

Automatic AI-based Extraction of Processes from Administrative Texts: the Case of Waste Management

Alessia Membrini¹, Maria Teresa Rossi²[0000–0003–0273–7324], Leonardo Mariani¹[0000–0001–9527–7042], and Daniela Micucci¹[0000–0003–1261–2234]

¹ University of Milano-Bicocca, Milan, Italy
{alessia.membrini,leonardo.mariani,daniela.micucci}@unimib.it

² Gran Sasso Science Institute, L’Aquila, Italy
mariateresa.rossi@gssi.it

Abstract. Public administrations produce large volumes of unstructured administrative documents that regulate operational procedures, responsibilities, and regulatory constraints. These documents are characterized by high information density and fragmented procedural knowledge, making both human interpretation and automated extraction challenging. This issue is particularly critical in the waste management domain, where regulatory complexity and local variability significantly impact operational processes. This paper presents a hybrid pipeline for the automatic extraction and structuring of operational procedures from unstructured administrative texts. The proposed approach combines LLMs with RAG to ensure faithful, traceable extraction constrained to the source document. The pipeline identifies procedural sections, extracts relevant information from each section, and generates visual workflows. The system is evaluated on real-world administrative documents for waste management. Results show that the integration of generative AI within a structured pipeline improves the accuracy and reliability of procedural knowledge extraction, while mitigating hallucination risks.

Keywords: Procedural Knowledge Extraction · Automated Workflow Generation · Retrieval-Augmented Generation (RAG)

1 Introduction

Public administrations and private organizations continuously produce large volumes of administrative documents aimed at regulating operational activities, assigning responsibilities, and ensuring compliance with evolving regulatory frameworks [11]. Such documents are predominantly written in natural language and are characterized by high informational density, implicit structural hierarchies, and a mixture of normative, descriptive, and procedural content. In domains subject to complex and dynamic regulation, such as waste management, these documents play a crucial role in defining operational workflows and accountability chains. However, their discursive and often fragmented structure makes systematic interpretation and reuse challenging, particularly when procedures are

distributed across multiple sections and interwoven with references to definitions and legal provisions. The need to *bridge the gap between textual prescriptions and structured process representations* has thus become increasingly relevant [12].

The problem addressed in this work concerns the *automatic identification, extraction, and structuring of operational procedures from unstructured administrative documents* with a specific focus on the waste management domain. More precisely, the objective is to reconstruct coherent and verifiable procedural workflows from documents in which actions, responsible agents, temporal constraints, and conditional branches are not explicitly formalized but embedded in heterogeneous textual segments. This task requires not only detecting procedural sections but also resolving implicit references, integrating distributed definitions, and preserving links between extracted elements and their textual evidence.

The relevance of this problem is twofold. From an operational perspective, the manual interpretation of lengthy administrative documents is time-consuming and prone to inconsistencies, especially when multiple actors are involved and responsibilities depend on context-specific definitions. From a computational perspective, the extraction of structured processes from natural language represents a challenging instance of information extraction and semantic reconstruction. In regulatory domains, such as waste management, incorrect interpretation may lead to compliance issues or procedural delays. Therefore, any automated solution must ensure *fidelity* to the source text, *traceability* of extracted information, and *robustness* to linguistic variability and structural fragmentation.

State-of-the-art approaches can be broadly categorized into pattern-based methods [1], rule- or ontology-driven pipelines [6], and end-to-end Large Language Model (LLM) solutions [2]. Pattern-based approaches rely on recurring statistical regularities and are effective in homogeneous datasets, but they struggle with long, heterogeneous, and unique administrative documents. Rule-based and ontology-driven methods provide stronger formal guarantees but require substantial domain modeling and are less scalable when regulations vary across institutions. End-to-end LLM-based systems demonstrate strong linguistic generalization capabilities, but they are prone to hallucination and limited traceability, particularly when processing long documents. Retrieval-Augmented Generation (RAG) [8] mitigates some of these limitations by grounding generation in retrieved evidence; however, naive configurations may introduce irrelevant context and reduce interpretability.

Our approach advances knowledge in the area by investigating a combination of *structured document analysis*, grounded on the characteristics of waste management texts, and *LLM-based text interpretation*. Instead of relying exclusively on generative inference or rule-based heuristics, the proposed approach combines them: document hierarchy reconstruction and section segmentation are performed deterministically using structural heuristics, while LLMs are employed only in the interpretative tasks. Retrieval mechanisms are activated selectively to recover distributed definitions and contextual evidence, reducing the risk of unsupported inferences. LLMs are used to extract actions, agents, conditions,

and dependencies, producing a JSON representation that is transformed into graphical workflows.

We evaluated the approach on three real administrative documents from the waste management domain. We considered two alternative configurations of the proposed approach, one using a RAG populated with similar procedures and one not using a RAG, to assess the impact of the knowledge of similar procedures on the accuracy of the results. We also considered a pure LLM-based configuration as a baseline. The results indicate that the RAG-based configuration achieves a more balanced trade-off between precision and recall, particularly in the identification of procedural sections and responsible agents. The purely generative configuration tends to maximize recall at the cost of low precision, highlighting the importance of contextual constraints and structural grounding.

2 Processes in Waste Management Documents

Waste management represents a particularly challenging context for automatic process extraction due to the complexity and dynamic evolution of its regulatory framework. In fact, waste management procedures are highly context-dependent and often vary across regional and local administrations. As discussed in [5], procedural obligations may differ depending on territorial jurisdiction, organizational structure, and administrative competence. This variability reduces the feasibility of purely template-driven or rule-based solutions.

A procedure can be modeled as a structured entity composed of interdependent elements, including *actions*, *responsible agents*, *temporal constraints*, *conditional branches*, and possible *exceptions*. In unstructured administrative texts, however, this structure is rarely explicitly formalized and procedural information is rarely localized in a single coherent section. Instead, relevant content is unstructured, typically written in discursive natural language and characterized by: *cross-references* (sections that include references to concepts defined in other sections), *implicit dependencies* (dependencies between activities are often left implicit, and must be deduced from the text), *high informational density and length* (documents may span dozens of pages and contain numerous interdependent clauses), *mixed hierarchical organization* (titles and subtitles may exist but do not necessarily follow standardized patterns), *heterogeneous content types* (operational, normative, explanatory, and contextual content coexist within the same structural layer), *semi-formal linguistic style* (imperative, conditional, and narrative formulations are interwoven, increasing ambiguity), *anaphoric references and implicit agents* (entities are often defined once, in dedicated sections, and later recalled in subsequent sections).

In Figure 1, we report an excerpt of a typical procedure coming from an administrative text. The objective is to reconstruct complete procedural workflows, and not just to summarize textual content. This requires explicitly identifying actions, agents, responsibilities, and constraints, resolving ambiguities, and preserving a verifiable link to the originating textual evidence.

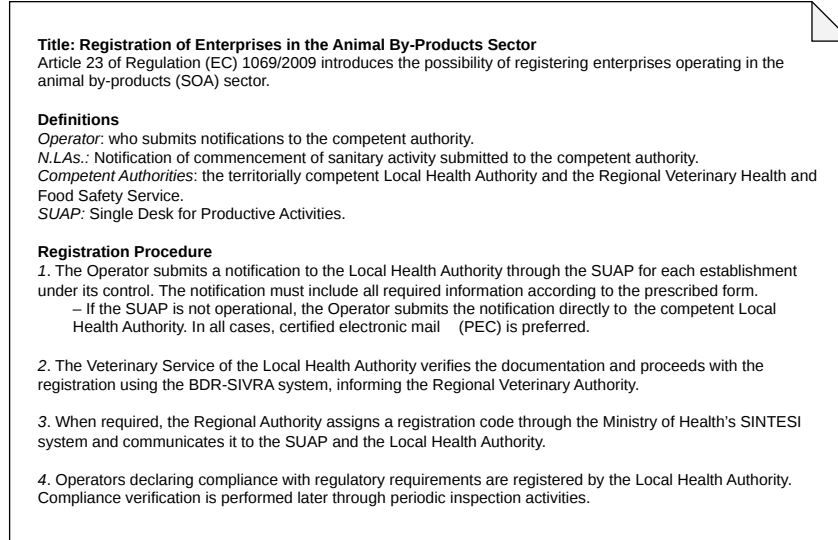


Fig. 1. Excerpt of a typical procedure coming from an administrative text

Due to the structural dispersion of the text (e.g., in Figure 1 the SUAP (Sportello Unico per le Attività Produttive) is described in the *Definitions* section and the corresponding action appears in a separate section), procedural extraction cannot rely on local sentence-level analysis alone, but requires context-aware integration across structural boundaries. The heterogeneity of expression further increases complexity. Actions may be formulated narratively; agents may be expressed through pronouns or generic labels; conditions may be introduced through subordinate clauses; temporal ordering may be implied rather than explicitly stated. For instance, in the example in Figure 1 the first step of the *Registration procedure* contains a conditional clause indented differently in the text. The challenge is therefore not limited to determining *where* procedural information appears, but extends to understanding *how* it is semantically expressed and how it relates to other elements.

3 Related Work

The problem of extracting structured process representations administrative documents has been addressed with pattern-based approaches, rule-based approaches, and LLM-based approaches.

Pattern-Based Approaches Early attempts to extract process models from text adopt bottom-up strategies based on pattern discovery and sequence mining.

These approaches assume that procedural knowledge can be inferred from recurring linguistic patterns or statistically frequent sequences observed across document collections. For instance, techniques inspired by process mining [1] have been applied to textual artifacts such as email logs or structured operational reports, where textual segments are treated as event traces. Systems such as MailOfMine [4] reconstruct workflows from email corpora by first classifying messages and then applying process discovery algorithms. Similarly, Shing *et al.* [10] combine topic modeling and sequence mining to identify activity clusters and reconstruct procedural dependencies. While these methods demonstrate that latent process structures can emerge from corpora, they rely on the availability of multiple similar instances and recurring patterns. In the context of administrative regulations, where each document may describe unique procedures and where textual regularity is limited, such corpus-driven assumptions are difficult to satisfy. Moreover, these approaches do not explicitly address traceability to source text segments, a requirement central to regulatory domains.

Rule-Based Methods A second line of research adopts top-down approaches based on syntactic parsing, handcrafted rules, and domain ontologies. In these systems, linguistic structures such as Subject–Verb–Object triples are extracted and mapped to predefined process constructs. For instance, Friedrich *et al.* [6] propose a pipeline for transforming natural language text into BPMN models using dependency parsing and rule-based transformation. Their method demonstrates that structured process models can be derived from textual descriptions when the input adheres to relatively controlled syntactic patterns. Similar approaches leverage semantic role labeling [3] and domain ontologies [13] to improve action–agent association. Although rule-based methods provide strong formal control, they require extensive engineering effort and domain adaptation. In heterogeneous administrative contexts characterized by evolving regulations and linguistic variability, maintaining rule sets and ontological mappings becomes increasingly costly. Furthermore, these approaches struggle with distributed definitions and long-range dependencies typical of regulatory texts.

LLM-Based Approaches The recent emergence of LLMs has significantly influenced document understanding tasks. In the context of process modeling, Ajmal *et al.* [2] propose an automated BPMN diagram generation framework in which a fine-tuned generative model directly extracts structured process representations from text. This end-to-end paradigm reduces the need for handcrafted rules and improves robustness to linguistic variation. However, purely generative approaches present well-documented limitations, including hallucination phenomena and limited traceability [7]. In regulatory domains, where faithfulness to the source document is mandatory, unconstrained generation poses reliability risks. Moreover, the fixed context window of LLMs complicates the processing of long administrative documents with hierarchical structures. To mitigate the limitations of purely generative models, hybrid architectures combining retrieval and generation have been proposed. RAG [8] has been successfully applied in question answering and knowledge-intensive NLP tasks, where grounding reduces halluci-

nation and improves factual consistency. However, its application to procedural extraction from administrative documents remains underexplored. In particular, naive RAG configurations may retrieve semantically related but structurally irrelevant fragments, potentially contaminating the generative phase. Recent research emphasizes the importance of structured context selection for reliable information extraction [9]. This work shows that combining semantic retrieval with structural constraints can enhance interpretability and robustness.

4 Proposed Approach

The architecture of the proposed pipeline separates structural analysis from semantic interpretation tasks delegated to LLMs, and integrates RAG only where context is required. This design choice ensures traceability and robustness to the linguistic and structural variability typical of administrative texts.

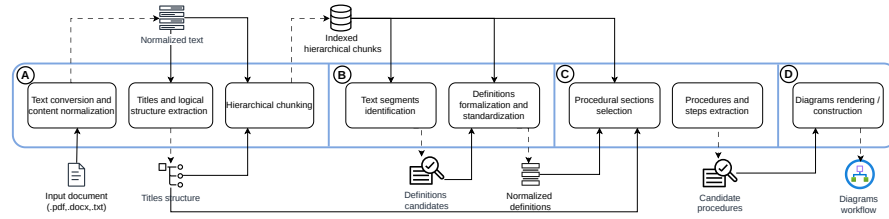


Fig. 2. Automatic process extraction.

Figure 2 illustrates the architecture of the pipeline composed of four main phases: **(A) Knowledge Base Creation**, **(B) Definitions Detection**, **(C) Procedures Detection**, and **(D) Diagrams Generation**. The input is a waste management document in unstructured textual form and the output is a diagrammatic representation of the extracted procedures.

At a glance, the workflow operates as follows. In **(A) Knowledge Base Creation**, the document is normalized by removing consecutive blank lines, repeated headers and page numbers, and segmented into hierarchically organized chunks enriched with metadata. This phase produces an indexed internal representation that serves as a persistent and interrogable knowledge base. In **(B) Definitions Detection**, the definitional segments are identified, normalized, and stored to build a terminological reference (e.g., "Operator: the person who submits notifications to the competent authority."). In **(C) Procedures Detection**, candidate procedural sections are analyzed using LLMs, supported by retrieval over the knowledge base if necessary. The output is a structured JSON representation of actions, agents, conditions, and dependencies that shall appear in the extracted workflow. Finally, in **(D) Diagrams Generation**, the structured representation is transformed into graphical workflows using dedicated tools. In the following, we

illustrate these steps in detail using the fragment of an administrative regulation in Figure 1 as a running example.

4.1 Knowledge Base Creation

The first phase of the approach is devoted to constructing a structured internal representation of the input document. The goal is to transform a heterogeneous text into a hierarchically organized and traceable text where titles and text between titles are isolated, to support subsequent retrieval and interpretation.

The process begins with *document acquisition and normalization*, which involves the removal of duplicated headers, page numbers, and redundant blank lines. This normalization reduces structural noise and prevents artificial fragmentation during segmentation.

Next, the system performs *structural analysis and heading detection*, which identifies titles and section headers using structural heuristics based on the following recurring patterns observed in administrative documents: (i) numbering patterns (e.g., “Art. 5”, “1.”, “1.1”), (ii) typographic conventions such as capitalization and short standalone lines, and (iii) domain-specific lexical markers (e.g., “Definitions”, “Procedure”, “Title”). These heuristics are used to extract a set of candidate headings, which are passed to an LLM that identifies false positives. This design reflects the observation that heading identification is primarily driven by syntactic and typographic features, while semantic interpretation is only required to disambiguate borderline cases.

In the running example in Figure 1, the document contains section headers such as TITLE: REGISTRATION OF ENTERPRISES IN THE ANIMAL BY-PRODUCTS SECTOR. This heading is detected through a combination of structural heuristics, including its standalone position, typographic prominence (e.g., capitalization and absence of surrounding text), and lexical cues typical of section headers such as the marker “Title”. The candidate is then validated through an LLM-based classification step to confirm its role as a structural heading.

Once the hierarchical structure is reconstructed, the text between titles/headers is segmented into *hierarchical chunks*. Each chunk corresponds to a semantically coherent portion of text within a specific section and is enriched with metadata, including: (i) section identifier, (ii) position within the section, and (iii) document identifier. This segmentation strategy produces chunks of variable length, as they are defined by the document structure rather than by fixed-size windows. Each chunk preserves semantic coherence, since it corresponds to a meaningful unit of text (e.g., a section or sub-section). In the example, the block describing the notification submission by the Operator and the subsequent verification by the Local Health Authority is stored as a chunk associated with the section “Registration of Enterprises in the Animal By-Products Sector”. This chunk becomes an indexed unit in the internal knowledge base and can later be retrieved either semantically (via embeddings) or structurally (via metadata filters constrained to the parent section).

4.2 Definitions Detection

Administrative documents frequently introduce definitions that specify the meaning of roles, entities, or procedures. These definitions are often distributed across different sections and may not be globally applicable. The *Definitions Detection* phase extracts, normalizes, and organizes them before procedural interpretation.

The detection process starts from the indexed chunks produced in the previous phase. Candidate definitional segments are identified using pattern-based heuristics, derived from recurring linguistic formulations in administrative texts. In particular, the approach targets structures such as term-definition patterns (e.g., “X: ...”) and explicit definitional expressions (e.g., “For X it is meant ...” or “For the purposes of this regulation, X means ...”). These patterns are used to identify candidate segments likely to introduce definitions. In addition, structural cues are considered, such as sections explicitly labeled as “*Definitions*”, which are treated as high-confidence sources of definitional content.

In the running example, suppose the document contains the statement “*Operator who submits notifications to the competent authority.*”. This segment is detected as definitional and extracted together with its structural metadata. Since definitions may vary slightly in formulation across sections, a normalization step is performed. Every time a chunk potentially including a definition is detected, an LLM is prompted to (i) isolate the defined term, (ii) extract its corresponding description through minimal paraphrasing for clarity, (iii) avoid introducing new concepts or external knowledge not present in the text, and (iv) produce a structured output in JSON.

The links between sections and the included definitions are stored for reference. This reference is later used to recognize agents in the procedural steps. For example, in the *Registration Procedure* section of the running example, a sentence such as “*The Operator submits the notification to the ASL through the SUAP.*” can be interpreted by linking the term “Operator” to its definition, thereby avoiding any generic or ambiguous interpretation of the terms, ensuring coherence across sections.

4.3 Procedures Detection

The *Procedures Detection* phase constitutes the core interpretative component of the pipeline. Its objective is to identify procedures represented as actions performed by agents under specific constraints, also considering control flows information (the links between actions).

The process begins with a *deterministic pre-filtering* of sections. Rather than performing an inefficient global semantic search for actions, the system analyzes each section returned by the previous step according to terms that mark the possible presence of procedures. These include deontic expressions (e.g., “must”, “is responsible for”), temporal markers (e.g., “within”, “subsequently”), enumerations, and action verbs. For each fragment that is marked as possibly describing procedures, the approach interprets the text semantically by prompting an LLM to use the knowledge base of the definitions extracted so far for returning a JSON

structured representation of the agent doing the action, the recipient of any data produced by the action, the temporal constraints and dependencies from other actions, if any.

Consider the following fragment from the running example: *“The Operator submits the notification to the Local Health Authority through the SUAP. The Veterinary Service of the Local Health Authority verifies the documentation and proceeds with the registration.”* From this fragment, the LLM generates a structured representation including two actions. The first action is **submission of notification**, with **Operator** as agent and **Local Health Authority (via SUAP)** as recipient. The second action is **verification and registration**, with **Veterinary Service of the Local Health Authority** as agent. The model also identifies a sequential dependency, namely that the verification activity occurs after the submission of the notification. If temporal expressions such as “within 7 days” are present, they are extracted as temporal constraints associated with the relevant action.

4.4 Diagrams Generation

The final phase transforms the structured JSON representation into graphical process models. The objective is to produce visual workflows that facilitate human interpretation while preserving alignment with the extracted structure. The transformation process adjusts the data stored in the JSON format according to the target visualization library. For instance, our prototype implementation supports both Graphviz³, for simple free-form graph-like representations, and ProcessPiper⁴ for the generation of BPMN-compliant diagrams.

In particular, the BPMN representation is obtained by applying the following rules: (a) Each action is converted into a BPMN activity node; (b) The agent is mapped into lanes, so that the responsibilities of the actions are visually attributed; and (c) The control-flow relations are translated into directed connections between activities.

5 Empirical Assessment

Our evaluation is driven by the following three research questions:

RQ1-Structure: *How accurate is the reconstruction of the hierarchical document schema?* This question focuses on the ability of the system to correctly identify titles, subtitles, and the hierarchical organization of the document, which constitutes the structural backbone for subsequent extraction steps.

RQ2-Agents: *How accurate is the identification and canonicalization of agents?* This question investigates how accurate the extraction and normalization of the agents is.

RQ3-Procedures: *How accurate is the identification of the procedures?* This question investigates the effectiveness of the proposed pipeline in detecting procedural sections and extracting structured representations of procedures.

³ <https://graphviz.org/>

⁴ <https://github.com/csgoh/processpiper>

5.1 Experiment Setup

The experimental evaluation was conducted on a corpus of three real administrative documents belonging to the waste management domain coming from different municipalities and regulating the waste disposal for public and private entities. These documents were selected because of their high regulatory complexity, heterogeneous structure, and strong contextual dependency between procedural sections and distributed definitions⁵. The documents include regional and local regulations, operational guidelines, and administrative provisions describing notification procedures, authorization workflows, and verification processes. In several cases, definitions of roles such as “Operator”, “Portal”, or “Local Health Authority” are introduced in dedicated sections and later referenced anaphorically in procedural paragraphs, thus providing a realistic setting to test both structural reconstruction and semantic extraction.

For each document, we manually constructed a ground truth, where we identified titles and subtitles, marked procedural paragraphs, and explicitly specified the extracted procedures and associated agents. The first document is 16 pages long and includes 31 agent definitions and 9 procedures. The second document is 1 page and includes 10 agent definitions and 1 procedure. The third document is 10 pages long and includes 14 agent definitions and 13 procedures.

We used the three documents and the associated ground truth to address RQ1–RQ3. We studied three configurations: *pipeline-w/o RAG*, which corresponds to the structured pipeline described in this paper without activating retrieval; *pipeline with RAG*, which uses the same pipeline but integrates a RAG populated with the hierarchical chunks extracted from the input document (sections, paragraphs, and definitional segments enriched with structural metadata), and *LLM-only*, which serves as a baseline approach relying exclusively on an LLM prompted to extract procedures without structural constraints or retrieval.

In the RAG configuration, retrieval is performed over the indexed chunks of the same document and is used to help the LLM in identifying candidate procedural sections and retrieving relevant definitions. In all experiments, we used the same GPT-based⁶ Large Language Model deployed through a cloud infrastructure, ensuring that differences across configurations depend on architectural choices rather than model capacity.

To answer the research questions, we compared the data extracted by the automatic procedures to the ground truth we created, computing precision, recall, and F1-score for the identification of titles and subtitles (RQ1), agents (RQ2), and procedures (RQ3).

For RQ1 and RQ2, retrieved items and ground truth elements are checked using exact matching. In the case of agents, surface variations (e.g., abbreviated forms or minor lexical differences) are tolerated only when they clearly refer to the same organizational role.

⁵ we cannot provide the link in the submitted version of the paper, since this would reveal information about the geographical origins of the authors. We will provide actual references to the documents in the final version of the paper, if accepted.

⁶ <https://openai.com/it-IT/index/gpt-4/>

For RQ3, a predicted procedure is considered correct if it corresponds to a gold-standard procedure describing the same operational sequence within the same document section. Minor lexical variations are tolerated, provided that (i) the actions exactly match those annotated in the ground truth, (ii) the responsible agents match the annotated ones, and (iii) the ordering relations between actions are preserved. A predicted procedure that partially overlaps with a gold procedure but omits essential actions or introduces additional unrelated actions is considered an incorrect match. Each gold-standard procedure can be matched to at most one predicted procedure.

The experimental material to reproduce our results is available online⁷.

5.2 RQ1 - Structure

RQ1 examines how accurately the structural elements (i.e., titles and subtitles) are identified in real administrative texts, so that the document can be hierarchically partitioned into multiple text segments. In the proposed pipeline, heading detection is based on typographic and syntactic heuristics and LLM is used as a validation step to confirm or reject the candidate headings. In the LLM-only configuration, by contrast, the model is directly prompted to identify titles without the support of structural constraints. Table 1 column RQ1-Structure reports the accuracy of the three considered configurations.

Pipeline	RQ1-Structure			RQ2-Agents			RQ3-Procedures		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
pipeline with RAG	0.89	0.76	0.82	0.69	0.30	0.40	0.68	0.60	0.63
pipeline w/o RAG	0.88	0.83	0.86	0.67	0.21	0.30	0.67	0.68	0.67
LLM-Only (baseline)	0.44	0.98	0.60	0.63	0.29	0.39	0.32	0.76	0.44

Table 1. Results about the extraction of the structure.

The results highlight markedly different behaviors across configurations. The LLM-only approach achieves the highest recall (0.9833), indicating that it tends to identify almost all true headings. However, its precision is significantly lower (0.4435), meaning that it frequently misclassifies non-structural text fragments as titles. In practice, the model confuses descriptive paragraphs or emphasized sentences with actual section headers. This behavior results in a large number of false positives and consequently a low F1 score.

The pipeline without RAG achieves the best F1 score (0.8561), representing the most balanced trade-off between precision and recall. The deterministic structural heuristics effectively limit the search space to plausible headings, while the LLM validation step reduces residual ambiguities. Errors in this configuration mainly occur in cases where headings deviate from expected formatting patterns or when numbering conventions are inconsistent.

⁷ <https://github.com/AlessiaMembrini/DocuProcessAI>

The pipeline with RAG shows slightly lower recall compared to the non-RAG configuration. This suggests that, in the context of structural detection, retrieval may introduce additional noise by supplying semantically related but structurally irrelevant fragments. Since heading identification is primarily a syntactic problem rather than a semantic one, constraining the model with semantically retrieved context does not necessarily improve performance and may occasionally mislead the validation phase.

Overall, these findings indicate that the combination of rule-based candidate generation and controlled LLM validation provides a robust compromise between completeness and precision in identifying document hierarchy.

5.3 RQ2 - Agents

RQ2 investigates the identification of the agents involved in procedures, which is a task sensitive to distributed definitions, anaphoric references, and variability in role naming. The results are reported in Tables 1 column RQ2 - Agents.

The results confirm that agent identification is the most challenging aspect of the extraction process. Recall values are consistently low across configurations, indicating that many agents are not explicitly detected. This difficulty stems from the characteristics of administrative texts: agents are frequently expressed implicitly, introduced once in a definitional section and later referred to through anaphoric expressions (e.g., “the operator”, “the competent authority”), or embedded within complex syntactic constructions. As a result, purely local sentence-level interpretation often fails to recover all responsible roles.

Despite the low recall, precision remains comparatively higher across all configurations. This indicates that when the system identifies an agent, it is often correct. In other words, the approach tends to be conservative: it avoids over-generating agents, but struggles to capture all implicit references.

The pipeline with RAG achieves the highest precision, recall, and F1 score among the three configurations. Although the improvement in recall is moderate, it is consistent. This suggests that retrieval plays a beneficial role in agent identification. In particular, the RAG component enables the LLM to access definitional segments and structurally related fragments within the same document, supporting the disambiguation and canonicalization of roles. When an agent is mentioned indirectly in a procedural section, retrieval of the corresponding definition helps the model associate the expression with the correct role.

The pipeline without RAG shows lower recall, reflecting the difficulty of resolving distributed definitions without explicit contextual retrieval. The LLM-only configuration achieves a recall comparable to the RAG setting but with slightly lower precision, indicating a tendency to occasionally infer agents without sufficient grounding. These results suggest that the current RAG configuration, which is populated with chunks extracted from the same input document, is only partially effective for agent identification. A possible limitation is that the retrieved context does not always include sufficiently explicit organizational knowledge to resolve implicit or anaphoric references to responsible

actors. Improving the content used to populate the retrieval component represents a promising direction for increasing recall without sacrificing precision.

5.4 RQ3 - Procedures

The objective of RQ3 is to assess the accuracy of the identification and extraction of procedural content. Results are shown in Table 1 Column RQ3-Procedures.

The LLM-only configuration achieves the highest recall (0.7578), indicating that it tends to identify a large portion of the procedures present in the documents. However, its precision is substantially low (0.3241), meaning that it frequently extracts spurious or over-segmented procedural fragments. In practice, the model often interprets descriptive or normative passages as independent procedures, resulting in a high number of false positives and consequently a low F1 score.

The pipeline without RAG achieves the highest F1 score (0.6686), representing the most balanced trade-off between completeness and correctness. The deterministic pre-filtering of candidate sections effectively reduces the search space, limiting the extraction to structurally plausible procedural segments. The subsequent LLM interpretation operates within these constrained contexts, reducing over-generation while preserving most of the relevant procedures.

The pipeline with RAG does not outperform the non-RAG configuration in this setting. Although retrieval is intended to provide additional contextual grounding, in the case of procedure detection it may introduce semantically related but procedurally irrelevant fragments from nearby sections. This additional context can blur section boundaries and lead to under-segmentation or misalignment with the ground truth, slightly reducing recall and overall F1.

5.5 Qualitative Analysis of the Generated Diagrams

To complement the quantitative evaluation, we provide a qualitative comparison of the generated diagrams. Figures 3 and 4 illustrate two complementary views of the extracted procedure.

Figure 3 shows the complete multi-boxes diagram generated from the structured JSON representation. The figure corresponds to the “Business Registration” process and displays the activities distributed across organizational actors such as the *Business Owner – Animal By-Products Sector*, the *Local Health Authority*, the *Veterinary Service*, and the *Regional Veterinary and Food Safety Service*.

The diagram can be read starting from the green “Beginning of the procedure” node and ending at the red “Conclusion of the business registration procedure” node. Each box represents a distinct organizational role, and the activities inside each box correspond to the actions extracted for that specific agent. The decision node “Check whether the portal is active” illustrates how conditional logic identified in the text is translated into an explicit branching structure.

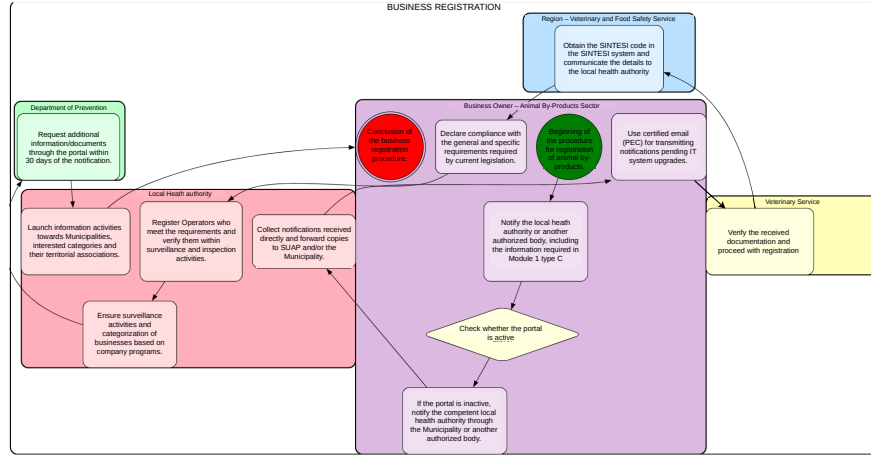


Fig. 3. Multi-lane diagram of the business registration procedure generated from the extracted structured representation.

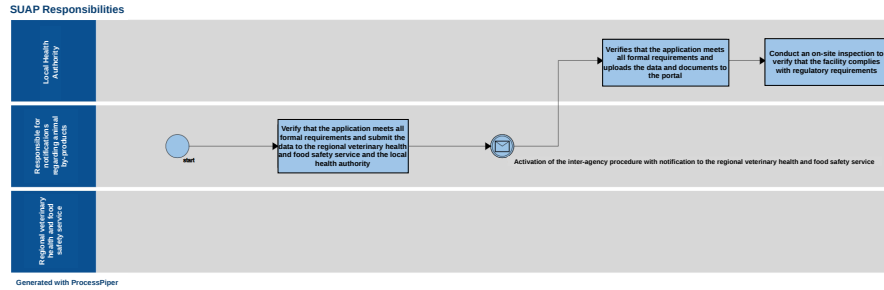


Fig. 4. Excerpt of the obtained BPMN representation of SUAP responsibilities.

Compared to the original textual description, the diagram makes the distribution of responsibilities immediately visible. For instance, the distinction between actions performed by the *Business Owner* (e.g., submitting notifications), those performed by the *Veterinary Service* (verification and registration), and those performed by the *Regional Authority* (SINTESI code assignment) becomes structurally explicit rather than being dispersed across paragraphs.

Figure 4 shows an excerpt of a BPMN-compliant representation generated with ProcessPiper. This diagram focuses on the responsibilities of the SUAP and related authorities. In this BPMN view, activities are placed within lanes corresponding to institutional roles (e.g., Local Health Authority, Responsible for animal by-products registration, Regional veterinary health and food safety service). The start event, task nodes, and sequence flows provide a standardized representation of the procedural flow.

Beyond improved clarity, the graphical representation offers analytical advantages over plain text. It enables immediate inspection of (i) responsibility distribution across actors, (ii) control-flow structure (sequence and branching), (iii) potential bottlenecks or decision points, and (iv) missing or ambiguous agents.

While textual descriptions require sequential reading and mental reconstruction of dependencies, the diagrammatic representation externalizes procedural logic into an explicit visual structure. This makes the extracted knowledge more accessible for validation, discussion with domain experts, and potential reuse in compliance or digital transformation scenarios.

5.6 Discussion

The empirical results indicate that the proposed pipeline should be considered a decision-support tool rather than a fully autonomous replacement for human process modeling. The scores obtained for structure and procedures suggest that the approach can produce useful initial process representations, especially compared to the LLM-only baseline. However, the lower performance on agent identification shows that human validation remains necessary in compliance-sensitive settings. In practice, experts have to validate agents, correct missing or implicit responsibilities, and refine procedural boundaries before operational use.

The pipeline can support decision makers by transforming dense administrative regulations into explicit process-oriented representations, making responsibilities, dependencies, ambiguities, and bottlenecks easier to inspect. Some aspects are domain-specific, such as terminology, procedural cues, and role definitions related to waste management, while the overall phases of the pipeline are reusable across administrative domains. The results also suggest that RAG should be applied selectively: it is useful for tasks requiring distributed contextual knowledge, such as resolving implicit agents or role definitions, but may introduce noise in tasks mainly driven by local structural evidence, such as heading detection. Therefore, administrations adopting similar pipelines should carefully design the knowledge base and retrieval configuration, possibly enriching them with stakeholder lists, organizational structures, and role descriptions.

These findings align with prior work on process extraction and document understanding. Pure LLM-based approaches provide flexibility, but our results confirm their risk of over-generation and limited grounding, as shown by the low precision of the LLM-only baseline for procedures. The proposed hybrid pipeline addresses these trade-offs through structural constraints, controlled LLM interpretation, and selective retrieval.

5.7 Threats to Validity

This empirical assessment is subject to several threats to validity.

Internal validity. A first threat concerns the construction of the ground truth. Titles, procedures, and agents were manually annotated, and although the annotation process followed consistent guidelines, the identification of procedural

boundaries and the canonicalization of agents may still involve subjective judgment, especially when procedures are fragmented across sections or when agents are implicit. This threat was mitigated by adopting one-to-one matching and by evaluating procedures through structural criteria (actions, agents, and ordering) rather than relying on purely textual similarity. This aspect can be strengthened in future studies by involving multiple annotators or domain experts and assessing inter-annotator agreement.

Construct validity. The evaluation relies on precision, recall, and F1-score computed through exact matching for structure and agents, and through a structural one-to-one correspondence for procedures. These metrics capture correctness and completeness of extraction, but they do not directly measure the usefulness of the extracted artifacts for downstream tasks nor the degree of interpretability of the produced diagrams. In addition, while the procedure matching tolerates minor lexical variation, it still requires alignment on core actions, agents, and ordering, potentially penalizing alternative but equivalent decompositions of the same procedure.

External validity. The dataset consists of three administrative documents from the waste management domain. Although the documents originate from different municipalities and exhibit heterogeneous structures, the sample size is limited, and the domain is specific. As a consequence, generalization to other administrative domains, document styles, or jurisdictions cannot be claimed without further validation. Yet, our empirical evidence is based on real-world documents. The evaluation relies on a single GPT-based model, compares against an internal LLM-only baseline, and provides an illustrative rather than standardized qualitative assessment. Future work should address these limitations, extending the empirical evidence to additional datasets, baselines, and models.

Conclusion validity. Results may be sensitive to configuration choices and prompt formulation. To reduce this threat, all configurations used the same GPT-based model and were evaluated on the same annotated corpus, but future studies should replicate the evaluation under different model versions and parameter settings.

6 Conclusion and Future Work

This work addresses the problem of extracting structured and verifiable procedural representations from unstructured administrative documents, specifically considering the case of waste management documents. The proposed approach combines structural analysis of the documents with controlled use of LLMs, and selective use of the RAG to obtain a pipeline that can provide a visual illustration of articulated procedures.

The evaluation demonstrates the effectiveness of the approach, while it shows challenges in extracting agents and fully accurate procedural descriptions.

Future work mainly concern with the definition of strategies to improve the accuracy of agent identification. Further, considering additional documents in other areas of public administration is another relevant research direction.

References

1. Van der Aalst, W.: *Process Mining: Data Science in Action*. Springer Publishing Company, Incorporated, 2nd edn. (2016)
2. Ajmal, F., Wijekoon, P., Dhanamina, H., Ravishan, Y., Nawinna, D., Atanayaka, B.: Automated bpmn diagram generation. In: 2024 6th International Conference on Advancements in Computing (ICAC). pp. 7–12 (2024). <https://doi.org/10.1109/ICAC64487.2024.10851120>
3. Cui, M., Huang, R., Hu, Z., Xia, F., Xu, X., Qi, L.: Semantic rule-based information extraction for meteorological reports. *International Journal of Machine Learning and Cybernetics* **15**(1), 177–188 (2024)
4. Di Ciccio, C., Mecella, M., Scannapieco, M., Zardetto, D., Catarci, T.: Mailofmine – analyzing mail messages for mining artful collaborative processes. In: *Business Process Management Workshops (BPM 2011 Workshops)*. Lecture Notes in Business Information Processing, vol. 116, pp. 55–81. Springer (2012)
5. Di Salle, A., Fedeli, A., Iovino, L., Mariani, L., Micucci, D., Rebelo, L., Rossi, M.T.: Waste management through digital twins and business process modeling. In: *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems Companion*. pp. 1–6 (2024)
6. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: *Advanced Information Systems Engineering (CAiSE 2011)*. Lecture Notes in Computer Science, vol. 6741, pp. 482–496. Springer (2011)
7. Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y.J., Madotto, A., Fung, P.: Survey of hallucination in natural language generation. *ACM Comput. Surv.* **55**(12) (2023)
8. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.t., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-augmented generation for knowledge-intensive nlp tasks. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) *Advances in Neural Information Processing Systems*. vol. 33, pp. 9459–9474. Curran Associates, Inc. (2020)
9. Perot, V., Kang, K., Luisier, F., Su, G., Sun, X., Boppana, R.S., Wang, Z., Wang, Z., Mu, J., Zhang, H., Lee, C., Hua, N.: LMDX: Language model-based document information extraction and localization. In: *Findings of the Association for Computational Linguistics: ACL 2024* (2024)
10. Shing, L., Wollaber, A., Chikkagoudar, S., Yuen, J., Alvino, P., Chambers, A., Allard, T.: Extracting workflows from natural language documents: A first step. In: *Business Process Management Workshops*. pp. 294–300. Springer International Publishing (2019)
11. Soliman, K.S., Affisco, J.F., Klischewski, R.: Ontologies for e-document management in public administration. *Business Process Management Journal* **12**(1), 34–47 (2006). <https://doi.org/10.1108/14637150610643742>
12. Vesperi, W., Melina, A.M., Ventura, M.: Organizing decision making processes in public administration: The impact of knowledge visualization. In: *ECKM 2023 24th European Conference on Knowledge Management Vol 2*. Academic Conferences and publishing limited (2023)
13. Zhong, B., Wu, H., Xiang, R., Guo, J.: Automatic information extraction from construction quality inspection regulations: A knowledge pattern-based ontological method. *Journal of Construction Engineering and Management* **148**(3), 04021207 (2022)