Laws of the Republic of Poland (Dziennik Ustaw, DU) and the Official Gazette of the Republic of Poland (Monitor Polski, MP). Each serves a distinct function determined by the legal force and purpose of the published documents. The Journal of Laws contains the most authoritative acts, including the Constitution, statutes, regulations, ratified international agreements, and decisions of the Constitutional Tribunal. In contrast, the Official Gazette publishes acts of internal or limited applicability, such as resolutions of the Council of Ministers or non-ratified international agreements [2].

Both journals include not only newly enacted laws but also amending acts, repeals, and consolidated texts. Access to their contents is provided through the Internet System of Legal Acts (ISAP) [3], which offers the text as published, consolidated versions of statutes, and links to the corresponding PDF files. ISAP also exposes machine-readable metadata via the Sejm ELI API, which implements the European Legislation Identifier (ELI) [1]. The ELI framework standardizes identifiers, metadata,

and publication formats to support interoperability and automated processing of legal information.

1.1.1 Structure of Legal Acts

Statutes published in the Journal of Laws follow the structure defined in the Rules of Legislative Technique [4]. Depending on their purpose, they may appear as:

- New (primary) acts, introducing a complete regulatory framework,
- Amending acts, modifying specific provisions of existing laws,
- Consolidated texts, officially incorporating all amendments adopted since the previous publication,
- Unofficial unified texts, prepared outside the legislative process and lacking legal authority [4].

A typical act consists of a title, substantive provisions (general and detailed), amendment provisions, transitional and adapting clauses, and final provisions governing the entry into force, including *Vacatio legis*, which is the period between the adoption and promulgation of a normative act and its entry into force [5]. Statutes may be further divided into chapters or other editorial units to improve clarity. This formalized structure ensures consistency across legislative documents and facilitates automated extraction and analysis.

1.2. Related Works

Research on automatic and semi-automatic extraction of elements from legal acts has developed intensively in many countries, driven by the growing need for transparent and efficient access to information on legislative changes. Traditional methods of publishing and archiving legal acts, especially in PDF format, often limit usability and hinder automated processing. This challenge affects not only legal professionals but also citizens who require clear and accessible information on amendments.

In Poland, several commercial systems such as Legalis [6], Inforlex [7] and Lex [8] provide tools for tracking legislative processes, though full functionality remains subscription-based.

International research, however, presents a wider range of automated approaches. The early, influential work "Automatically Processing Amendments to Legislation" by Arnold-Moore [9] introduced the use of Augmented Transition Networks and later developed templates for automatically generating amendments in "Automatic Generation of Amendment Legislation" [10]. Similar template-based or rule-based methods were proposed for Japanese [11] and Italian [12] [13] legislation, often using XML formats and achieving high processing accuracy. Semi-automatic systems, such as for Greek legislation [14], combine regular expressions with manual verification but remain limited by document quality and the need for human oversight. For Polish law, studies include clustering of Civil Code amendments [15] and machine-learning-based extraction of legislative changes [16], with transformer models achieving strong results.

Beyond amendment extraction, numerous projects focus on constructing legal knowledge graphs, such as Italian [17] and Vietnamese [18] legal knowledge bases, which transform normative documents into RDF structures to represent relationships between legal entities. Other approaches use hybrid extraction techniques [19], ontological modeling [20], or heterogeneous graph representations to structure and visualize legal information [21].

Finally, research also explores process-mining applications in legislative contexts, including event-log-based analysis of procedural workflows. In the UK, visual interfaces such as Graphie provide network-based representations of legislative dependencies, demonstrating the value of graphical tools for navigating complex legal systems [22].

2. Methods

This section presents the key theoretical and technological foundations used in the system, together with examples of libraries and tools applied in the implementation. The discussion is organized into four main thematic areas: extraction and preprocessing of legislative data, process modeling through event logs, process mining, and graphical visualization of the resulting structures.

The first part focuses on automatic metadata extraction from digital documents, particularly PDF files containing legal acts. This includes techniques for identifying relevant legislative content and transforming raw text into structured representations suitable for further analysis. The second part introduces the concept of event logs and explains how they are adapted for modeling legislative changes. The following subsection addresses key elements of process mining, which enable the discovery and analysis of amendment sequences in legal documents. Finally, the last section describes selected tools and frameworks used for graph-based visualization of dependencies between acts.

The system for extracting and visualizing legislative changes was implemented entirely in Python using the Visual Studio Code environment. Python was chosen for its readability, extensive ecosystem of libraries, and relevance to text processing and process mining tasks. A range of specialized libraries supported each stage of the

workflow, from data acquisition and semantic analysis to interactive presentation:

- PyMuPDF – used for extracting text, metadata, and structural elements from PDF files containing legislative acts,
- Pyparsing – used to define declarative parsing patterns reflecting the structure of legal language, such as article numbers, publication dates, and amendment formulations,
- PM4PY – initially applied for process-mining tasks, though later largely replaced by custom implementations tailored to the characteristics of legislative data,
- Dash Cytoscape – used to create interactive graph visualizations of relationships between acts and sequences of amendments.

These tools were integrated into a modular architecture that automates the complete workflow, from document acquisition to content analysis and visual presentation of changes. Custom solutions were developed to increase flexibility and adapt the system to the specific structure and semantics of legal documents. These include:

- Heuristic filtering of PDF content – a mechanism for identifying lines likely to contain legislative changes, based on characteristic phrasing, document structure, and typographic features, reducing the amount of text requiring deep parsing,
- Custom parsers for amendment detection – a set of Pyparsing-based parsers for identifying amendments, extracting publication and enactment dates, and recognizing deviations from default timelines,
- Custom loading of XES event logs – an extended XES importer that supports additional attributes relevant to legislative data, including trace-level information representing entire acts,
- Modification of the Directly-Follows Graph (DFG) discovery algorithm – an adapted version of the DFG algorithm that incorporates both event-level changes and structural elements representing complete acts, enabling more granular modeling of amendment dependencies,
- Dynamic interaction mechanisms in the graphical interface – tools enabling users to expand and collapse graph nodes, preserve viewport state, and trigger real-time updates through callback-based logic in Dash Cytoscape.

Together, these implementations form a flexible system for analyzing legislative changes, extending standard process-mining techniques with domain-specific features tailored to the complexity of legal documents.

2.1. Metadata Extraction

Legal acts, both contemporary and archival, are widely made available online in many countries in PDF format [23] [24]. These documents constitute a primary source of information in legal and procedural analyses, yet their structure, layout, and digitization quality significantly affect the ease of data extraction. Older acts are often stored as scanned images, making direct processing difficult, while newer acts, although digitally produced, retain a print-oriented layout rather than a machine-readable structure, which is characteristic of the PDF format.

In Poland, access to legal acts in PDF is provided through services such as the Sejm API [1] and the ISAP (Internetowy System Aktów Prawnych) publication system [3]. Although these documents generally adhere to publication standards, they are not optimized for automated processing and therefore require dedicated preprocessing to extract the relevant metadata.

Metadata extraction from PDF documents is made possible through specialized parsing libraries and document-processing tools. One such tool is PyMuPDF, a Python library enabling text extraction, word positioning, and structural interpretation of documents. It supports PDF as well as formats such as EPUB, MOBI, and DOCX, and, unlike many Python libraries, offers access to tables and vector graphics [25]. After obtaining raw text from a PDF, the next step is the identification of metadata—key descriptors of the document and its legislative structure. Typical metadata include:

- Act number (e.g., "2024/162"),
- Title of the legal act (e.g., "Act Amending the Public Finance Act"),
- Publication date,
- Effective date,
- Document type (amending act or new act),
- Amended articles, sections, and points,
- Optional information on article-level Vacatio legis.

Extracting metadata from PDFs presents a significant challenge due to inconsistencies in document structure and the lack of a standardized header format. For this reason, structural parsing alone is insufficient, and extraction must rely on regular expressions and semantic heuristics that capture patterns typical of Polish legislative language. Expressions such as "*Dz.U.*" (English: Journal of Laws), "*z dnia*" (English: dated), "*art.*" (English: article) as well as amendment-specific phrases like "*dodaje się*" (English: is added), "*otrzymuje brzmienie*" (English: is amended to read as follows), or "*skreśla się wyrazy*" (English: is deleted) serve as strong semantic indicators used to detect publication dates, journal numbers, amendment details, and other metadata

components.

The complexity of Polish grammar further complicates this process. Polish is a a morphologically rich and highly inflected language [26], and key legal terms frequently appear in multiple morphological forms depending on case, number, and syntactic context. For example, the Polish term *ustawa* (English: act) refers to a legislative act. Due to grammatical inflection in Polish, the term may appear in different forms, including *ustawy*, *ustawie*, and *ustawą* and need distinct pattern representations to ensure accurate matching. Similarly, amendment-related verbs occur in numerous conjugated forms, and references to articles or points often vary in structure due to declension or compound phrasing. To achieve reliable extraction, heuristics must account not only for fixed expressions but also for these systematic linguistic variations. This requires regex patterns that generalize across possible inflectional endings, as well as semantic rules capable of recognizing context even when surface forms differ [27].

Regular expressions provide a flexible mechanism for matching such text patterns and are widely used for information extraction from unstructured documents [28]. In this project, the implementation is based on Python, which offers built-in regex functionality as well as more advanced tools. Among them, the pyparsing library enables the definition of structured grammars for text analysis [29]. It is this library that was used to implement metadata and amendment extraction from legal texts, following the initial retrieval of content using PyMuPDF. The library was selected for its high-fidelity text extraction, which preserves line structure and improves readability, ultimately facilitating both regex-based and grammar-based processing.

2.2. Event Logs

Process mining relies on the analysis of event logs, which constitute the primary input for discovering, monitoring, and improving processes. For such analysis to be effective, the event logs must be correctly extracted, structured, and enriched with information describing the sequence and context of events—for example, amendments to documents, version changes, or procedural steps carried out by institutions. In this project, event logs serve as the formal representation of legislative changes extracted from legal acts.

As described in Wil van der Aalst's "Process Mining: Data Science in Action" [30], event logs may originate from diverse sources such as relational databases, text files, transactional logs, emails, or document management systems. Textual sources like legal acts pose specific challenges; their structure is highly heterogeneous, and interpreting them requires attention to both syntactic layout and semantic meaning. This complexity directly influenced how raw legislative data was transformed into event-log format within the implemented system.

The extraction process necessarily begins with raw data—in this case, PDF files drawn from public legal-information systems. These documents may vary in formatting, completeness, and level of digitization, and they rarely include standardized metadata fields. As van der Aalst emphasizes, data extraction should be guided by analytical questions rather than by the sheer availability of data [30]. For this project, the guiding questions concerned how to represent legislative amendments as a process and how to capture the evolution of legal acts over time.

Event logs are typically stored in standardized formats such as XES (eXtensible Event Stream) or MXML (Mining eXtensible Markup Language). Simpler formats, such as CSV, are also widely used and compatible with process-mining tools like PM4PY [31]. XES is particularly suitable because it supports extensible metadata through trace-level and event-level attributes, enabling rich contextual descriptions. Extensions, such as the Time module, allow timestamps to be encoded in a consistent way, facilitating the use of existing process-mining algorithms. For example:

```
<extension name="Time" prefix="time"
uri="http://www.xes-standard.org/time.xesext"
/>
```

Official XES extensions are documented by the IEEE Task Force on Process Mining [32].

In the context of legislative processes, event logs must capture both document-level metadata (act number, title, publication date, entry into force) and the detailed structure of amendments (articles, sections, and points). In this project, these data were obtained directly from PDF documents and, when available, supplemented with structured metadata provided through the European Legislation Identifier (ELI) API [1]. After extraction, the information was transformed into a custom event-log representation designed to reflect the nested and hierarchical nature of legislative amendments—something not directly supported by standard XES event models.

To accommodate these requirements, the system implements event logs using a set of custom Python classes, each representing a different level of legislative structure:

- **Act** – stores key information about a legal act, including publication and entry-into-force dates, identifiers (year, position), document type (new or amending act), title, a list of modified acts (if applicable), and optional entry-date exceptions for specific provisions,
- **ChangedAct** – represents all amendments made to a specific act, including the act title, amendment date,

identifiers, and a collection of modified articles,

- **ChangedArticle** – aggregates details about an altered article, including the type of modification (addition, deletion, modification), the article identifier, and a details object describing the content of the change,
- **ChangedArticleDetailsOnArticleChange** – stores the full text of a modified article,
- **ChangedArticleDetailsOnWordsChange** – records the exact wording before and after a textual change,
- **ChangedArticleDetailsOnAdd** – represents an added article, including its identifier and full content.

This hierarchical representation parallels the logic of event logs; each Act corresponds to a high-level trace, while individual amendments and article-level modifications function as events within that trace. This structure enabled the system to convert legislative amendments into event logs suitable for process-mining techniques, including graph-based discovery of directly-following relations (DFG) and temporal analysis of amendment sequences.

Through this approach, the project not only reconstructs the progression of legislative changes but also enables their analysis and visualization using the conventions of process mining, while simultaneously accommodating the complexities inherent to legal documents.

2.3. Process Mining

Process discovery is a central component of process mining, enabling the reconstruction of real organizational processes based on event data extracted from information systems. Unlike traditional modeling approaches—which rely on subjective observations, interviews, or documentation—process mining generates objective process models derived directly from event logs, such as system traces, document histories, or user interactions. This data-driven perspective makes it possible to understand how processes operate in practice, rather than how they are assumed or described.

The field gained significant momentum around 2010 with the formalization of process-mining techniques by Wil van der Aalst, widely recognized as the "Godfather of process mining." His earlier work on workflow patterns in the 1990s laid the foundations for modern automated process-discovery methods. Today, process mining is applied across a broad range of systems, including ERP, CRM, and SCM platforms, and replaces manually constructed, often incomplete models with representations grounded in actual observed behavior [33].

As a discipline, process mining encompasses three main tasks: discovery, conformance checking, and enhancement. Process discovery, the first and most fundamental of these tasks, focuses on deriving a process model from event logs without relying on prior knowledge. Although its early forms were based on manually collected, unverifiable data, contemporary process discovery is deeply integrated into automated process-mining workflows [33].

The discovery procedure begins with event logs, discussed in Section 2.2. Based on the extracted event data, algorithms such as the Alpha algorithm, heuristic miners, and inductive miners construct models that capture the observed sequences of activities [34]. However, these algorithms must operate under the constraints of real-world data, which may be incomplete, contain exceptional behavior, or include only a subset of all possible process executions.

Beyond visualizing actual process flows, process mining supports the detection of bottlenecks, inefficiencies, and deviations from expected procedures. This analytical capability makes it a powerful tool for iterative process improvement. It also forms the foundation for conformance checking—verifying alignment between event data and predefined reference models—and model enhancement, which integrates additional information such as performance metrics.

In the context of this project, process discovery serves as a method for reconstructing the evolution of legal acts. The legislative process—expressed through amendments, additions, deletions, and modifications—can be interpreted as a sequence of events that together form a complex procedural flow. By applying discovery techniques to event logs representing legislative changes, the system is able to identify patterns, sequences, and relationships among acts and amendments. This enables the construction of a Directly-Follows Graph (DFG), which visualizes the progression of modifications across legal documents and supports deeper analysis of legislative dynamics.

2.4. Graphical Visualization

The choice of visualization tools depends primarily on the type of analysis to be performed and the required degree of interactivity. For simple presentations or static publications, solutions based on Graphviz are often sufficient. However, applications designed for exploratory analysis or interactive user engagement benefit from more advanced libraries, such as Dash Cytoscape, which offer greater flexibility and extensive configuration options.

Among JavaScript-based visualization frameworks, D3.js and Cytoscape.js are particularly noteworthy. Both libraries provide rich functionality for constructing interactive and customizable network visualizations ren-

dered directly in web browsers. For Python users, Dash Cytoscape is especially advantageous, as it exposes the capabilities of Cytoscape.js without requiring JavaScript development or additional frontend tooling. Integrated with the Dash framework by Plotly, Dash Cytoscape enables the creation of Python-based web applications featuring advanced interaction logic, including dynamic filtering, selective display of nodes, click-based actions, and contextual information panels.

Within the system developed for analyzing and visualizing legislative changes, an interactive graphical interface was designed to support exploration of relationships among legal acts published in the Journal of Laws of the Republic of Poland. The interface is built using Dash Cytoscape, which combines interactive graph components with Python-driven application logic.

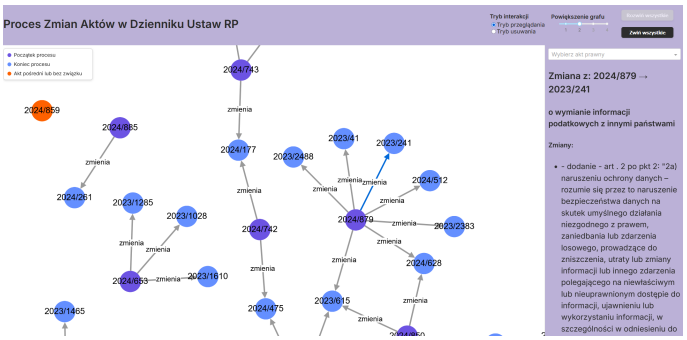


Figure 1: Interface view of the system for visualizing changes in legal acts

As illustrated in Figure 1, the central element of the interface is an interactive graph of legislative modifications. Nodes represent individual legal acts, while directed edges encode amendment relationships (e.g., "Act X amends Act Y", in Polish "Akt X zmienia Akt Y"). Node colors convey additional semantic information: green nodes indicate acts that have been amended by others, whereas red nodes represent acts that introduce amendments. The currently selected act is highlighted with a distinctive outline, improving its visibility within the network.

A search panel is placed on the right side of the interface, allowing users to locate a specific act by number. Upon selection, detailed information is displayed below the search bar, including the act's title, publication date, and a list of amendments it introduces, complete with article references and descriptions of the changes.

The upper section of the interface features global view-control buttons, Expand All and Collapse All, which adjust the visibility of nodes and edges across the entire graph. These controls rely on dynamic callbacks and enhance the clarity of the visualization by allowing users to tailor the level of detail to their analytical needs.

The adopted design prioritizes clarity and intuitive navigation, which are crucial when analyzing complex legislative structures. Through interactive components and customizable views, users can flexibly explore connections between acts, gaining both a high-level overview and detailed insight into legislative evolution.

2.5. System Architecture

To ensure scalability, clarity, and ease of further development, the system was designed as a set of modular, clearly separated components. This modular approach facilitates unit testing, supports independent evolution of system elements, and improves maintainability. The overall system architecture is illustrated in Fig.2.

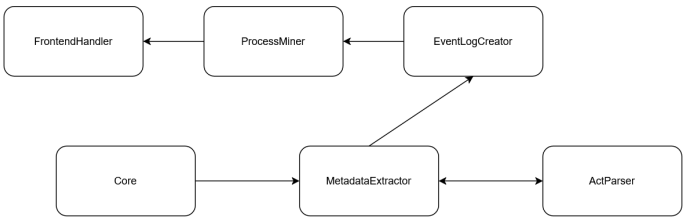


Figure 2: Diagram showing module dependencies within the system.

The system consists of six main modules, each responsible for a distinct stage of the extraction, transformation, and presentation pipeline for legislative data:

- **Core** – the primary entry point for the developer or user. It coordinates data flow between modules, integrates the system's components, and provides auxiliary functionalities such as configuration management, parameter propagation, and error logging,
- **ActParser** – the component responsible for analysing legislative documents. It incorporates a set of parsers based on regular expressions and domain-specific heuristics to identify key information, including the type of act, promulgation and entry-into-force dates, and the structure and details of amendments,
- **MetadataExtractor** – a module that transforms raw parser output into structured representations. It handles tasks such as converting dates to standard formats, classifying act types, and constructing internal data structures representing acts, articles, and amendment operations,
- **EventLogCreator** – the component that generates event logs from the structured metadata. It supports both creating new logs in the XES format and reading or updating existing logs, ensuring compatibility with process mining tools,
- **ProcessMiner** – the analysis module responsible for processing event logs and producing process models. It includes custom implementations for generating Directly-Follows Graphs (DFG), representing dependencies between successive amendment events. The resulting data are prepared for seamless integration

with the visualization layer,

- **FrontendHandler** – the presentation layer of the system. It manages the user interface, generates interactive graphs showing relationships between acts, and supports user interactions such as browsing amendment details or filtering the displayed information.

The architecture was designed to support the automated processing of large collections of legislative documents and to provide an intuitive presentation layer for end users. The layered design and strict separation of responsibilities improve code quality and enable straightforward extension of functionality, like adding support for additional sources of legal data or incorporating new analytical methods.

3. Results

This section presents the outcomes of the implemented solution, focusing on the practical evaluation of the system and its components. The results demonstrate how the proposed methods—spanning metadata extraction, event log construction, process mining, and graphical visualization—operate within a real-world legislative data environment.

In Section 3.2, use cases are described to illustrate the system's ability to process legal documents, extract structured information, and generate event logs suitable for downstream analyses. These examples highlight both typical scenarios and edge cases relevant to the characteristics of Polish legislative publications.

Subsequently, Section 3.3 examines the results of functional and performance tests conducted to assess correctness, robustness, and efficiency. Particular attention is given to the reliability of metadata extraction, the integrity of generated logs, and the usability of produced process models and visualizations. Together, these results provide empirical validation of the system's design assumptions and demonstrate its applicability to automated analysis of legal documents.

3.1. Comparison with Existing Solutions

Section 1.2 reviewed existing tools and research addressing the automation of legislative-change analysis, ranging from official metadata APIs and commercial legal-information systems to rule-based and machine-learning approaches developed for various jurisdictions. To clarify the novelty and practical value of the proposed system, Table 1 summarizes how these solutions compare with the system developed in this work across five criteria: automated extraction directly from published PDF acts, article-level granularity of amendment detail, process-mining-based modeling of amendment sequences, interactive graphical visualization, and open/free accessibility.

As shown in Table 1, existing official sources such as the Sejm ELI API expose only structured metadata without the detailed article-level content of amendments [1], while commercial systems such as Legalis, Inforlex, and Lex provide curated legal content, but restrict full functionality to paid subscriptions and do not expose an interactive, process-oriented view of legislative evolution. International rule-based and template-based approaches [9–12] and machine learning-based extraction methods developed for Polish legislation [15, 16] achieve accurate extraction of amendment content, but stop short of modeling the extracted data as a process or providing a graphical interface for exploration. Legal knowledge-graph projects [17, 18] and visual tools such as Graphie [22] address structuring and visualization separately, but none combines automated PDF-based extraction, article-level granularity, process-mining-based modeling of amendment sequences (via the DFG), and free interactive visualization within a single, openly accessible tool. This combination constitutes the novelty and practical contribution of the system presented in this work.

3.2. Use Cases

The following use cases illustrate how the system operates in practice and the types of interactions that typically occur during the analysis of legislative changes. Each scenario highlights a distinct aspect of the system's functionality, showing how automated processing and user-oriented features work together to support the examination of legal documents.

3.2.1 Automatic Detection of Newly Published Acts Containing Legislative Changes

One of the system's core capabilities is the continuous monitoring of newly published legal acts. Once access to the Sejm API is established and the monitoring process is activated, the system regularly scans incoming publications and reviews their metadata. When a newly released document contains legislative amendments, the system automatically identifies it as relevant and initiates the data-extraction workflow. This automated stage ensures that updates to the legal corpus are captured without requiring manual intervention and that all subsequent processing begins promptly.

Table 1: Comparison of the Proposed System with Existing Solutions

Solution	PDF extraction	Article-level detail	Interactive visualization	Open access
Sejm ELI API / ISAP [1, 3]	Metadata only	No	No	Yes
Legalis, Inforlex, Lex [6–8]	Partial (curated by publisher)	Yes	Limited	No (subscription)
Rule/template-based systems (UK, Japan, Italy) [9–12]	Yes	Yes	No	Varies
ML-based Polish amendment extraction [15, 16]	Yes	Partial	No	Research prototype
Legal knowledge graphs [17, 18]	No (structured input assumed)	Yes	Limited (graph query)	Research prototype
Graphie (UK) [22]	No	No	Yes	Research prototype
Proposed system	**Yes**	**Yes**	**Yes**	**Yes**

3.2.2 Extraction of Key Information from Legislative Documents

After a document containing amendments has been detected, the system proceeds to interpret its content. Using predefined templates and linguistic patterns adapted for legal texts, the system analyses the document and extracts essential information—such as the act number, publication date, affected articles, and the detailed wording of each amendment. Dates and identifiers are standardized, and the extracted elements are stored within structured data models representing acts, articles, and individual changes. This process transforms unstructured legal text into data suitable for further analysis and visualization.

3.2.3 Recording Changes in the Form of an Event Log

To support advanced process-analysis techniques, such as process mining, the system records each amendment and its context as an event. Whenever new or updated information is available, the system appends it to an existing event log or creates a new one if necessary. The log is then saved in XES format, a widely used standard for event-based analytical workflows. This representation allows researchers and analysts to trace the evolution of legislative processes over time and apply established analytical methods to study their structure.

3.2.4 Visualization of Relationships Between Legislative Changes

For legal analysts, understanding how different acts influence one another is essential. Using the processed data, the system generates a visual network of relationships between legal acts and the amendments they introduce. This graphical representation highlights how modifications propagate through the legal system and allows users to quickly grasp the dependencies linking various documents. By presenting these relationships in a clear and accessible manner, the visualization supports both high-level overview and detailed investigation.

3.2.5 Interactive Exploration of the Graph of Changes

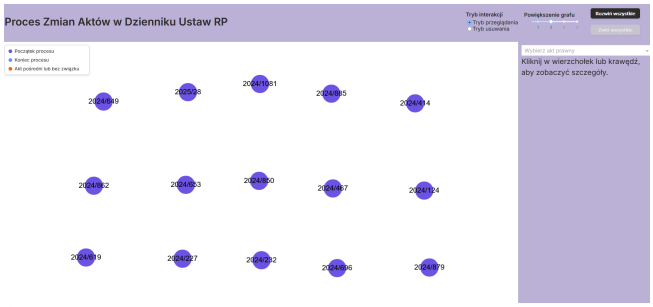
Beyond static visualization, the system offers an interactive interface that enables users to explore legislative relationships dynamically. Any user with access to the graphical interface can expand or collapse nodes, highlight a specific act, and navigate through the network at their own pace. This interactive mode is particularly valuable when working with large or complex datasets, as it allows users to tailor the level of detail to their needs and focus on the aspects most relevant to their analysis. Figure 3 illustrates this behavior: the initial view presents acts as collapsed, unconnected nodes, while after interaction the graph expands to reveal amendment relationships, and a specific act can be searched for and highlighted, with its details displayed alongside.

3.3. System Testing and User Evaluation

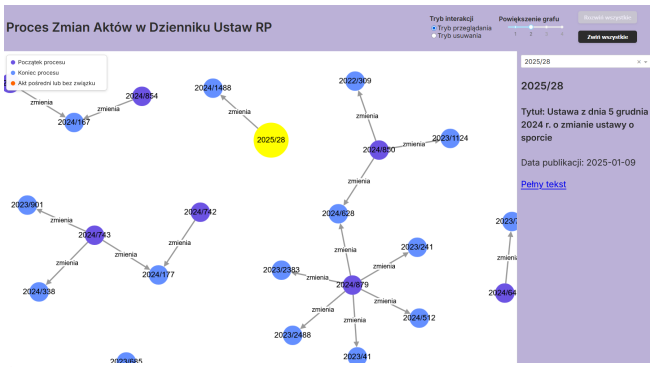
Testing the developed system constituted a crucial stage in assessing both the correctness of its operation and its practical usefulness from the perspective of end users. The evaluation process included two complementary components: technical verification through unit testing, and a preliminary usability pilot study involving real users.

To ensure the reliability of the system's core components, a suite of 51 automated unit tests was implemented using pytest, covering the three main processing modules: legislative text parsing (13 tests, including date and article-amendment parsing), metadata extraction (17 tests, including date-format conversion and act-identifier processing), and process mining (21 tests, covering XES event-log import and DFG discovery). Beyond well-formed inputs, the tests specifically targeted robustness under incomplete or atypical data, including malformed dates, missing DataFrame columns, duplicate events, invalid XML, and empty event logs. All 51 tests passed.

This stage therefore provides reproducible evidence that the implemented mechanisms for metadata extraction, event-log generation, and graph construction behave as expected under diverse conditions.



(a) Initial view: acts are shown as collapsed, unconnected nodes.



(b) After interaction: nodes are expanded to reveal amendment relationships, and the searched act (2025/28) is highlighted with its details displayed in the side panel.

Figure 3: Interactive graph traversal: from the collapsed initial state (a) to an expanded, explored view with a highlighted, searched act (b).

The second part of the evaluation focused on assessing the practical usability of the application. Given its small size (n=5) and the inclusion of participants without a professional legal background, this component of the evaluation is best characterized as a pilot study offering preliminary indications of usability rather than statistically robust evidence. A usability pilot was conducted with five participants of student or middle-aged background, two of whom regularly use legal information systems and represent the tool's intended target group. The goal of this pilot was to gather early feedback regarding the intuitiveness of the interface, the clarity of the visualisations, and the perceived usefulness of the system in analyzing legislative changes, and to inform the design of a larger, more representative evaluation in future work.

Participants were first provided with an instruction manual and allowed to explore the functionality of the application independently. Afterwards, they were asked to complete a questionnaire composed of two sections. Section A contained closed-ended questions assessed on a five-point Likert scale (1 – strongly disagree, 5 – strongly agree), while Section B consisted of open-ended questions designed to capture qualitative impressions.

3.3.1 Section A: Application Evaluation

Respondents assessed seven statements, each rated on a five-point Likert scale (1 – strongly disagree, 5 – strongly agree). Table 2 presents the statements together with the resulting scores. Given the small sample size (n=5), the median and range are reported rather than the mean, as these statistics better represent the tendency and variability in small samples.

Table 2: Median and Range of Responses Obtained in the Questionnaire (Part A, n=5)

Statement	Median	Range
A1. The interface is clear and intuitive.	4	1 (4–5)
A2. The application provides relevant information on relationships between legal acts.	5	1 (4–5)
A3. The visualization helps in understanding the changes.	5	1 (4–5)
A4. The ability to explore amended articles is useful.	4	2 (3–5)
A5. Using the application was not difficult.	4	2 (3–5)
A6. The presented results meet my expectations.	4	1 (4–5)
A7. The application may be useful in my work and/or interests.	3 / 4[1]	3 (2–5) / 2 (3–5)

Respondents rated the clarity and usefulness of the visualization (A3) and the relevance of information on dependencies between legal acts (A2) most highly, with a median of 5 and a narrow range of 1 (4–5), indicating consistent agreement across participants. The interface (A1) and alignment with expectations (A6) also received a median of 4 with the same range.

Somewhat wider variability was observed for the ease of exploring amended articles (A4) and the overall ease of use (A5), both with a median of 4 but a range of 2 (3–5), reflecting some divergence in participant experience. The lowest and most variable ratings concerned perceived applicability in daily work (A7): a median of 3 with a range of 3 (2–5) across all respondents, likely reflecting the inclusion of participants without regular exposure to legal-analysis tools. Among the two participants who do use such tools regularly, the median rose to 4 with a narrower range of 2 (3–5), suggesting stronger perceived value among the tool's intended target users.

Given the pilot nature and small size of the sample, these patterns should be read as preliminary indications rather than statistically robust or generalizable conclusions.

[1]The first value is for all respondents; the second considers only the two participants who regularly use legal information systems.

3.3.2 Section B: Open-Ended Responses

Participants also provided open-ended comments in response to four questions (B1: most-liked aspects, B2: difficulties encountered, B3: errors noticed, B4: suggested improvements). Rather than reporting responses per question, comments were inductively grouped into emergent thematic categories cutting across the original questions, summarized in Table 3.

Table 3: Thematic Summary of Open-Ended Responses (Part B)

Theme	Positive feedback	Suggested improvements
Search & information retrieval	Fast search (3); intuitive information search (2)	Difficulty searching without knowing the act title (1); requested search by act name or date (1) and centering the graph on the found element (1)
Interface & graph navigation	Clear, intuitive interface (3); dynamic graph manipulation (1)	Graph layout hard to follow (1); unclear how to restore hidden/collapsed nodes (1); requested keyboard-shortcut zooming (2)
Reliability & data accuracy	Correct functionality (1); no errors identified (5)	Requested improved typo-detection in extracted article text (2)
Data coverage & scope	–	Requested extension to additional legal sources (3)

The pilot feedback offers an encouraging, though preliminary, indication of the system's usability, particularly regarding the clarity of the interface and the speed and accuracy of search. Participants did not report any functional errors and several valued the tool's interactivity and transparency. At the same time, participants identified concrete areas for improvement across all four themes, including graph-navigation controls, search flexibility, and coverage of legal sources. Given the small and partly non-expert sample, these observations should be interpreted as directions for iterative refinement rather than a validated assessment of usability or effectiveness. They nonetheless point toward promising directions for future development, including extending data sources, refining text-processing mechanisms, and enhancing interactive functionality.

4. Conclusions and Future Work

The primary goal of this study was to develop a system capable of automatically extracting the evolution of legal acts published in the *Journal of Laws of the Republic of Poland*. Achieving this objective required an in-depth examination of the structure of legal documents available as PDF files through the governmental API, the identification of key components that constitute legislation—particularly amending acts—and the design of a system for exploring and visualizing these changes by means of a Directly-Follows Graph (DFG).

The resulting solution enables automated extraction of essential elements from both standard and amending acts, including titles, publication dates, detailed amendment provisions, and associated exceptions. This functionality was implemented using PyMuPDF for PDF processing and pyparsing for syntactic analysis. The extracted metadata is then transformed into an event-log representation, from which a DFG is derived using the pm4py process-mining library supplemented with custom algorithms. Finally, the system visualizes the graph in an interactive web application served locally, complete with a dedicated user interface.

The outcomes of the implementation, supported by usability testing conducted with a small group of end-users, confirm the practical utility of the proposed approach for analyzing legal documents and understanding relationships among legislative changes. The interactive visualization facilitates navigation through the continuously evolving legislative landscape and supports users who may not possess specialized legal expertise.

The developed tool offers a foundation for future expansion as a component of systems supporting legislators, legal analysts, or researchers tracking the evolution of statutory law. Further improvements may focus on enhancing the robustness of the metadata extraction module, particularly with regard to inconsistencies in source document formatting. Promising directions include incorporating machine-learning methods for classifying amendments and identifying types of legal acts, as well as applying natural-language processing techniques to detect potential typographical errors without manual intervention.

Additional development could extend the range of supported legal sources—for example, documents from the Official Gazette or local government regulations—or enable integration with existing legal information systems to provide verification and enrichment of extracted data. Enhancements to the user interface, such as search by act title or publication date, keyboard-shortcut support, or more advanced graph navigation features (e.g., automatic centering and zooming on selected acts), would further improve the overall usability and analytical value of the system.

Beyond these technical extensions, future work should prioritize a more rigorous empirical evaluation

of the tool. The usability pilot reported in this study involved a small sample that included participants without professional legal experience. Subsequent evaluations should recruit a substantially larger and more representative group of stakeholders, including practicing lawyer and legislative analysts to enable statistically robust and generalizable conclusions about the tool's usability, effectiveness, and practical applicability in real-world settings. Structured, task-based usability testing and longer-term deployment studies with the intended target groups would further strengthen the evidence base supporting the tool's practical value.

Acknowledgements

The research was supported [in part] by project "Cloud Artificial Intelligence Service Engineering (CAISE) platform to create universal and smart services for various application areas", No. KPOD.05.10-IW.10-0005/24, as part of the European IPCEI-CIS program, financed by NRRP (National Recovery and Resilience Plan) funds.

References

[1] Chancellery of the Sejm of Republic of Poland, "Eli api - sejm api 1.0 documentation." https://api.sejm.gov.pl/eli_pl.html#intro, 2025. Access: 15th October 2025.

[2] P. Ratyński, "Prawo - ministerstwo cyfryzacji (law - ministry of digital affairs)." https://www.gov.pl/web/cyfryzacja/prawo, 2025. Access: 15th October 2025.

[3] Chancellery of the Sejm of Republic of Poland, "Isap - internetowy system aktów prawnych (online system of legal acts)." https://isap.sejm.gov.pl/, 2025. Access: 15th October 2025.

[4] Senate of the Republic of Poland, "Principles of legislative drafting." https://www.senat.gov.pl/gfx/senat/userfiles/_public/k10/kancelaria/wydawnictwa/pdf/zasady_techniki.pdf, 2022. Access: 25th October 2025.

[5] "Obwieszczenie marszałka sejmu rzeczypospolitej polskiej z dnia 19 lipca 2019 r. w sprawie ogłoszenia jednolitego tekstu ustawy o ogłaszaniu aktów normatywnych i niektórych innych aktów prawnych." Dziennik Ustaw 2019, poz. 1461, 2019. Access: 2nd November 2025.

[6] C.H.Beck Publishing House, "Legalis legal information system." https://www.beck.pl/system-informacji-prawnej-legalis/, 2025. Access: 4th November 2025.

[7] Infor Pl S.A., "System inforlex home page." https://www.inforlex.pl, 2025. Access: 4th November 2025.

[8] Wolters Kluwer, "System lex – legal information database." https://www.wolterskluwer.com/pl-pl/solutions/lex/baza-informacji-prawnej, 2025. Access: 4th November 2025.

[9] T. Arnold-Moore, "Automatically processing amendments to legislation," in *Proceedings of the 5th International Conference on Artificial Intelligence and Law*, ICAIL '95, (New York, NY, USA), p. 297–306, Association for Computing Machinery, 1995.

[10] T. Arnold-Moore, "Automatic generation of amendment legislation," in *Proceedings of the 6th International Conference on Artificial Intelligence and Law*, ICAIL '97, (New York, NY, USA), p. 56–62, Association for Computing Machinery, 1997.

[11] Y. Ogawa, S. Inagaki, and K. Toyama, "Automatic consolidation of japanese statutes based on formalization of amendment sentences," in *New Frontiers in Artificial Intelligence* (K. Satoh, A. Inokuchi, K. Nagao, and T. Kawamura, eds.), (Berlin, Heidelberg), pp. 363–376, Springer Berlin Heidelberg, 2008.

[12] L. Robaldo, L. Lesmo, and D. P. Radicioni, "Compiling regular expressions to extract legal modifications," in *Proceedings of the The 25th International Conference on Legal Knowledge and Information Systems*, (Amsterdam, The Netherlands), IOS Press, 2012.

[13] G. Boella, L. Humphreys, M. Martin, P. Rossi, and L. van der Torre, "Eunomos, a legal document and knowledge management system to build legal services," in *AI Approaches to the Complexity of Legal Systems. Models and Ethical Challenges for Legal Systems, Legal Language and Legal Ontologies, Argumentation and Software Agents* (M. Palmirani, U. Pagallo, P. Casanovas, and G. Sartor, eds.), (Berlin, Heidelberg), pp. 131–146, Springer Berlin Heidelberg, 2012.

[14] J. Garofalakis, K. Plessas, and A. Plessas, "A semi-automatic system for the consolidation of greek legislative texts," in *Proceedings of the 20th Pan-Hellenic Conference on Informatics*, PCI '16, (New York, NY, USA), Association for Computing Machinery, 2016.

[15] L. Górski, "Towards legal change analysis: Clustering of polish civil code amendments," in *ASAIL@ICAIL*, 2019.

[16] A. Smywiński-Pohl, M. Piech, Z. Kaleta, and K. Wróbel, "Automatic extraction of amendments from polish statutory law," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*, ICAIL '21, (New York, NY, USA), p. 225–229, Association for Computing Machinery, 2021.

[17] V. W. Anelli, E. Brienza, M. Recupero, F. Greco, A. D. Maria, T. D. Noia, and E. D. Sciascio, "Navigating the legal landscape: Developing italy's official legal knowledge graph for enhanced legislative and public services," in *Ital-IA*, pp. 223–228, 2023.

[18] T.-H.-Y. Vuong, M.-Q. Hoang, T.-M. Nguyen, H.-T. Nguyen, and H.-T. Nguyen, "Constructing a knowledge graph for vietnamese legal cases with heterogeneous graphs," 2023.

[19] H. S. Abdellah, M. M'hamed, S. Faouzi, O. Aymen, H. Fedoua, A. Mohamed Chakib, and B. Takieddine, "Transforming legal documents into knowledge goldmines: Application on the algerian official journal," in *2024 IEEE 18th International Conference on Application of Information and Communication Technologies (AICT)*, pp. 1–7, 2024.

[20] H. Q. Ngo, H. D. Nguyen, and N.-A. Le-Khac, "Ontology knowledge map approach towards building linked data for vietnamese legal applications," *Vietnam Journal of Computer Science*, vol. 11, no. 02, pp. 323–342, 2024.

[21] D. Audrito, L. D. Caro, L. Genga, R. Mignone, R. Nai, I. Spada, E. Sulis, and V. M. S. Trifiletti, "Standardization and harmonization of law through automated process analysis and similarity techniques," in *Processes, Laws and Compliance 2024*, pp. 34–45, 2024.

[22] E. Tzanis, P. Vivo, Y. P. Förster, L. Gamberi, and A. Annibale, "Graphie: A network-based visual interface for the uk's primary legislation," *F1000Research*, vol. 12, p. 236, 2023.

[23] N-Lex, "N-lex – legislative websites of non-eu countries." https://n-lex.europa.eu/n-lex/related_links/related_links, 2025. Access: 14th November 2025.

[24] N-Lex, "N-lex – prawo w poszczególnych krajach ue." https://n-lex.europa.eu/index?lang=pl, 2025. Access: 14th November 2025.

[25] Artifex, "Pymupdf 1.25.5 documentation – features comparison." https://pymupdf.readthedocs.io/en/latest/about.html, 2025. Access: 8th November 2025.

[26] A. Przepiórkowski, E. Hajnicz, A. Patejuk, M. Woliński, F. Skwarski, and M. Świdziński, "Walenty: Towards a comprehensive valence dictionary of Polish," in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)* (N. Calzolari, K. Choukri, T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, eds.), (Reykjavik, Iceland), pp. 2785–2792, European Language Resources Association (ELRA), May 2014.

[27] M. Woliński, "Morfeusz — a practical tool for the morphological analysis of polish," in *Intelligent Information Processing and Web Mining* (M. A. Kłopotek, S. T. Wierzchoń, and K. Trojanowski, eds.), (Berlin, Heidelberg), pp. 511–520, Springer Berlin Heidelberg, 2006.

[28] Oracle, "Java se 8 documentation – class pattern." https://docs.oracle.com/javase/8/docs/api/java/util/regex/Pattern.html, 2025. Access: 7th November 2025.

[29] P. McGuire, "Pyparsing 3.2.0 documentation – using the pyparsing module." https://pyparsing-docs.readthedocs.io/en/latest/HowToUsePyparsing.html, 2025. Access: 8th November 2025.

[30] W.M.P. van der Aalst, *Process Mining: Data Science in Action*. Springer-Verlag, Berlin, 2016.

[31] Process Intelligence Solutions, "Pm4py: Getting started: Importing your first event log: File types: Csv and xes." https://processintelligence.solutions/static/api/2.7.11/getting_started.html#file-types-csv-and-xes, 2024. Access: 20th November 2025.

[32] IEEE – eric.verbeek, "About xes – standard extensions." https://www.tf-pm.org/resources/xes-standard/about-xes/standard-extensions, 2020. Access: 20th November 2025.

[33] W. M. P. van der Aalst, *Foundations of Process Discovery*, pp. 37–75. Cham: Springer International Publishing, 2022.

[34] Process and Data Science Group – RWTH Aachen University, "Overview / process discovery." https://www.processmining.org/process-discovery.html, 2025. Access: 22nd November 2025.