Journal of Visual Language and Computing

journal homepage: www.ksiresearch.org/jvlc

Designing an Efficient Document Management System (DMS) using Ontology and SHACL Shapes[⋆,⋆⋆]

Maria Assunta Cappelli[a,*], Ashley Caselli[a,*] and Giovanna Di Marzo Serugendo[a,*]

[a]Centre Universitaire d'Informatique Université de Genève–CUI, Geneva, Switzerland

ARTICLE INFO

Article History:
Submitted 5.10.2023
Revised 7.31.2023
Accepted 8.2.2023

Keywords:
Automatic Document Processing
Data Management Systems
Knowledge Graph-based Approach
Reasoning Engine

ABSTRACT

Document management systems (DMS) are widely used for the management of business documents because they use metadata to organise and categorise digital documents. However, they are often based on unstructured and monolithic files and this structure raises questions about the quality and completeness of the information. To overcome this problem, this paper proposes a semantic rule-based approach for an RDF-based DMS, which uses a combination of ontology and Shapes Constraint Language (SHACL) rules to integrate legal aspects, validate data (expressed in an RDF triple store), reason and infer new information. The process is dynamic because the proposed DMS can automatically reason and create new inferences based on the information and data extracted from documents. The process also reasons on user profiles and underlying rules, capturing specific legal regulations, enabling further accurate and automated document management. The ontology used in the process captures specific concepts of Swiss tax returns, while the SHACL rules serve to reason about actual RDF triples relating to different tax households. The proposed DMS is innovative for its ability to reason on a specific domain, improve the accuracy and completeness of information managed. This work is relevant for any domain involving administrative documents and regulations (e.g. fiduciary or insurance sector).

1. Introduction

Organisations are using Document Management System (DMS) tools to simplify the management of their digital documents and files. A DMS is used to create, store, organise, retrieve, and update documents and files in a secure and efficient manner [8]. By using a DMS, organisations can streamline their document-dependent workflows, reduce manual document handling, and improve the accuracy and accessibility of their information.

A DMS can use metadata to organise and categorise documents. By tagging documents with metadata, users can easily search and retrieve documents based on specific criteria, such as date range, author, or content.

In the last ten years, the emergence of metadata-driven DMS platforms has transformed the way organisations handle their documents. These platforms have simplified the task of classifying, searching, and retrieving documents, resulting in enhanced productivity and collaboration among employees. Moreover, the adoption of cloud-based DMS solutions has enabled staff to access and work on documents from anywhere, at any time, further augmenting the efficiency and effectiveness of document-based workflows.

Efficient document management is crucial not only for the optimal retrieval and use of documents but also for effective and efficient work organisation. Indeed, Gorelashvili [6] notes that in the legal sector, automated document management is essential to improve and streamline the way lawyers manage their practice. DMS ensures that documents are easily accessible, well-organised, and protected. Abbasova [1] highlights the beneficial effects of DMS on workflow forms by automating the routing of documents between people, eliminating bottlenecks and optimising business processes.

⋆This document is the results of the research project n. 50606.1 IP-ICT "Addmin" funded by Innosuisse.

⋆⋆The present paper is an extended and revised version of the paper [3] presented at DMSVIVA 2023.

*Corresponding author

maria.cappelli@unige.ch (M.A. Cappelli); ashley.caselli@unige.ch (A. Caselli); giovanna.dimarzo@unige.ch (G.D.M. Serugendo)

ORCID(s): 0000-0001-8492-0354 (A. Caselli)

DOI reference number: 10.18293/JVLC2023-N2-034

DMS ensures more accurate organisation of business processes within the company through effective management and support of the quality system in accordance with inter-national standards, as well as efficient storage, management, and access to information and knowledge. Gostojić et al. [7] list five main benefits of DMS for organisations, including paper cost savings, more efficient use of space, and increased productivity. It also provides document security, easy access to documents, and a version control feature that allows ac-cess to previous versions. In addition, it provides damage control through backup creation and repository export and ensures consistency of procedures through protocol enforce-ment. Yousufi [19] discusses the benefits of a "paperless" workplace, where the use of paper is reduced through the digitisation of documents and the use of DMS. Among the benefits highlighted are the ability to save space, time and money, as well as improving document security and simpli-fying data transfer, and a positive impact on the environment. The paper production is associated with deforestation, the use of large amounts of water, and the production of green-house gas emissions.

Despite the benefits of using a DMS, some DMS solutions require significant setup processes and associated fees, or may not be well-suited to industry specific professions, creating further challenges for organisations seeking to implement an efficient DMS. In addition, since most solutions are enterprise-based, there are currently no solutions that automate access to critical documents for individual consumers. There is still no DMS that can classify, understand, and reason with customer documents, automatically process a bundle of customer documents and create a customer profile in compliance with regulations. In particular, in the domain of fiduciary, insurance brokerage, or tax returns, documents are still processed manually, sent by email or via a customer's cloud platform.

To overcome this problem this paper proposes a semantic rule-based approach for an RDF-based DMS. The proposed DMS uses a combination of various techniques: (1) an ontology for defining the concepts of the domain and their relationships (e.g. tax return); (2) data extracted from documents organised as RDF triples stored in a triplestore, following the ontology structure; and (3) Shapes Constraint Language (SHACL) rules for data validation, capturing regulations related to the domain, reasoning and inferring new information. The ontology organises the data in a structured way and defines the relationships between entities. This makes it easier to search and access documents. SHACL is an RDF-based rule specification language used to define shape properties, constraints and rules for data validation and verification [9]. Their combination allows the creation of a highly efficient, and automated DMS. In particular, the use of an ontology simplifies data organisation and management, while SHACL ensures data quality and reliability.

Besides providing a complete approach and a workflow, we addressed the specific case study of Swiss tax returns. We designed and implemented a rule-based process that dynamically builds, updates and reasons on users' profiles, integrating underlying legal regulations providing a DMS compliant by-design. An additional module completes this process by automatically extracting information from documents provided by users. Based on an ontology capturing the concepts of Swiss tax returns, we designed SHACL shapes to reason about asserted RDF triples of different tax households. A detailed description of the approach can be found in the technical report [4].

The remaining sections of the paper are organised as follows. Section 2 provides an overview of existing approaches in the context of DMS. Section 3 describes the proposed semantic-based approach in detail, our research questions, case study and workflow. The ontology used in the approach is discussed in section 4, and section 5 presents the rule-based methodology used to model the data. The implementation of the SHACL-based rules and their execution is shown in section 6, which includes details of how the rules were integrated into the system. Section 7 provides examples of validating RDF data against the defined SHACL shapes, and evaluating the defined rules. Finally, section 8 concludes the paper by summarising the main contributions of the proposed approach and discussing limitations and possible future research directions in the field of DMS.

2. Related work

Our proposed DMS system is innovative compared to other systems on the market because it uses a combination of semantic techniques (ontology and SHACL shapes), which allows even more precise and automated document management. In addition, the use of inference techniques allows new information to be derived from existing data, improving the accuracy and completeness of the information managed by the system. Therefore, we focus here on research works that either use ontologies or consider semantic approaches.

2.1. Ontology Approaches for DMS

Some researchers propose the use of ontologies as part of semantic document management approaches. Ontologies can formally define the structure, content and relationships of different types of documents. An ontology-based DMS can monitor document processes and workflows, track dependencies between documents, analyse how changes to one document may affect other related documents, and design the synchronisation steps needed to maintain consistency across interrelated document collections.

Fuertes et al. [5] develop an ontology for DMS concerning the construction sector. The ontology aims to classify documents along the lifecycle of the construction project, to reduce interoperability and information exchange problems, to establish a hierarchical structure of the different domains that correspond to the lifecycle of such projects, and finally to enable an interconnected system between these domains.

Doc2KG [16] is a framework that provides a continuous transformation of open data into a knowledge graph, using existing domain ontology standards. The system handles the initial conversion of a DMS into a knowledge graph and supports the continuous populating of the created knowledge

graph with new documents. The authors rely on a combination of natural language processing techniques to facilitate information extraction and constraint-solving techniques for knowledge graph creation and manipulation.

Lee et al. [10] develop a domain-specific ontology to support automatic document categorisation. The ontology contains a complete and detailed hierarchy of concepts used to represent documents related to information systems and technology as a set of concepts with relative weights. Although researchers recognise the advantages of using an ontology with classes in terms of the interpretability and comprehensibility of classification decisions, no reference is made to the definition of semantic rules to make the use of the ontology more flexible.

Sheng and Lingling [14] propose the use of ontology in the context of e-governance to model government data and create a semantic environment for managing government information. They present a semantic-based e-government system structure and use OWL as the ontology description language to provide the basis for data sharing and analysis. The authors detail the conceptual entity, the conceptual property and the relationship between concepts, which correspond to classes, properties and axioms respectively in the OWL language. However, they do not model semantic rules to define constraints and restrictions on the data, to ensure that it is consistent with the ontological structure they have defined, or to exploit its reasoning power.

Sladić et al. [15] present an approach to improve DMS through the use of a formal and explicit document model based on ontologies. This document allows the formal and explicit representation of the information contained in documents and the clear definition of concepts and relationships between them. In this way, the semantics of document content can be understood by machines, enabling more efficient and accurate analysis, classification and retrieval of documents. As a result, DMS can automatically classify documents based on their content, identify relationships between concepts, and support semantic search of documents.

2.2. Semantic Approaches for DMS

Wang et al. [18] organise and manage large amounts of documents through a representation of document semantics. The representation of document semantics is based on a set of attributes and a content vector, which allows for more accurate document identification and provides associative search capabilities. In addition, this study presents keyword-based indexing techniques and structural querying techniques for XML data, which are widely used for representing and exchanging data on the Web.

Amato et al. [2] propose a semantic analysis-based approach to make up for the lack of an adequate data structure of DMS. This lack can raise a problem for the application of appropriate security policies in DMS. The semantic methodology is able to retrieve information from specific parts of the document that can be useful for classification, security, etc. Semantic analysis serves for implementing fine-grained access control on sensitive data contained in unstructured and monolithic files, such as those found in DMS. The case study concerns the formalisation and protection of electronic health records.

Leukel et al. [11] propose a software architecture for cooperative semantic document management. They argue that semantic approaches to document management rely on enriching metadata and deriving semantic document models, but the quality of the metadata and the underlying domain ontology can limit the discovery of relationships between documents. The proposed software architecture aims to solve this problem by separating the semantic representation of individual documents from the knowledge of domain-specific relationships in two architectural layers.

Some of the above studies use ontologies to improve the effectiveness of DMS, while others use other semantic techniques. The former differ in their specific application domains and objectives, but they share the use of ontologies as a tool for semantic document management. These studies focus on semantic approaches to DMS but do not necessarily use semantic rules explicitly. These works often use techniques such as keyword-based indexing and structural retrieval, semantic analysis for sensitive data protection, or software architecture to improve the quality of metadata and domain ontologies. However, none of this research appears to use the combined approach of ontologies, and SHACL shapes, which can provide a more comprehensive and sophisticated DMS. Such an approach can ensure data quality and consistency while improving the effectiveness of document analysis, classification and retrieval.

3. Semantic-based approach

We discuss here our approach, starting with research questions, defining our case study, and providing our global workflow and architecture.

3.1. Research questions

The research questions proposed in this study relate to the processing of business documents, in particular administrative documents, for the company's customers. The aim is to propose a semantic rule-based approach that can help companies manage their customers' documents more efficiently and effectively. Our proposal addresses the following research questions:

A) How can a document be classified and multi-labelled based on extracted information that provides its key features?
B) How can customer profiles be created and updated based on the documents provided and the information extracted from them?
C) How can a reasoning process determine which documents customers need to provide based on their profiles?

3.2. Case study

In this paper, we focus on the Swiss tax return of households and the documents required to complete the tax return. We limit our case study to household profiles consisting of a

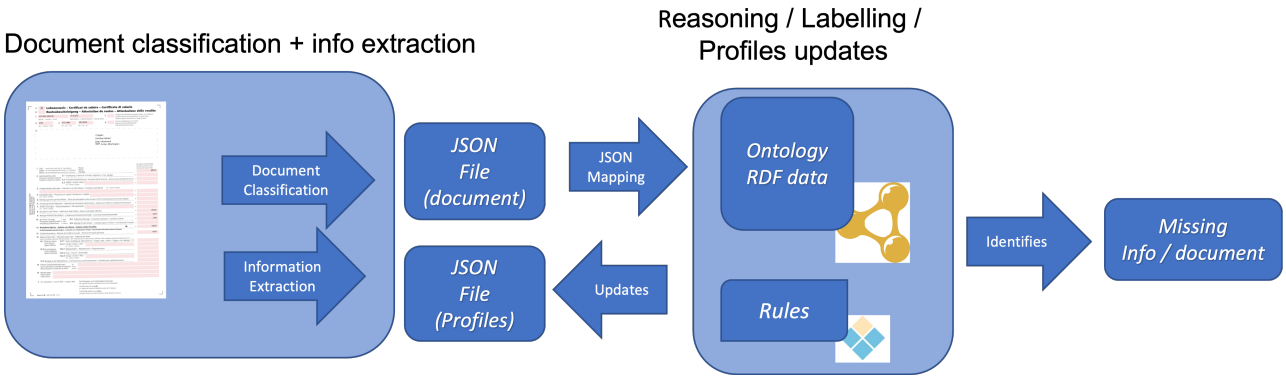

Figure 1: Overview of the Global Workflow as presented in [4]

single person, a widow/widower, couples, households with or without children or other dependants, retired or working. We have also limited our case study to the minimum set of administrative documents required to complete the Swiss tax returns of the households described above, namely: annual income, bank statements, health insurance policies and benefits, and family allowances, for each household member. We have included the health insurance statements because they are legally mandatory in Switzerland and everyone has to provide them for tax purposes.

Let us consider a tax household consisting of two working parents with children. As each parent is employed, data are extracted from their two salary certificates. The data extraction process identifies the main features of the document that are necessary conditions for a document to be classified as a wage (revenue) statement.

In response to the first research question A), shape properties and rules are applied to classify documents as salary certificates based on the extracted features. The system then assigns a double label (or tag) of "Tax" and "Income" to the salary statement.

In response to the second research question B), the system profiles both parents as employees.

Finally, in relation to the third research question C), the system identifies other necessary documents that the two parents and their children need to provide, such as health insurance.

3.3. Global Workflow and Architecture

Figure 1 shows the global overview of the workflow of our approach, which is described in detail in [4].

Such an architecture consists of three modules:

1) The documents (native PDFs or scanned documents) are processed by a *Document Classification and Information Extraction Module*. This module generates *JSON files for each document*, identifying its classes (e.g. health insurance policy, etc.), as well as specific information extracted from the document, such as date and amount.
2) The information extracted from the documents is also used to feed *JSON files profiles* of the household and its various members (e.g. widow/er, child, etc.);
3) The JSON information is then mapped to RDF by the *Reasoning, Labelling and Profiles Updates* module, using an ontology for Swiss tax returns and personal profiles. This module also contains a semantic rule-based reasoner, which is used on the one hand to update the information in the profiles (e.g. health insurance for a new child means that the child must be added to the household, possibly changing the household profile from a couple without children to a couple with children), and on the other hand to identify any missing documents of the household based on the existing profiles (e.g. health insurance or benefits are missing for a person identified as part of the household).
The *Reasoning, Labelling, and Profiles update* module has the following components: *(i)* an *ontology for managing Swiss tax and administrative documents*, based on actual official legal tax documents and on actual administrative documents needed to complete the tax declaration. The ontology defines concepts such as documents, user profiles, tax items, and changes in residence and status; *(ii)* The *rules* defined for validating documents, updating profiles based on new information, labelling documents, identifying missing documents (e.g. not provided in the bundle), integrating legal regulations aspects; and *(iii)* the *RDF data* mapped from the JSON files (actual data) containing information automatically extracted from the documents using an information extraction module.

4. Ontology for the management of tax and administrative documents

The ontology for the management of tax and administrative documents is the foundation for the DMS. It includes classes such as tax and administrative documents, user profiles, tax items and changes of status and domicile. The ontology has been developed in French[1] through Protégé [12].

The ontology creation process was based on a middle-out approach [17] because of the limited number of documents

[1]The ontology is available (on request) at the following website https://gitlab.unige.ch/addmin/doc-onto/-/wikis/home

available at the beginning of the project. The middle-out approach allows us to start with a limited set of data and then gradually expand the system as we acquired more information. This approach also ensures to adapt the system to user needs and to improve its accuracy over time. The step-by-step development process involved analysing tax return forms from 2020, as well as the instructions for filling them out issued by the Canton of Geneva, and model forms from other documents such as salary statements, 2nd and 3rd pillar pension funds, health insurance, etc. The results of the step-by-step development process were further validated by tax experts to ensure their validity and consistency over time.

The middle-out approach was applied as follows. First, we identified the basic concepts of the domain in terms of *(i)* documents and *(ii)* user profiles. We then developed two further parts for specific sub-areas, such as *(iii)* a section for tax items, representing the different categories of taxes and fees that the taxpayer has to pay, and *(iv)* a section for managing changes of status and domicile (address) of the user. These parts were then integrated into a larger and more complex ontology for the domain of tax and administrative documents.

To facilitate the integration of the different parts, we used SHACL shapes to define the relationships between the RDF nodes to ensure that the parts were correctly and consistently integrated into the larger ontology (section 6).

Figure 2 shows the middle-out process for generating the ontology.

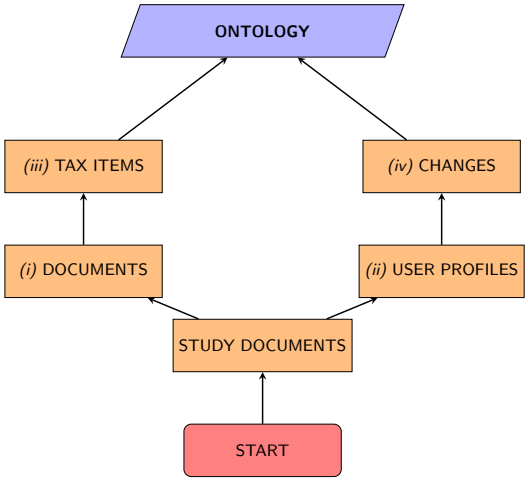

Figure 2: The development of the ontology

The middle-out approach takes into account real-world details, allowing specific domain information to be incorporated into ontological models. Finally, the approach helps to reduce the risk of instability and inconsistencies and ensures that the system can be tested and user feedback can be gathered more quickly as it is developed incrementally.

The results of the ontology development process are summarised in figure 3, which shows an extract of the ontology metrics, including 240 classes, 24 data type properties, 613 axioms, and 15 object properties.

Ontology metrics:

Metrics	
Axiom	613
Logical axiom count	327
Declaration axioms count	279
Class count	240
Object property count	15
Data property count	24

Figure 3: Ontology metrics

4.1. Ontology - Swiss Tax return

The ontology was developed following the steps (proposed by Noy and McGuinness [13]) to describe the domain of document management for tax and administrative returns. The following steps are followed:

1) **Defining the domain and scope of the ontology**.
In order to define the scope of the ontology and its functionalities, we asked questions such as: What tax and administrative documents are relevant to a particular tax return? How should these documents be organised and archived? What metadata is associated with the documents? What are the relationships between the documents, such as hierarchical dependencies between documents? What data needs to be extracted from the documents? Which household and user profiles are relevant for tax return? Answering these questions helped us to define the scope of the ontology and to identify its main functionalities.

2) **Defining the classes and the class hierarchy**.
The top-level classes of the ontology are: *(i)* documents, *(ii)* user profiles, *(iii)* tax items, *(iv)* changes. They have several middle-level classes. These middle-level classes are then further organised into sub-classes. Defining middle-level classes helps to make the ontology more understandable. It also allows to search and retrieve information from the ontology, as users can navigate through the hierarchy to find the information they need.

   (i) Documents. Within our ontology, we have defined a hierarchy of classes for fiscal and administrative documents. The top-level class Document represents the parent class of all other classes, including: “Copie”, “Attestation”, “Bordereau”, “Certificat”, etc. Additionally, we have defined further sub-classes for each of these classes to organise the different types of documents in a hierarchical manner, as shown for the class “Certificat” in figure 4. Among these, we distinguish for instance the Salary Certificate “CSalaire”, which is the main document for income report.

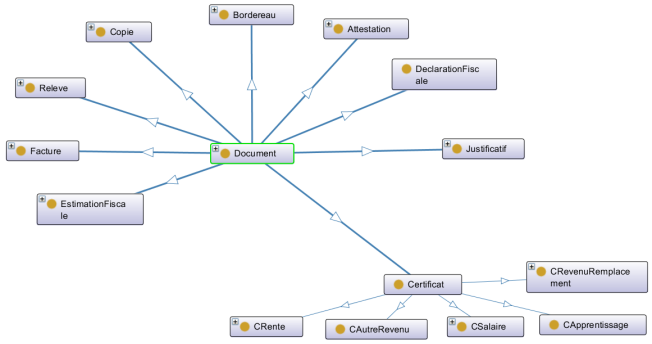

Figure 4: The section of *Documents* within the Ontology (figure adapted from Protégé ontology editor tool

(ii) User profiles. We identified the user profiles on the basis of a set of relevant and significant criteria for the ontology of the tax and administrative documents concerned. We chose the following criteria: type of employment, ownership of real estate, financial accounts, or movable property (e.g. car), any dependant person linked to the household, and civil status as shown figure 5.

The criteria used to identify user profiles were chosen on the basis of their relevance and importance for the ontology of the tax and administrative documents in question. The employment type criterion was used because the applicable tax regime and the required documents may vary depending on the employment status of the user. For example, an employee may have different tax documents than a self-employed person or an entrepreneur. We selected the criterion of Ownership of real estate or financial accounts, because ownership of such property may affect the user's tax situation. For example, the ownership of real estate may require the submission of specific documents for tax declaration. Similarly, the criterion of the number of dependants affects the user's tax situation and the documents required. For example, the presence of dependant children requires the submission of specific documents in order to obtain tax benefits. Finally, we defined the marital status criterion, because it affects the household tax situation and the documents required. Marriage or cohabitation may have tax implications and requires the submission of specific documents.

The ontology also includes user profiles, which are defined by a combination of profile elements. For example, a single person with real estate and a car would be defined by the combination:

Célibataire ⊓ AvecBienImmobilier ⊓ AvecBienMobilier

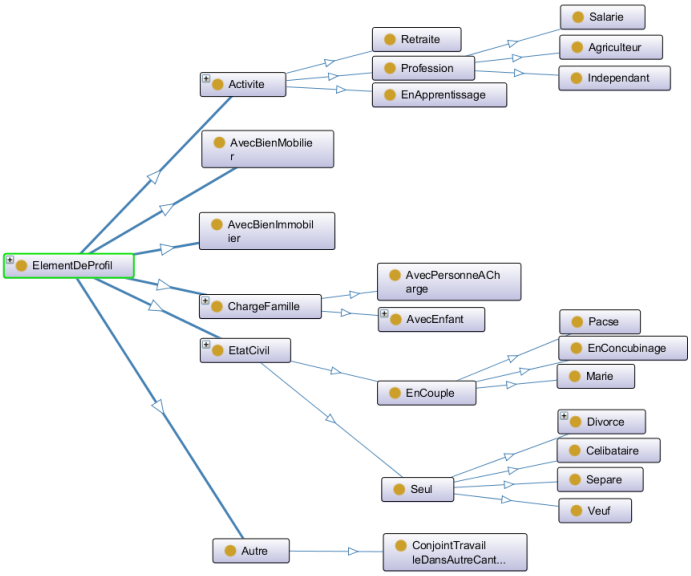

Figure 5: The section of *User Profiles* within the ontology (figure taken from Protégé ontology editor tool)

(iii) Tax Items. The ontology defines the tax items, which represent the different categories of taxes and duties applicable in a given Geneva tax jurisdiction. The list of these items is directly linked to the Geneva Tax return form. Each document in the ontology is associated with one or more tax items. Figure 6 provides a visual representation. Tax items are organised into three groups: Deduction (linked to any deduction we can provide such as doctors' bills or work related travel expenses); Fortune (savings accounts, real estate, etc.); Revenu (any type of income or rent from various activities).

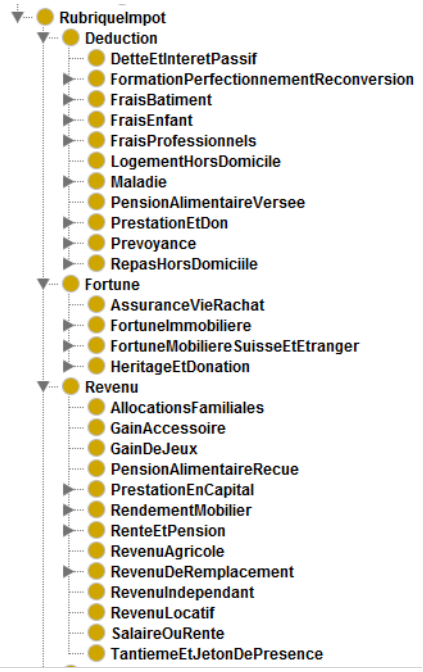

Figure 6: The section of *Tax Items* within the ontology

(iv) Changes. The ontology includes a definition for household profile changes. Figure 7 gives a visual representation of the section of changes. Similarly to the Tax items above, it is linked to the Geneva Tax return form and includes information related to changes in marital status, income activities, children, domicile, etc.

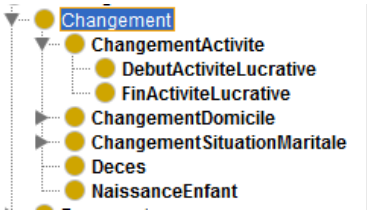

Figure 7: The section of *changes* within the ontology

3) **Defining the properties of the classes**
We defined properties for documents in relation to data that we need to extract from the documents and that serve to classify, label the document, or build a person or household profile.
For each class of document, these elements are identified through a combination of two methods: data extraction and evaluation of their usefulness for the documents in question. We identified the most recurrent properties for all documents, as shown in table 1. Specific machine learning algorithms are employed to extract information from administrative documents (e.g. from health insurance documents). This information is then stored as an RDF triple on which further reasoning will apply.

Table 1
Ontology Properties

Property	Description
nomPersonne	The name of a person
dateNaissance	The date of birth of a person
adressePersonne	The address of a person
emetteur	The issuer of an invoice
destinataire	The recipient of an invoice
echeanceContrat	The deadline for fulfilling the terms of a contract
canton	The canton in which a taxpayer is domiciled
commune	The municipality in which a taxpayer is domiciled
noAVS	The AVS number of a taxpayer
devise	The currency used in a fiscal document
noClient	The client number assigned to a taxpayer by a tax authority
noContrat	The unique identifier assigned to a contract
noCompte	The bank account number associated with a taxpayer
noDepot	The file number assigned to a tax file
codePostal	The postal code of a taxpayer's address
codeCommuneTravail	The municipality in which a taxpayer works
adresseLieuTravail	The address of a taxpayer's workplace
anneeFiscale	The fiscal year to which a tax document or liability relates
montant	The monetary amount associated with a fiscal document

Figure 8 shows the common properties for an account certificate, such as "IBAN", "account number", "user client", "account opening date" and "account closing date".

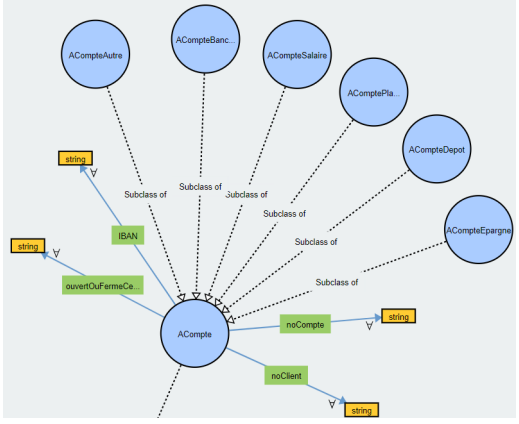

Figure 8: Ontology properties for an account certificate (figure adopted by WebVOWL tool)

5. Data Model expressed with Description Logic

In order to perform semantic reasoning, we need to define "shapes" to describe the structure, properties, and relationships of data. We first express here these shapes and relationships in Description Logic, before expressing them in SHACL in the next Section.

This approach addresses the research questions A, B, and C defined in Section 3.

A) **Document classification and multi-labelling rules**
Classes of documents are uniquely defined by distinctive characteristics. For example, a salary certificate document must contain the employee's last name and first name, the employer, and the amount of the salary. The following Description Logic (DL) notation represents some basic requirements that must be met for a document to be classified as a salary certificate.

SalaryCertificate
$\sqsubseteq delivered By$ some $Employee$
$\sqcap contains$ some $EmployeeLastName$
$\sqcap contains$ some $EmployeeFirstName$
$\sqcap contains$ some $SalaryAmount$

For the multi-labelling rule, one or more labels are assigned to each document to allow the automatic organisation of the documents into several predefined categories. The following DL notation is used to assign one or more labels to a salary statement that has a double "Tag".

SalaryCertificate
$\sqsubseteq hasTag$ some $Income$
$\sqsubseteq hasTag$ some Tax

B) **Customer profile rules**
In order to infer the users' profile by analysing the documents they provided, some **direct rules** are defined. For example, if a user delivers a salary certificate then the user is tagged as being an employee. The following DL notation allows the automated inference of user profile based on the documents the user provide.

DocumentDelivery
$\sqsubseteq delivered By$ some $User$
$\sqcap isType$ some $SalaryCertificate$
$\sqsubseteq deliverd By$ some $Employee$

C) **Documents delivery rules**
In order to infer which documents match the profile of the user, some **inverse rules** are defined. For example, if a person is tagged as an employee, then this person has to deliver a salary certificate. Additionally, each person has to provide a health insurance policy. The following DL notation allows for the automated inference of the type of documents to be delivered based on the user's profile.

$Person$
$\sqcap isType$ some $Employee$
$\sqsubseteq hasToDeliver$ some $SalaryCertificate$

The SHACL shapes are useful to define a set of property shapes and semantic rules for multi-label document classification and user profiling.
Specifically, we defined 92 property shapes and three sets of semantic rules, resulting in a total of 120 rules. These rules included 78 multi-label rules, 21 customer profile rules and 21 document delivery rules. Using SHACL allowed us to validate RDF data against these rules and ensure that the data complied with the defined constraints.

6. Implementation

We implemented SHACL node shapes for each class of the ontology. Each node shape is linked to its respective class through `sh:targetClass` property. By defining a target class within the shape's definition, it becomes applicable to all instances of that specific class. For example, *Salary Certificate Shape* is a type of node shape, and it is connected to the *SalaryCertifcate* class. By using the target, it is possible to ensure that all instances of *SalaryCertificate* are checked against the conditions defined within the *SalaryCertificateShape*. Below, we present the implementation of the SHACL shapes, following the adopted methodology described in section 5.

A) **Classification of documents**
For each document that a user should submit, we identify its *sine qua non* elements.
Within the relative SHACL shape, we defined the relevant elements the document must contain as SHACL property shapes. As shown in listing 1, a salary certificate has to contain: the employee's surname and first name, one and only one employee, and the amount.
The predicates have constraints that can describe different values for each attribute shape. We use pre-built constraint types as `sh:datatype` to describe the type of literal values; `sh:minCount` to describe the maximum required number of values; `sh:maxCount` to specify the maximum number of value nodes.
By using such a SHACL shape, we can run a validation process that validates (or not) the document as being of the appropriate type. This can also be interpreted as "if the document contains all *sine qua non* elements, i.e. the validation is positive, then it is of that specific class", and is therefore assigned to that class.

A) **Multi-labelling documents**
Multi-labelling rules assign one (or more) label to each document. The assigned labels can then be used to automatically organise documents into predefined categories. We defined such rules as SHACL inference rules and their execution generates inferred triples of the form:

```
< document impots:tag label >
```

The `document` is the RDF individual of the document that is being labelled; `impots:tag` is a data property, defined in the ontology, for assigning the label to a document; and `label` is a string literal (`xsd:string`) containing the actual text value of the label.
Listing 2 shows a rule labelling a document of type salary certificate as both a tax document and an income document.

```
impots:SalaryCertificateShape
  rdf:type sh:NodeShape ;
      sh:property [
      sh:path impots:personSurname ;
      sh:datatype xsd:string ;
      sh:minCount 1 ;
      sh:maxCount 1 ;
      sh:class impots:PersonSurname ;
      sh:name "Person Surname" ;
    ] ;
    sh:property [
      sh:path impots:employer ;
      sh:minCount 1 ;
      sh:maxCount 1 ;
      sh:class impots:Employer ;
      sh:name "Employer" ;
    ] ;
    sh:property [
      sh:path impots:personFirstName ;
      sh:datatype xsd:string ;
      sh:minCount 1 ;
      sh:maxCount 1 ;
      sh:class impots:PersonFirstName ;
      sh:name "Person First Name" ;
    ] ;
  sh:property [
      sh:path impots:amount ;
      sh:datatype xsd:string ;
      sh:minCount 1 ;
      sh:maxCount 1 ;
      sh:class impots:SalaryAmount ;
      sh:name "Salary Amount" ;
    ] ;
.
```

Listing 1: Relevant features of a salary certificate document represented as SHACL shapes

```
impots:SalaryCertificateShape
    rdf:type sh:NodeShape ;
    sh:rule [
        rdf:type sh:TripleRule ;
        sh:subject sh:this ;
        sh:predicate impots:tag ;
        sh:object "Tax" ;
    ] ;
    sh:rule [
        rdf:type sh:TripleRule ;
        sh:subject sh:this ;
        sh:predicate impots:tag ;
        sh:object "Income" ;
    ] ;
    sh:targetClass impots:SalaryCertificate ;
.
```

Listing 2: SHACL inference rule for labelling a document of type salary certificate

Suppose we have a knowledge graph containing data from various documents, including some salary certificates. By executing the rule shown in listing 2, we can infer new triples that explicitly state the type of the salary certificate. These inferred triples are shown in listing 3.

```
:documentX impots:tag "Tax" .
:documentX impots:tag "Income" .
```

Listing 3: Triples inferred by the inference rule shown in listing 2

B) **Customer profile rules**

As multi-labelling rules, the customers' profile rules are defined as SHACL inference rules.

Listing 4 shows an example of such rules. Contrary to the example previously shown (where the targeted documents are all the RDF individuals of a defined class), this example shows an extended targeting condition expressed using the SPARQL language.

The rule defined in listing 4 states that all nodes that satisfy the shape must have the `rdf:type property` equal to `impots:Employee`.

Therefore, if all nodes that represent the recipients of a salary certificate have `rdf:type` equal to employee, the validation of the shape will be successful. On the other hand, if one or more recipients of a salary certificate are not correctly represented in the RDF graph (because `rdf:type` is different from employee), the validation of the shape will be negative and an error will be reported.

```
impots:SalaryCertificate_Employee-Shape
    rdf:type sh:NodeShape ;
    sh:rule [
        rdf:type sh:TripleRule ;
        sh:object impots:Employee ;
        sh:predicate rdf:type ;
        sh:subject sh:this ;
    ] ;
    sh:target [
        rdf:type sh:SPARQLTarget ;
        sh:prefixes impots: rdf: ;
        sh:select """
            SELECT ?this
            WHERE {
                ?sc rdf:type
                    impots:SalaryCertificate .
                ?sc impots:recipient ?this .
                ?this rdf:type impots:Person .
            }
        """ ;
    ] ;
.
```

Listing 4: SHACL direct rule inferring the employee profile of a person from the salary certificate provided

The execution of this direct rule infers new RDF triples of the form:

`< person rdf:type impots:Employee >`

where `person` corresponds to the specific RDF individual; `rdf:type` is the property used to state that a resource is an instance of a class; and `impots:Employee` is the inferred class to which `person` belongs.

C) **Document delivery rules**

The rule defined within `impots:EmployeeShape` in listing 5 aims to verify the correspondence between employees and the salary certificates they have delivered. In particular, the shape uses a validation rule that requires all nodes representing employees to have a "deliver" relationship with at least one node representing a salary certificate.

```
impots:EmployeeShape
  rdf:type sh:NodeShape ;
  sh:rule [
      rdf:type sh:TripleRule ;
      sh:subject sh:this ;
      sh:predicate impots:delivers ;
      sh:object impots:SalaryCertificate;
    ] ;
  sh:targetClass impots:Employee ;
.
```

Listing 5: SHACL inverse rule inferring the need for an employee profile to deliver a salary certificate

The execution of the rule defined in listing 5 infers new triples of the form:

```
< employee impots:delivers impots:SalaryCertificate >
```

which means that every employee has to deliver a salary certificate.
The shape of the rule *PersonShape* in listing 6 uses a validation rule, expressed as a triple rule, which specifies that all nodes representing persons must have a deliver relationship with at least one node representing an instance of health insurance. In other words, the shape verifies whether the persons have delivered at least one health insurance certificate.

```
impots:PersonShape
  rdf:type sh:NodeShape ;
  sh:rule [
      rdf:type sh:TripleRule ;
      sh:subject sh:this ;
      sh:predicate impots:delivers ;
       sh:object impots:HealthInsurance;
    ] ;
  sh:targetClass impots:Person ;
.
```

Listing 6: SHACL inverse rule for inferring that a person profile needs to deliver a Health Insurance

The execution of the rule defined in listing 6 infers new triples of the form:

```
< person impots:delivers impots:HealthInsurance >
```

which in turn means that each person has to deliver a health insurance policy document.

7. Validating and Evaluation

We first show how we validate the data against SHACL shapes, and second how we evaluate the rules.

7.1. Validating data with SHACL Shape

To validate the RDF data against the defined SHACL shapes, we use a SHACL validation engine such as the one integrated into TopBraid Composer[2], called the TopBraid Validator.

We create SHACL validation test cases in TopBraid Composer to ensure that the RDF data conforms to the specified shapes. These test cases define a set of RDF data and corresponding SHACL shapes, as well as validation constraints that must be applied to this data to verify its compliance with the RDF data model specifications. The test cases allow the correct implementation of the SHACL validation rules to be verified and any validation errors to be detected. This ensures that the data is accurate, consistent and conforms to the specifications of the RDF data model.

We wrote six graph validation test cases with regard to 3rd pillar attestation, Deposit account attestation, AVS[3] pension or disability insurance attestation, LPP[4] pension attestation, and Salary certificate. Each test case performs a SHACL constraint validation on the entire graph and compares the results with the expected validation results stored with the test case.

The test case for "PersonShape", in listing 7, defines two instances of the `impots:Person` class: an invalid resource and a valid resource. The former has a value for the *noAVS* property that violates a constraint defined in the *AVS shape*. Indeed, the AVS number must always start with "756", and must be followed by two groups of 4 digits, and finish with a group of two digits. In this case, it properly starts with "756", but then continues with only three digits "023" instead of four. The latter satisfies all the constraints defined in the shape, as the second group is made of 4 digits and is "0123". We deliberately insert this error to verify that the SHACL rules work correctly and are able to detect any problems or violations of the specified constraints.

The expected result of the validation is defined as a validation report, with information about the constraint that was violated, the form that defines the constraint, the path to the property that caused the violation, the value causing the violation, and the severity of the violation. The report will also indicate that the validation did not conform.

[2]For Top Braid Composer, see: http://www.topquadrant.com/products/TB_Composer.html

[3]The AVS or OASI number is the social insurance number uniquely associated with individuals in Switzerland https://www.bsv.admin.ch/bsv/en/home/social-insurance/ahv/legal-bases-and-legislation/ahv-nummer.html

[4]Pension related fund

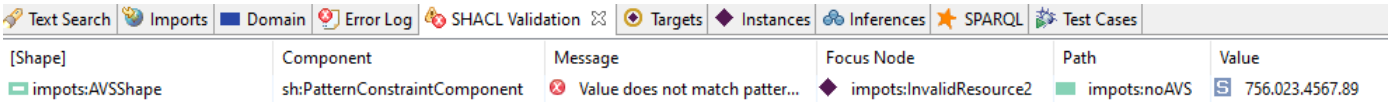

Figure 9: Results of SHACL validation

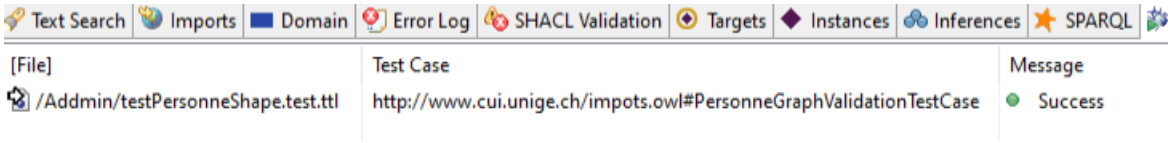

Figure 10: Result of "Person Shape" test case

```
<http://www.cui.unige.ch/PersonShape.test>
  rdf:type owl:Ontology ;
  rdfs:label "Test of PersonShape" ;
  owl:imports <http://datashapes.org/dash> ;
  owl:imports <http://www.cui.unige.ch/impots.shapes> ;
  owl:versionInfo "Created with TopBraid Composer" ;
.
impots:InvalidResource2
  rdf:type impots:Person ;
  impots:noAVS "756.023.4567.89" ;
  impots:personSurname "Zola" ;
  impots:personFirstName "Giovanna" ;
.
impots:PersonGraphValidationTestCase
  rdf:type dash:GraphValidationTestCase ;
  dash:expectedResult [
      rdf:type sh:ValidationReport ;
      sh:conforms "false"^^xsd:boolean ;
      sh:result [
          rdf:type sh:ValidationResult ;
          sh:focusNode impots:InvalidResource2 ;
          sh:resultPath impots:noAVS ;
          sh:resultSeverity sh:Violation ;
          sh:sourceConstraintComponent
          sh:PatternConstraintComponent ;
          sh:sourceShape impots:AVSShape ;
          sh:value "756.023.4567.89" ;
        ] ;
      .
impots:ValidResource
  rdf:type impots:Person ;
  impots:noAVS "756.0123.4567.89" ;
  impots:personSurname "Zola" ;
  impots:personFirstName "Giovanna" ;
.
```

Listing 7: Graph validation test case of "PersonShape"

The SHACL test case returned the error message for the invalid resource *AVS*, as shown in figure 9. The presence of this error message indicates that the SHACL rules are working correctly and that the invalid resource has been identified and reported.

Figure 10 shows that the result of the "PersonShape" test case has been successful. This means that all the data instances that satisfy the "PersonShape" also satisfy the SHACL rules specified for that shape. This is a positive result, indicating that the validated data conforms to the SHACL rules and that the applied SHACL rules work correctly on this data.

Figure 11 shows positive results for the other five tests cases concerning five tax documents, such as 3rd pillar attestation, deposit account attestation, AVS pension or disability insurance attestation, LPP pension attestation, salary certificate.

7.2. Evaluation

After validating the data against the defined shapes, we applied SHACL rules to the dataset to generate new information and improve data quality. In this paragraph, we will describe the results of applying the SHACL rules on the validated data and evaluate the effectiveness of the SHACL rules in meeting the requirements of the application.

7.2.1. Evaluation of multi-labelling rules

As we can see in figure 12, the execution of the rules shown in listing 2 infers two new triples that assign the two labels "Income" and "Tax" to the document with ID "Salary C 12.3.334", which is of type "SalaryCertificate".

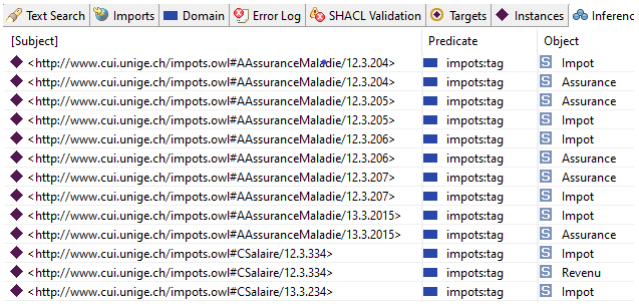

Figure 12: Inferred triples that assign two labels to a document of type salary certificate

7.2.2. Evaluation of users profile rules

We defined two individuals Zola Giovanna and Ladoumegue Jules. We assume that Zola Giovanna has provided a salary certificate. Based on the direct rule defined in listing 4, since Zola Giovanna delivered such a certificate, the rule infers that she is an employee. Figure 13 shows the inferred triples.

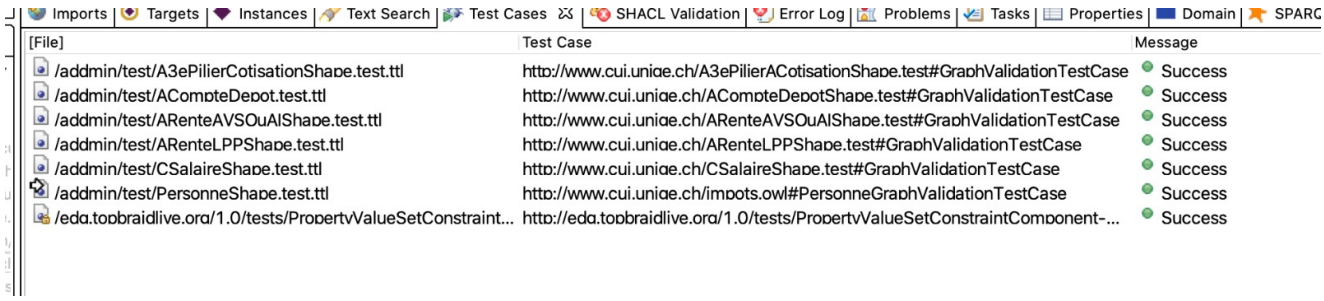

Figure 11: Result of six shape test cases

Figure 13: The result of the execution of direct rules to a person who has issued a salary certificate

Conversely, we defined Ladoumegue Jules as an employee. Therefore, according to the inverse rule defined in listing 5, the execution infers that since he is of class employee, he must deliver a salary certificate document. According to Listing 6, since he is also a person, he has to provide a health insurance policy. Figure 14 shows the inferences mentioned.

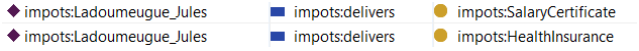

Figure 14: The result of running inverse rules on a person with an employee profile

8. Conclusion

This paper presents a semantic rule-based approach for a semantically enriched DMS, which facilitates the management of administrative documents, user profiling, and other related document management activities. This approach is capable of overcoming the limitations of traditional Document Management Systems (DMS) that rely solely on metadata to organise documents. The proposed approach uses a combination of ontology and SHACL rules to capture knowledge of the domain and legal regulations, validate data, and infer new information. The process is designed to be dynamic and based on data provided by users. Additionally, the process takes into account users' profiles and underlying rules to enable accurate and automated document management.

The paper demonstrates the innovative nature of the proposed approach and its potential to improve the accuracy and completeness of information managed by a DMS. The ontology used in the process captures specific concepts of Swiss tax returns, while the SHACL rules are used to validate and reason about asserted RDF triples of actual data from different tax households on the basis of actual regulations.

The limited availability of Swiss tax documents presented a major challenge during the development of the system. As a large dataset of tax documents is required to perform machine learning tasks and related analyses, this was an obstacle to the progress of the project. This had an impact on the quality of the information extraction module. However, it did not interfere with the ontology or the SHACL rules, which cover most of the various household configurations. Nevertheless, the implementation still needs to be validated on a large dataset of documents. Future work will consider the dynamic nature of profiles and the integration of such a module into a wider DMS service.

Furthermore, two additional challenges were identified during the development of the project. The first challenge was dealing with multilingualism, as Switzerland has four official languages (German, French, Italian, and Romansh). This required the project team to develop techniques for processing and analysing tax documents in multiple languages. The second challenge was managing the different tax laws enacted by the cantons. Each canton has its own laws, rules, tax documents, and procedures, so it was necessary to develop a flexible system that could adapt to the specific requirements of each canton.

It should be noted that while the project has a broader scope, targeting any type of administrative document, the work presented in this paper focuses specifically on Swiss tax return documents written in French. The project has not dealt with German, Italian, or English documents.

It is important to note that a business-to-business (B2B) system for professionals integrating this solution must also take into account privacy and confidentiality issues. These concerns must be carefully considered and addressed to ensure that the system complies with relevant laws and regulations regarding privacy and confidentiality.

In general, the adoption of our semantic approach could simplify the management of administrative documents and improve user profiling. The proposed DMS can be used in various contexts, such as tax filing, document management for an insurance company, or legal document management.

Acknowledgments

This research was supported by Innosuisse within the framework of the innovation project 50606.1 IP-ICT "Addmin". The authors thank Anne-Françoise Cutting-Decelle, Assane Wade, Claudine Métral, Gilles Falquet, Graham Cutting, and Sami Ghadfi for their valuable collaboration with

the “Addmin” project.

References

[1] Abbasova, V., 2020. Main concepts of the document management system required for its implementation in enterprises. ScienceRise 1, 32–37. doi:10.21303/sr.v0i1.1149.

[2] Amato, F., Casola, V., Mazzocca, N., Romano, S., 2011. A semantic-based document processing framework: A security perspective, in: 2011 International Conference on Complex, Intelligent, and Software Intensive Systems, pp. 197–202. doi:10.1109/CISIS.2011.37.

[3] Cappelli, M.A., Caselli, A., Di Marzo Serugendo, G., 2023. Enriching rdf-based document management system with semantic-based reasoning, in: Chang, S. (Ed.), The 29th International DMS Conference on Visualization and Visual Languages, KSIR Virtual Conference Center, USA, June 29-July 3, 2023, KSI Research Inc.. pp. 44–50. URL: https://doi.org/10.18293/DMSVIVA23-034, doi:10.18293/DMSVIVA23-034.

[4] Di Marzo Serugendo, G., Falquet, G., Metral, C., Cappelli, M.A., Wade, A., Ghadfi, S., Cutting-Decelle, A.F., Caselli, A., Cutting, G., 2022. Addmin: Private computing for consumers’ online documents access: Scientific technical report .

[5] Fuertes, A., Forcada, N., Casals, M., Gangolells, M., Roca, X., 2007. Development of an ontology for the document management systems for construction, in: Complex Systems Concurrent Engineering. Springer, pp. 529–536.

[6] Gorelashvili, L., 2023. The importance of digitalization of legal documents preparing process and its impact on peoples’ legal guarantees, in: Geibel, R., Machavariani, S. (Eds.), Digital Management in Covid-19 Pandemic and Post-Pandemic Times. Springer, Cham. doi:10.1007/978-3-031-20148-6_3.

[7] Gostojić, S., Sladić, G., Milosavljević, B., Zarić, M., Konjović, Z., 2014. Semantic driven document and workflow management, in: Proceedings of the international conference on applied internet and information technologies (ICAIIT 2014). Zrenjanin, Serbia, pp. 229–234.

[8] IEC, I., 2001. 82045-1, document management–part 1: Principles and methods. International Organization for Standardization .

[9] Knublauch, H., Kontokostas, D., 2017. Shapes Constraint Language (SHACL). W3C Recommendation. W3C. URL: https://www.w3.org/TR/2017/REC-shacl-20170720/.

[10] Lee, Y.H., Hu, P.J.H., Tsao, W.J., Li, L., 2021. Use of a domain-specific ontology to support automated document categorization at the concept level: Method development and evaluation. Expert Systems with Applications 174, 114681. URL: https://www.sciencedirect.com/science/article/pii/S0957417421001226, doi:https://doi.org/10.1016/j.eswa.2021.114681.

[11] Leukel, J., Schuele, M., Scheuermann, A., Ressel, D., Kessler, W., 2011. Cooperative semantic document management, in: Business Information Systems: 14th International Conference, BIS 2011, Poznań, Poland, June 15-17, 2011. Proceedings 14, Springer. pp. 254–265.

[12] Musen, M.A., 2015. The protégé project: a look back and a look forward. AI Matters 1, 4–12. URL: https://doi.org/10.1145/2757001.2757003, doi:10.1145/2757001.2757003.

[13] Noy, N.F., McGuinness, D.L., et al., 2001. Ontology development 101: A guide to creating your first ontology.

[14] Sheng, L., Lingling, L., 2011. Application of ontology in e-government, in: 2011 Fifth International Conference on Management of e-Commerce and e-Government, IEEE. pp. 93–96.

[15] Sladić, G., Cverdelj-Fogaraši, I., Gostojić, S., Savić, G., Segedinac, M., Zarić, M., 2017. Multilayer document model for semantic document management services. Journal of Documentation 73, 803–824.

[16] Stylianou, N., Vlachava, D., Konstantinidis, I., Bassiliades, N., Peristeras, V., 2022. Doc2kg: Transforming document repositories to knowledge graphs. Int. J. Semantic Web Inf. Syst. 18, 1–20. URL: https://doi.org/10.4018/ijswis.295552, doi:10.4018/ijswis.295552.

[17] Uschold, M., Gruninger, M., 1996. Ontologies: Principles, methods and applications. The knowledge engineering review 11, 93–136.

[18] Wang, G., Wang, B., Han, D., Qiao, B., 2005. Design and implementation of a semantic document management system. Information Technology Journal 4, 21–31.

[19] Yousufi, M., 2023. Exploring paperless working: A step towards low carbon footprint. European Journal of Sustainable Development Research 7.