University of Echahid Hamma Lakhdar d'El-Oued

Faculties of Exact Sciences

Computer Science department

Field: Mathematics and Computer Science

Sector: Computer science

Specialty: Distributed systems and Artificial Intelligence

To obtain the Master's degree in Computer Science

Master's thesis

Prepared by

SOUSSA Abdelhak
HERZALLAH Adel

Theme

Conception and realization of an intelligent judicial assistant system (JAS)

Discussed it by the committee composed of:

Supervisor: MEFTAH Mouhammed Charaf Eddine.

President of the jury: YAKOUB Mouhammed Amine.

Examiner: Medilah Sassi.

University year: 2022-2023

Abstract

In response to the dynamic nature of human society, laws and regulations undergo continuous modifications led by experts in the respective fields. In our specific national context, there has been a notable surge in revisions and updates to legal frameworks in recent years. This has resulted in the accumulation of a substantial repository of legal information. However, the manual retrieval of information from such a vast database is a time-consuming endeavour that can significantly impede the efficiency of administrators.

To address this challenge, we have put forth an IT solution centred around the application of ontology. This approach involves the utilization of UML for modelling the underlying knowledge structure. Following this, we employed Protégé as a tool for refining and enhancing the ontology. For the development phase, we selected the Visual Studio Code integrated development environment in conjunction with the Python programming language.

In terms of operational functionality, we harnessed OwlReady2 for ontology operations, along with the SPARQL query language for efficient extraction of content from the ontology. Additionally, we integrated Camel-tools to facilitate natural language processing specifically tailored for the Arabic language. To manage the database, we employed SQLite as the underlying platform.

This comprehensive IT solution is poised to significantly enhance the accessibility, searchability, and overall efficiency of the legal database, thereby empowering administrators to navigate and utilize the extensive corpus of legal information with greater ease and effectiveness.

Keywords: Legal texts, Algeria , Information search, Natural language processing (Arabic), Ontology.

1. Résumé

Les lois et les réglementations sont sujets à des modifications apportées par des experts du domaine en réponse aux évolutions dans la vie humaine. En prenant notre pays comme exemple, on constate une profonde refonte et une mise à jour régulière des lois au cours des dernières années. Cela a conduit à la création d'une vaste base de données d'informations juridiques. Cependant, la recherche manuelle dans une telle base de données est chronophage et a un impact sur l'efficacité des administrateurs.

Pour remédier à cette problématique, nous avons élaboré une solution informatique reposant sur une ontologie. Nous avons fait usage de la modélisation UML pour structurer les connaissances sous-jacentes. Après avoir édité notre ontologie à l'aide de Protégé, nous avons sélectionné un ensemble d'outils pour le développement de notre système. Nous avons opté pour l'environnement de développement Visual Studio Code conjointement avec le langage de programmation Python.

En ce qui concerne l'exploitation du système, nous avons mis en œuvre OwlReady2 pour les opérations liées à l'ontologie, ainsi que le langage de requêtes SPARQL pour extraire efficacement le contenu de celle-ci. Par ailleurs, nous avons intégré Camel-tools pour le traitement du langage naturel, notamment pour la langue arabe. Enfin, nous avons utilisé SQLite comme base de données pour l'extraction des contenus réglementaires.

Cette solution informatique globale vise à améliorer considérablement l'accessibilité, la recherche et l'efficacité globale de la base de données juridiques, permettant ainsi aux administrateurs de naviguer et d'utiliser plus facilement et efficacement le vaste corpus d'informations juridiques à leur disposition.

Mot clés: Textes juridiques, Algérie, Recherche d'information, Traitement du langage naturel (L'Arab), ontologie.

2. الملخص

تتم تعديل القوانين واللوائح من قبل الخبراء في المجال استجابةً للتغيرات المستمرة في حياة الإنسان. إذا أخذنا بلدنا كمثال، نجد أن هناك تعديلات وتحديثات كبيرة تتم في القوانين خلال السنوات القليلة الماضية، مما أدى إلى إنشاء قاعدة بيانات ضخمة من المعلومات القانونية. ومن المعروف أن البحث اليدوي عن المعلومات في مثل هذه القاعدة يتسبب في استهلاك الوقت ويؤثر على كفاءة الإداريين.

لحل هذه المشكلة، قدمنا حلاً تكنولوجياً. اخترنا استخدام الأنطولوجيا كأساس للحلول المقترحة. لتصميم الهيكل المعرفي، استخدمنا لغة نمذجة UML. بعد تحرير الأنطولوجيا باستخدام برنامج Protégé، اخترنا مجموعة من الأدوات لتطوير النظام. اعتمدنا على بيئة تطوير Visual Studio Code بالاشتراك مع لغة البرمجة Python.

Abstract

في عمليات التشغيل، اعتمدنا على حزمة OwlReady2 لإجراء العمليات المتعلقة بالأنطولوجيا، بالإضافة إلى لغة الاستعلامات SPARQL لاستخراج المحتوى منها بكفاءة. كما استخدمنا حزمة Camel-tools لمعالجة اللغة الطبيعية، خاصة باللغة العربية. وأخيراً، استخدمنا قاعدة بيانات SQLite لاستخراج محتويات اللوائح.

يهدف هذا الحل التكنولوجي الشامل إلى تحسين الوصول وسهولة البحث والكفاءة العامة لقاعدة البيانات القانونية، مما يمكن المسؤولين من تصفح واستخدام الكم الهائل من المعلومات القانونية بسهولة وفعالية أكبر.

كلمات مفتاحية: النصوص القانونية، الجزائر، البحث عن المعلومات، معالجة اللغة الطبيعية (اللغة العربية)، الأنطولوجيا.

Contents

List of figures

List of tables

GENERAL INTRODUCTION

Following the absence of knowledge bases in the field of Algerian regulation, we thought of setting up a platform in this field, with the aim of facilitating the tasks and the fast access to information by the administrators and the specialists of the field. Due to the nature of the regulatory domain which is very changeable from one day to another, the application of an artificial intelligence approach specifically an ontology of the domain is considered as a primordial solution, by benefiting from the advantages that characterize this approach.

The main objective of our work is to propose an intelligent system reliable, powerful, adaptable with all the changes carried out in this field and which answers the requirement of the administrative personnel and expert of the field of the Algerian regulation.

To get to this stage, five chapters have been developed as follows:

The first chapter will be a presentation of the field of regulation. We will give the definitions of the different concepts of this field as: a decree, regulation, regulatory text, approaches for the research of information,...

The second chapter NLP is essential for parsing user queries. It involves tasks like entity recognition (identifying names, dates, locations, legal terms), syntactic parsing (analysing sentence structure).

The third chapter is reserved to present the approach used to solve our problem. In this sense we present the components of an ontology, the process of development of an ontology, the tools and methodologies of development of ontologies...

In the fourth chapter will be dedicated to the design of our system. We will start by presenting the chosen method and the approach used for the indexing of regulatory documents.

In The five chapter, we present the implementation procedure of our application. We will start by presenting the tools used. We will present the Visual Studio Code development environment with the Python programming language, the OWLReady inference engine, the SPAQL query language.

Finally, we will present screenshots of the different steps of our application.

Chapter 01: Background

CHAPTER 01

Background

1. Information retrieval system

We will provide an overview related to the processing have been made. We begin by providing definitions of the terms necessary to understand the field.

We will also discuss some of the characteristics of our search to distinguish it from other types of inputs that are commonly analyzed and managed in this model to engineering steps for results. The latter must be updated continuously in order to adapt to humanitarian change.

1.1. Information system

1.1.1. Definition

An information system is a structured and organized collection of people, processes, data, and technology designed to gather, process, store, and disseminate information for a specific purpose within an organization or a broader context. It involves the acquisition, storage, manipulation, and distribution of data in a way that supports decision-making and helps achieve organizational goals. It can be found in various domains, including business, healthcare, education, government, and many others. [1]

1.1.2. Typical components of information systems

Information systems can vary widely in complexity and scope. They can range from simple manual systems, like a paper-based filing system, to highly sophisticated computer-based systems that utilize advanced software, databases, and network infrastructure.

a- Software

These are the programs and applications that enable the manipulation, processing, and presentation of data. This includes operating systems, databases, and application software.

b- Data

Raw facts and figures that are collected and stored by the system. Data can be structured (organized in a specific format) or unstructured (not organized in a specific manner).

c- Procedures

These are the set of instructions or protocols that govern how data is collected, processed, stored, and disseminated within the system. Procedures help ensure consistency and efficiency in handling information.

d- Communication Networks

These facilitate the transmission of data and information within the system, connecting various hardware components and allowing them to communicate with each other.

e- Feedback and Control:

This component involves mechanisms to monitor the performance of the information system and make necessary adjustments to ensure it meets its objectives effectively.

1.2. Information retrieval

1.2.1. Definition

Information or document retrieval consists of identifying the most relevant items with respect to a given query. Most of the approaches proposed in the literature are adaptations of traditional models. [2]

Adding semantics to document retrieval is an important area of research ***nowadays.***

1.2.2. The approaches proposed in Information search

.The majority of suggested methods for document retrieval depend on indexing systems that utilize keywords or terms.

- These approaches solely consider the frequency of term occurrences within documents or their components, thereby neglecting the semantic meaning of the words.
- Enhancing the performance of document information retrieval (IR) systems can be achieved by incorporating the semantics of indexing terms. This form of indexing transitions from individual words to conceptual levels in order to provide a more comprehensive description of document content and search queries.
- These methods utilize semantic resources during both the indexing and retrieval stages. When employing an Information Retrieval System (IRS), it is possible for extensive search results to include documents that are not pertinent to the specific queries. In this context, ontologies present a promising solution. (Figure 2).

Information search in the context of artificial intelligence can encompass various techniques and strategies for finding and retrieving relevant information from a dataset, database, or the internet. Here are some common approaches and techniques used in information search:

Keyword-based Search:

This is the most common method where users input keywords or phrases, and the system retrieves documents containing those terms.

Search engines like Google use complex algorithms to rank and retrieve relevant web pages. [3]

➢ When it comes to planning digital marketing strategies, the selection of keywords for search engine optimization plays a pivotal role in attracting customers, and the investment in this aspect is becoming a more significant component of the marketing budget. To what degree can the allocation of resources for improving a brand's search engine positioning be fine-tuned over time? Frequently, the expenses associated with keywords are perceived as financial costs, and research in keyword auctions indicates a tendency toward a stable, long-term expenditure trajectory. "Nonetheless, search engine optimization techniques, which come with implicit costs, have predominantly emphasized short-term outcomes. The extraction of long-term insights from the estimated cost per click is suggested and seen as the economic expense associated with organic keywords."

- Web analytics software is used to gather these data for leading fashion e-commerce. A time series test of transition dynamics identifies convergence of economic keyword costs between branded and generic keywords or catch-up, as an early state of convergence, for almost all considered e-commerce.

Web analytics software is used to gather these data for leading fashion e-commerce. A time series test of transition dynamics identifies convergence of economic keyword costs between branded and generic keywords or catch-up, as an early state of convergence, for almost all considered e-commerce.

Information Retrieval Models:

These are mathematical models used to rank documents by their relevance to a query.

Common models include TF-IDF (Term Frequency-Inverse Document Frequency) and BM25.[4]

When it comes to planning digital marketing strategies, the selection of keywords for search engine optimization plays a pivotal role in attracting customers, and the investment in this aspect is becoming a more significant component of the marketing budget. To what degree can the allocation of resources for improving a brand's search engine positioning be fine-tuned over time? Frequently, the expenses associated with keywords are perceived as financial costs, and research in keyword auctions indicates a tendency toward a stable, long-term expenditure trajectory.

Machine Learning Models for Ranking:

Supervised learning techniques can be used to train models to rank documents based on relevance.

This involves providing labeled data (relevant vs. non-relevant documents) for training. [5]

The major focus of the study is the task of providing users with related query suggestions after they have initiated a search on a web search engine. Our proposed approach leverages machine learning to estimate the likelihood that a user would find a subsequent query useful and relevant based on their initial query. This approach is built on a machine learning model that allows us to make predictions for queries that haven't been previously seen in the search logs. We train the model using co-occurrence data from the search logs, utilizing innovative utility and relevance models. Notably, the machine learning process doesn't require labeled data from human judges. By learning from past observations, our model can generate query suggestions that go beyond queries that have co-occurred in the past. This results in a significant increase in query coverage, with only modest improvements in relevance. Both offline evaluations, involving human judges, and online evaluations, based on millions of user interactions, confirm that our approach outperforms strong baseline methods..

Semantic Search:

This approach goes beyond keyword matching and tries to understand the context and meaning of a query.

Techniques like word embedding's or contextual embedding's (e.g., BERT) can be used for this purpose. [6]

A broad range of approaches to semantic document retrieval has been developed in the context of the Semantic Web. This survey builds bridges among them. We introduce a classification scheme for semantic search engines and clarify terminology. We present an overview of ten selected approaches and compare them by means of our classification criteria. Based on this comparison, the identification of not only common concepts and outstanding features, but also open issues is done. Finally, we give directions for future application development and research.

Federated Search:

In cases where information is scattered across multiple sources or databases, federated search allows querying multiple sources simultaneously.

Relevance Feedback:

This interactive technique involves the user providing feedback on initial search results to refine subsequent searches.

Concept-based Search:

Instead of relying on specific keywords, this approach involves searching for concepts or ideas, which may be represented in different ways in documents.

Meta-search Engines:

These are search engines that send user queries to multiple search engines and aggregate the results.

Deep Learning for Information Retrieval:

Deep learning models like Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) can be used for tasks like document classification or ranking.

Graph-based Search:

In some cases, representing information as a graph (nodes and edges) can be useful for performing complex searches, especially in knowledge graphs.

Personalization and Context-aware Search:

Consideration of user preferences, location, and historical behavior can be used to tailor search results to individual users.

Remember, the effectiveness of these approaches depends on various factors including the type of data, the nature of the search task, and the available resources. Therefore, the choice of approach should be based on the specific requirements of the information search task at hand.

1.2.3. General framework

The general framework to be considered is shown in the following figure [01].

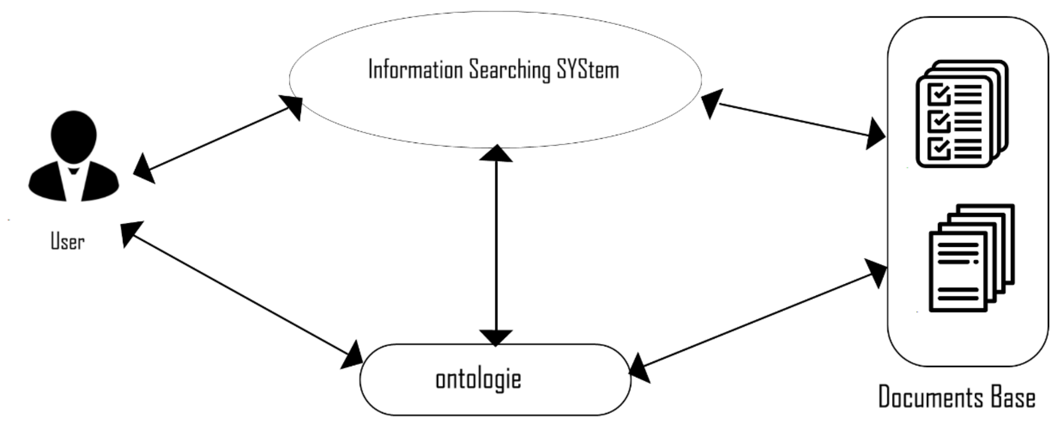


Figure 01: General of the framework study. [7]

- **A** user expresses his need for information by means of a query.
- The Information Retrieval System matches the user's query against an index of documents, to provide the user with a set of documents that meet their expectations.

1.2.4. Conceptual indexing

One of the problems of the current indexing of documents by words, or groups of words, is that it does not take into account the language phenomena of synonymy and homonymy:

- When different words refer to the same meaning, the indexer generally favors one of these words, the one that appears in the document. Assuming that the user uses another word in his query. As a result, the user does not have access to documents dealing with this notion. Synonymy creates documentary silence.
- When identical words have different meanings, the opposite effect occurs. The user's use of the word results in a response containing documents dealing with all of these concepts ... even if he is only interested in one of these concepts. Homonymy creates document noise. To overcome these problems, the solution consists in indexing documents by concepts, while keeping the links: concept/term(s). To show the contribution of a conceptual indexing, we distinguish, on the one hand, the exploitation of the links: concept/term(s), and on the other hand, the exploitation of the semantic relations structuring the ontology

2. Natural language processing (NLP)

Natural Language Processing (NLP) represents a dynamic intersection of artificial intelligence and linguistics, aiming to empower machines with the ability to comprehend, analyze, and generate human language. This multidisciplinary field encompasses a diverse range of algorithms and techniques that enable computers to process and understand text and speech data in a manner that is both contextually relevant and valuable. [8]

2.1. Core Tasks in NLP [9]

2.1.1. Text Classification

Text classification, a fundamental task within NLP, involves categorizing textual data into predefined classes or labels. Applications range from sentiment analysis in social media to email categorization for spam filtering.

2.1.2. Named Entity Recognition (NER)

Named Entity Recognition is the process of identifying and categorizing specific entities mentioned in text, such as names of people, places, organizations, and more. This task is pivotal in applications like information extraction and entity linking.

2.1.3. Sentiment Analysis

Sentiment Analysis entails determining the emotional tone or sentiment expressed in a piece of text, classifying it as positive, negative, or neutral. This capability is invaluable for applications like customer feedback analysis and brand perception monitoring.

2.1.4. Machine Translation

Machine Translation involves automatically converting text from one language to another, fostering cross-lingual communication and accessibility to a global audience.

2.1.5. Text Summarization

Text Summarization is the art of generating concise and coherent summaries of longer textual content, providing readers with distilled versions of complex information.

2.1.6. Speech Recognition

Speech Recognition technology enables machines to transcribe spoken language into written text, revolutionizing voice-operated interfaces and accessibility for diverse user groups.

2.1.7. Question Answering

Question Answering systems aim to provide accurate and contextually relevant responses to user-posed questions, ranging from fact-based queries to more complex inquiries.

2.2. Applications of NLP

NLP is applied in various real-world applications, ranging from chatbots and virtual assistants to language translation, sentiment analysis in social media, email filtering, and more. [9]

2.2.1. Customer Support Chatbots

NLP-powered chatbots revolutionize customer support by offering automated, natural language-based interactions, providing timely assistance and information retrieval.

2.2.2. Language Translation Services

NLP plays a pivotal role in enabling seamless language translation, breaking down global communication barriers and fostering international collaboration.

2.2.3. Social Media Analysis

Sentiment analysis and topic modelling in social media data offer valuable insights into public opinions, trends, and brand perception, informing marketing and PR strategies.

2.2.4. *Healthcare Informatics*

NLP facilitates the extraction of structured information from unstructured clinical notes, aiding in diagnoses, treatment planning, and medical research.

2.2.5. *Virtual Assistants*

NLP underpins the capabilities of virtual assistants like Siri, Google Assistant, and Alexa, enabling them to comprehend and respond to user commands and queries.

2.3. Future Trends and Challenges

As NLP continues to evolve, emerging technologies like transformers and deep learning architectures are poised to revolutionize the field. However, challenges such as bias in language models and ethical considerations remain focal points for research and development, including ambiguity in language, understanding context, handling slang or colloquial expressions, and dealing with multiple languages. [10]

Natural Language Processing (NLP) is a subfield of Artificial Intelligence (AI) that focuses on enabling computers to understand, interpret, and generate human language in a way that is valuable and meaningful. Here's a breakdown of the relationship between NLP and AI:

2.3.1. *NLP as a Subfield of AI*

AI refers to the ability of machines to perform tasks that typically require human intelligence. This includes tasks like learning, reasoning, problem-solving, understanding natural language, and more. [11]

NLP is a specialized area within the broader field of AI. It deals specifically with the interaction between computers and human language.

2.3.2. *Language Understanding*

NLP focuses on the development of algorithms and models that allow computers to understand, interpret, and generate human language. This includes tasks such as sentiment analysis, language translation, named entity recognition, and more.

2.3.3. NLP and Machine Learning

NLP heavily relies on machine learning techniques, which is a subset of AI. Machine learning algorithms learn patterns from data to perform tasks without being explicitly programmed. In NLP, this involves training models on large datasets of text to understand language.

2.3.4. NLP and Deep Learning

Deep learning, a subset of machine learning, is particularly effective in NLP tasks. Deep learning models, such as recurrent neural networks (RNNs) and transformers, have revolutionized the field by enabling the processing of large amounts of sequential data like text.

2.3.5. Semantic Understanding

NLP aims to enable computers to not just understand the syntax of language but also the semantics, or meaning, behind it. This involves tasks like understanding context, identifying relationships between words, and more.

2.3.6. AI Beyond NLP

AI encompasses a broader range of capabilities beyond NLP. This includes computer vision (for processing and understanding visual information), robotics, game playing, decision-making, and more.

2.3.7. Synergy and Integration

NLP is often integrated with other AI techniques and technologies to create more comprehensive AI systems. For example, a virtual assistant may combine NLP for natural language interaction with computer vision for visual understanding.

3. The ontologies

For problems, for which only linguistic knowledge is available, formal modelling plays a crucial role. Currently, ontologies are a strategic issue in the representation and modelling of knowledge. They define the necessary primitives of a domain and their semantics in a particular context. In this chapter, we will describe what an ontology is, as well as the different elements that make it up and the steps necessary for its construction. We will then move on to the methods, methodologies and different tools for developing ontologies. We will then present the main ontology description languages. Finally, we will end with a conclusion.

3.1. Notion of ontology

Ontology is seen as a set of concepts allowing to model a set of knowledge in a given domain. a concept can have several thematic meanings. concepts are linked together by semantic, composition and inheritance relations. a general definition has been given by thomasr. gruber where he describes an ontology as "an explicit specification of a conceptualization" modeling concepts and the relations between relationships between concepts. Other authors later added the word formal and the word shared. Thus, the most famous definition is "An ontology is a formal and explicit specification of a shared conceptualization". These ontologies can also be used for knowledge reuse and sharing. [12]

3.1.1. Definitions from philosophy

In the field of philosophy, ontology is considered as a branch of metaphysics that is concerned with the nature and organization of reality. It has a broader meaning, that of "science of what exists" in which one does not seek to explain the world, but to represent it. It applies to "being as being" [13]

3.1.2. Definitions from artificial intelligence

Artificial intelligence has made it possible to represent the knowledge of a domain in the form of a base, known as a knowledge base, and to automate its use and

the resolution of problems around it, through data inference. However, knowledge bases are, on the whole, not reusable, which limits their use. The notion of ontology has been introduced, among others, to overcome this limitation.

Generally speaking, an ontology is seen as a set of concepts that make it possible to model a body of knowledge in a given domain. A concept can have several thematic meanings. Concepts are linked together by semantic, compositional and inheritance relationships. In order to clarify this notion, many researchers have proposed definitions that it is useful to examine:

- Thomas R. Gruber has given a general definition where he describes an ontology as an explicit specification of a conceptualization modeling concepts and the relationships between concepts. Thomas R. Gruber defends this view as follows: "An ontology is a specification of a conceptualization. That is, an ontology is a description (like a formal specification of a program) of the concepts and relationships that may exist for an agent or a community of agents. This definition is consistent with the use of ontology as a set of concept definitions, but it is more general.
- Later, John F. Sowa specified this notion more precisely. In his definition, ontology is seen as a catalogue of types derived from the study of categories of abstract and concrete entities that exist or can exist in a domain. It is defined as follows: "The subject of ontology is the study of the categories of things that exist or can exist in a domain. The product of such a study, called an ontology, is a catalogue of the types of things that are supposed to exist in a domain of interest D from the point of view of a person who uses a language L to talk about D. The types of the ontology represent the predicates, word meanings, or types of concepts and relations of the language L when it is used to discuss topics in the domain D.
- Afterwards, in order to complete the original philosophical meaning, Guarinoa introduces the notion of formal ontology, which is defined as a conceptual modelling, or a representation of such modelling. "An ontology is an agreement on a shared and possibly partial conceptualization.
- Similarly, Uschold defines an ontology as a formal description of entities and their properties, relations, constraints and behaviours. Furthermore, the authors introduced the notion of explicit ontology "An explicit ontologymaytake a variety of forms, but necessarily it will include a vocabulary of terms and some specification of their meaning".
- Finally, Christophe Roche gave a generic and simple definition "An ontology is a conceptualization of a domain to which one or more vocabularies of terms are The definition of an ontology is retained because it encompasses and summarizes the previous explanations. In the following sections, the elements that constitute the ontology will be detailed in a non-comprehensive manner. [14]

3.2. The components of ontology

An ontology consists of the following elements:

3.2.1. Concepts

A concept represents a set of objects and their common properties described by a term, a concept can represent an object, an idea, or an abstract notion. They are also called ontology classes, a concept is designated by 3 parts:

1. **The term:** (or label) of a concept is the linguistic expression commonly used to refer to it. To refer to it.
2. **Meaning or notion:** this is the intentional definition using the set of attributes, common properties that a concept attributes, common properties that a concept encompasses.
3. **Denoted objects (the concept extension):** exhaustive description of everything that obeys the definition, i.e. all the beings that a concept encompasses, all its instances.

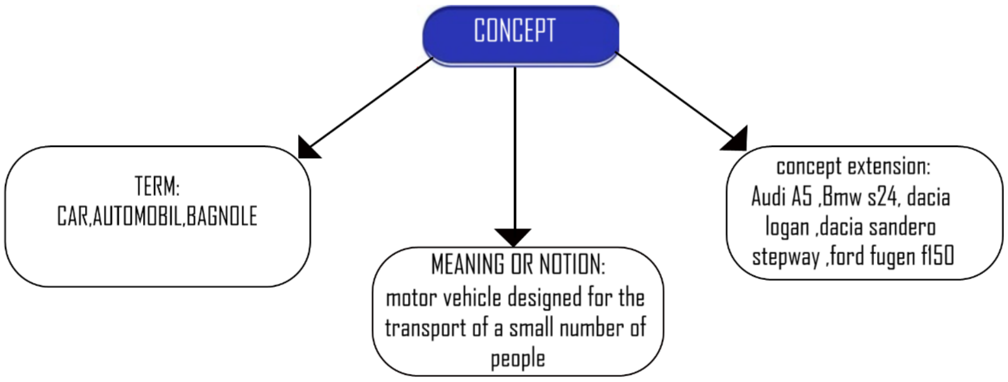

CONCEPT
TERM: CAR,AUTOMOBIL,BAGNOLE
MEANING OR NOTION: motor vehicle designed for the transport of a small number of people
concept extension: Audi A5 ,Bmw s24, dacia logan ,dacia sandero stepway ,ford fugen f150

Figure 02: example of a concept

3.2.2. Relation

A notion of association or link between concepts, usually expressed by a term or by a literal or other symbol.

3.2.3. Axioms

which constitute assertions, accepted as truths, concerning abstractions translated from this domain by ntology.

3.2.4. Individuals

Individuals (instances) are the basic, "ground level" components of an ontology. The individuals in an ontology may include concrete objects such as people, animals, tables, automobiles, molecules, and planets, as well as abstract individuals such as numbers and words (although there are differences of opinion as to whether numbers and words are classes or individuals). Strictly speaking, an ontology need not include any individuals, but one of the general purposes of an ontology is to provide a means of classifying individuals, even if those individuals are not explicitly part of the ontology.[15]

3.3. The ontology development process

The process of building an ontology (Figure 2) is a collaboration that brings together knowledge domain experts, knowledge engineers, and even the future users of the ontology. This collaboration can only be useful if the objectives of the process have been clearly defined, as well as the resulting requirements.

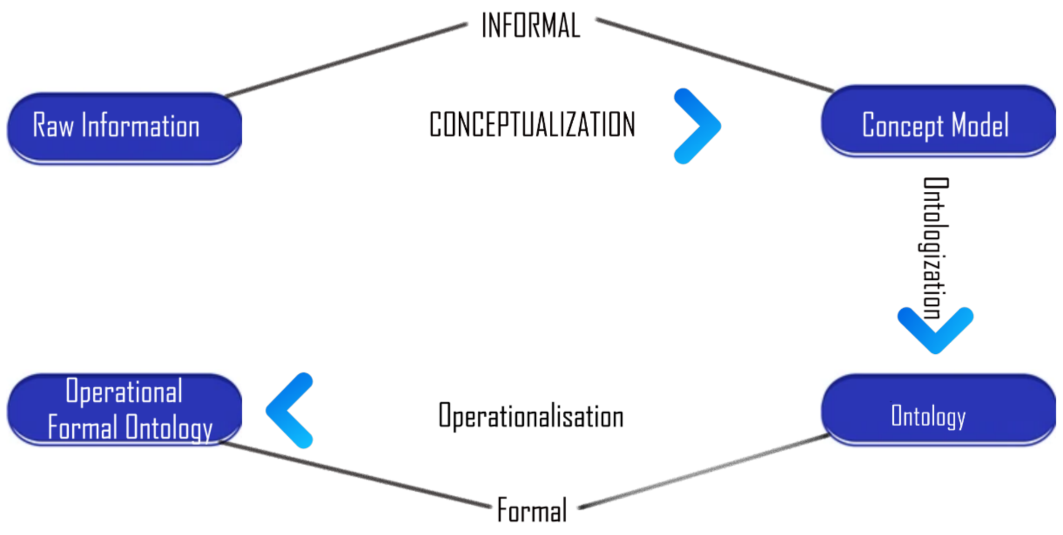


Figure 03: Ontology building process

3.3.1. Needs assessment

- The operational objective: specify the operational objective of the ontology, in particular through usage scenarios.
- The domain of knowledge: to delimit precisely the domain of knowledge.
- Users: identify target users.

3.3.2. Conceptualisation

Based on both document analysis and interviews with experts in the field, this stage leads to an informal model, generally expressed in natural language. It consists of extracting from the raw data the concepts and the relationships between these concepts that describe the knowledge domain.

3.3.3. Ontologisation

Ontologisation consists of a partial formalisation, without loss of information, of the conceptual model obtained in the previous step. This facilitates its subsequent representation in a completely formal and operational language. It transcribes the knowledge into a certain knowledge formalism.

3.3.4. Operationalisation

This step consists of completely formalising the ontology obtained in a formal (i.e. with a syntax and semantics) and operational (i.e. with inferential services to implement reasoning) knowledge representation language, for example, the Conceptual Graphs model or Description Logic. A formal representation of the domain knowledge is then obtained. Thus, the formal character of the ontology allows a machine, via this ontology, to manipulate domain knowledge. The machine must therefore be able to use mechanisms operating on the ontology's representations.

3.4. Methods and methodologies for developing ontologies

Among the existing ones, we quote:

3.4.1. The "Ontology Development 101" method

"Ontology Development 101" was developed at Stanford University and seeks to build formal ontologies by reusing and adapting existing ontologies, and proposes the following approaches:

- Determine the domain and scope of the ontology
- Consider reuse of existing ontologies
- List the most important terms in the ontology
- Define classes and class hierarchy
- Define class properties

- Define attribute facets
- Build instances.
- It uses the Protégé and Ontolingua tools as support.

3.4.2. The On-To-Knowledge methodology

On-to-Knowledge is a methodology developed in the framework of a project whose partners are the AIFB Institute of the University of Kalsruhe, the Free University of Amsterdam, and the British Telecom company. On-to-Knowledge recommends an iterative development process, with four main phases a requirement specification phase, an improvement phase, an evaluation phase and an application and evolution phase. On-To-Knowledge proposes the acquisition of knowledge by specialising a generic ontology. It proposes to build the ontology taking into account how it will be used in other applications. Therefore, ontologies developed with this methodology are highly application dependent.[16]

3.5. Ontology development tools

There are many questions to ask when choosing a development tool:

Does the tool offer development assistance? Does the tool have an inference engine? What ontology languages does the tool support? Does the tool allow importing/exporting ontologies? Does the tool support the reuse of existing ontologies? Does the tool allow for the documentation of constructed ontologies? Does the tool offer graphical support for ontology construction? Is the tool stable, user-friendly, "mature"? The answers to all these questions could be decisive in the choice of one or other tool. Several tools exist on the market. it can be mentioned:

3.5.1. Protégé

This is an ontology editor distributed as open source by Stanford University of Medical Informatics. Protégé is a tool used by developers and domain experts to develop knowledge-based systems. Applications developed with Protégé are used in problem solving and decision making in a particular domain. Protégé also allows the creation or import of ontologies written in different ontology languages such as: RDF-Schema, OWL, DAML, OIL, ...etc. This is made possible through the use of plugins that are available for download for most of these languages.

3.5.2. OntoEdit and its suite OntoStudio

OntoStudio is the sequel to its predecessor OntoEdit. It was developed at the University of Karlsruhe in Germany. OntoStudio is an ontology editor for the Semantic Web, available in freeware and professional (paid) versions. It is a tool that proposes the On-To-Knowledge methodology [17]

3.5.3. Ontolingua Created at Stanford University

the Ontolingua server is the best known ontology building environment in the Ontolingua language. Ontolingua is a language based on Kif and Frame Ontology. It consists of a set of tools and services that support the cooperative construction of ontologies between geographically separated groups.

3.5.4. OilEd (Oil Editor)

OilEd is an ontology editor using the OIL formalism. It is primarily dedicated to the construction of small ontologies which can then be tested for consistency using FACT, an inference engine built on OIL.

3.6. Ontology specification languages

Several ontology specification languages (or ontology languages) have been developed during the last few years, and they will surely become ontology languages in the context of the Semantic Web. Some of them are based on the syntax of XML, such as XOL (Ontology Exchange Language), SHOE (Simple HTML Ontology Extension - which was previously based on HTML), OML (Ontology Markup Language), RDF (Resource Description Framework), RDF Schema. The last two are languages created by working groups of the World Wide Web Consortium (W3C).

Finally, three additional languages are built on top of RDF(S) to improve its features: OIL (Ontology Inference Layer), DAML+OIL and OWL (Web Ontology Language).

The figure below presents ontology specification languages, which have been recently developed. The figure below represents the main relationships between all these languages in the form of a pyramid of Semantic Web languages.

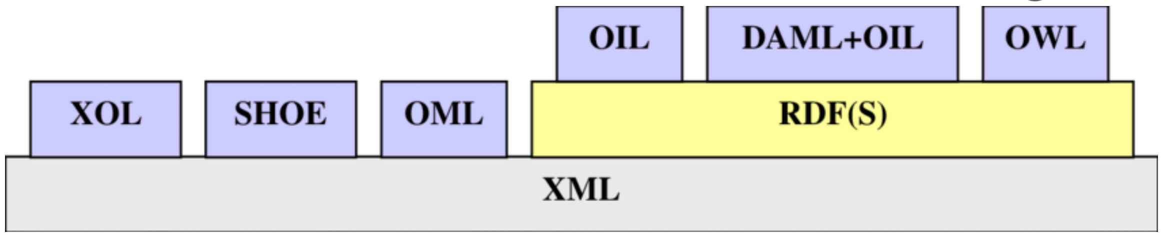


Figure 04: the web base language pyramid

a- RDF

RDF is an assertion and annotation language. Assertions assert the existence of relationships between objects. They are therefore suitable for expressing the annotations that can be tassociated with Web resources. RDF is a formal language for asserting relationships between "resources". The RDF model defines three types of objects:

4. **Resources:** They are all objects described by RDF. Generally, these resources can be web pages as well as any real-world object or person. Resources are then identified by their URI (Uniform Resource Identifier);
5. **Properties:** A property is an attribute, an aspect, a characteristic that applies to a resource. It can also be a link to another resource;
6. **Values:** The values in question are the particular values that the properties take.

These three types of objects can be related by assertions, i.e. triplets (resource, property, value), or (subject, predicate, object).

b- RDF(S)

As its name suggests, RDFS is intended to define metadata schemas. It defines the meaning, characteristics and relationships of a set of properties. The definition can include constraints for potential values and inheritance of properties from other schemas. It is, in effect, a semantic extension of RDF to provide a mechanism for describing associated groups of resources and the relationships between resources. The value of RDFS is that it facilitates inference about data and enhances research on that data.

c- DAML-OIL 1

DAML is a language that aims to provide the foundation for the next generation of the Semantic Web. The language first adopted RDFS as an ontology language for semantic interoperability between projects. As RDFS is not expressive enough for the requirements of the Semantic Web, a new language called

DAML-ONT was developed as an extension of RDF with the capabilities of a knowledge representation language: object-oriented and framework-based.

At the same time, a group of researchers (most of them European) within the European Union STI, with the same goal, is developing another ontology language called OIL. This language has a syntax based on RDF and is explicitly constructed so that its semantics can be specified through a very expressive logical description, the SHIQ-type description logic.

DAML+OIL is the combination of these two languages. It inherits the advantages of both languages. As a result, DAML+OIL is a very expressive and machine-readable language as well as human-readable with a syntax based on RDF.

d- OWL

OWL stands for Web Ontology Language. It is defined by the W3C in The OWL language is based on research in the field of description logic. OWL is used to describe ontologies, i.e. it allows the definition of terminologies to describe concrete domains. A terminology consists of concepts and properties (also called roles in description logics). A domain consists of consists of instances of concepts.

The OWL language consists of three sub-languages of increasing expressiveness, each designed for specific developer communities and users: OWL Lite, OWL DL, OWL Full. Each is an extension of its simpler predecessor. OWL Lite meets the needs of classification hierarchy and simple constraint functionalities of simple constraints of cardinality 0 or 1. A cardinality of 0 or 1 corresponds to functional functional relationships, for example, a person has an address. However, this person may. However, this person may have one or more first names, so OWL Lite is not sufficient for this situation.

OWL DL is for users who want maximum expressiveness coupled with completeness of the computation (this means that all inferences will be guaranteed to be taken into account) and (i.e. all inferences are guaranteed to be taken into account) and decidability of the reasoning system (i.e. all calculations will be completed in a finite time interval). This language includes all OWL with some restrictions, such as the separation of types: a class cannot also be an individual or a property. It is called DL because it corresponds to descriptive logic.

OWL Full is intended for people who want maximum expressiveness. It has the advantage of complete compatibility with RDF/RDFS, but the disadvantage of having a high level of descriptive capacity, even if this means that the completeness and decidability of ontology-related calculations cannot be guaranteed.[18]

3.7. SPARQL queries

Since version 0.30, Owlready proposes 2 methods for performing SPARQL queries: the native SPARQL engine and RDFlib.

3.7.1. Native SPARQL engine

The native SPARQL engine automatically translates SPARQL queries into SQL queries, and then run the SQL queries with SQLite3.

The native SPARQL engine has better performances than RDFlib (about 60 times faster when tested on Gene Ontology, but it highly depends on queries and data). It also has no dependencies and it has a much shorter start-up time.

However, it currently supports only a subset of SPARQL.

3.7.2. SPARQL elements supported

7. SELECT, INSERT and DELETE queries
8. UNION
9. OPTIONAL (with a single triple)
10. FILTER, BIND, FILTER EXISTS, FILTER NOT EXISTS
11. SELECT sub queries
12. VALUES in SELECT queries
13. All SPARQL functions and aggregation functions
14. Blank nodes notations with square bracket, e.g. ‘[a XXX]’
15. Parameters in queries (i.e. ‘??’)
16. Property path expressions, e.g. ‘a/rdfs:subClassOf*’, excepted those listed below

3.7.3. SPARQL elements not supported

17. ASK, DESCRIBE, LOAD, ADD, MOVE, COPY, CLEAR, DROP, CONSTRUCT queries
18. INSERT DATA, DELETE DATA, DELETE WHERE queries (may use INSERT or DELETE instead)
19. OPTIONAL (with more than one triple)
20. SERVICE (Federated queries)
21. GRAPH, FROM, FROM NAMED keywords
22. MINUS
23. Property path expressions with parentheses of the following forms:

- nested repeats, e.g. (a/p*)*
- sequence nested inside a repeat, e.g. (p1/p2)*
- negative property set nested inside a repeat, e.g. (!(p1 | p2))*

i.e. repeats cannot contain other repeats, sequences and negative property sets

3.7.4. Performing SPARQL queries

The .sparql() methods of the World object can be used to perform a SPARQL query and obtain the results. Notice that .sparql() returns a generator, so here the list() functionis used to show the results. The list contains one row for each result found, with one or more columns (depending on the query).

Notice that the following prefixes are automatically pre-defined:

```
rdf: -> http://www.w3.org/1999/02/22-rdf-syntax-ns#

rdfs: -> http://www.w3.org/2000/01/rdf-schema#

owl: -> http://www.w3.org/2002/07/owl#

xsd: -> http://www.w3.org/2001/XMLSchema#

obo: -> http://purl.obolibrary.org/obo/

owlready: -> http://www.lesfleursdunormal.fr/static/_downloads/owlready_ontology.owl#
```

In addition, Owlready automatically create prefixes from the last part of ontology IRI (without .owl extension), e.g. the ontology “http://test.org/onto.owl” with be automatically associated with the “onto:” prefix. Consequently, in most case we don’t need to define prefixes (but we can still define them if we want).

The classes counted above include OWL named classes, but also some OWL constructs. One may count only named classes using a FILTER condition and the ISIRI function, as follows.

4. Conclusion

After having presented the essentials of ontologies in this chapter, we will move on to the next chapter in which we will begin the construction of our ontology on regulatory texts and present our information retrieval system.

The next chapter will deal with the necessary elements of their development, i.e. the representation languages, the editing and interrogation tools, etc.

CHAPTER 02
Conception
of
Judicial Assistant System
(JAS)

1. Introduction

In this chapter, we will present the design steps of our regulatory information retrieval system. We will start with the design of our domain ontology, after having chosen a construction method among several existing in the literature.

What follows will be devoted to the conceptual description of our system exploiting the existing knowledge in our ontology.

2. Overview of the system

We want to create information system for legal texts using the Python programming language, Natural Language Processing (NLP) and ontology via web interface, so that it makes it easy for its users to index and search accurately in the huge and growing amount of laws issued by the Algerian legislator, through what NLP libraries provide in terms of understanding human language and creating new sentences that meet the purpose in terms of meaning and feelings. As well as using the power of ontology to store data, represent data, and facilitate access to it despite its complexity through SQLite.

3. System architecture

Architecture is a critical aspect of designing a system, as it sets the foundation for how the system will function and be built. It is the process of making high-level decisions about the organization of a system, including the selection of hardware and software components, the design of interfaces, and the overall system structure. Here we will discuss the **general architecture** of the system as well as its important **algorithms**.

3.1. Global architecture

Our architecture is web application. The user enters a query (text) and the system gives him the results, after analyses text using NLP and consulting ontology.

The regulatory database is two tables that contain the following properties: ontology file (owl) and its link where is located in the server and stopword list.

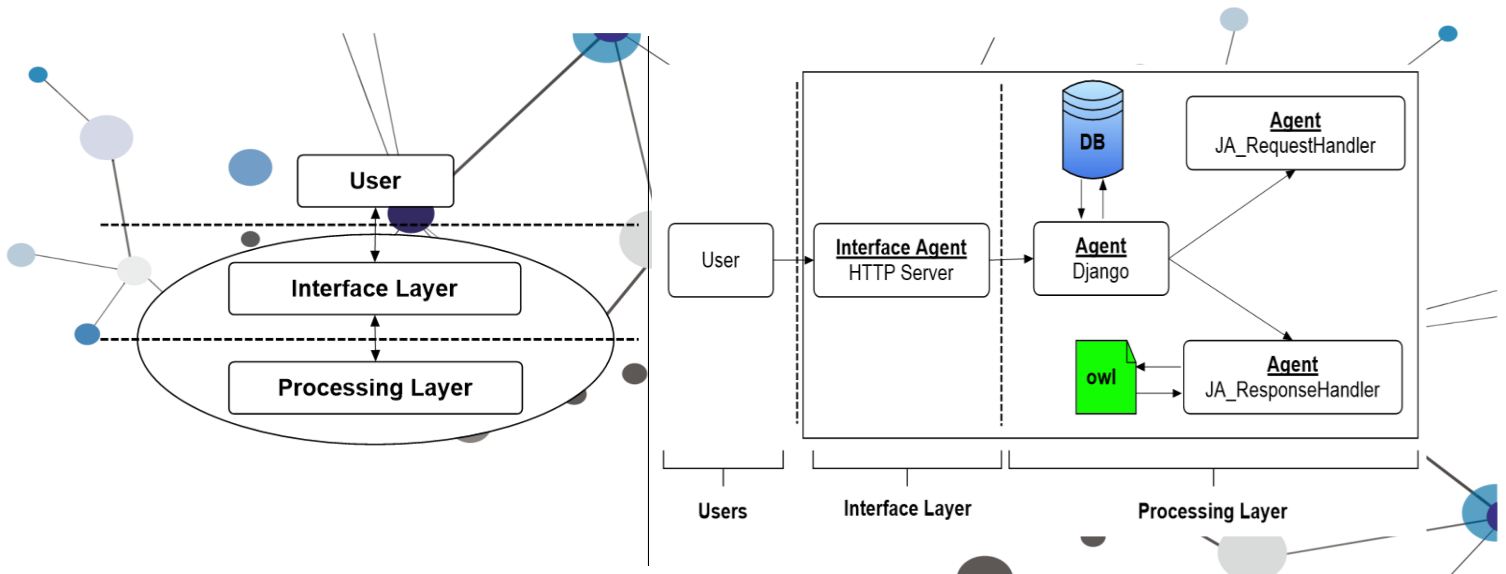


Figure 05: Global architecture of the system

3.2. Algorithms

3.2.1. Algorithms of JA_RequestHandler

a- Extract tokens

This algorithm extracts initial tokens from text entered via the system's web page.

```
Algorithm extract_tokens (string text):
    if text not empty then:

        mled = MLEDisambiguator.pretrained('calima-msa-r13');
        tokenizer = Morpho logicalTokenizer();

        return tokenizer.tokenize(text);
    endif;
end.
```

b- Handle tokens

This algorithm processes the initial tokens generated by the previous algorithm by removing unimportant search words and stop words.

```
Algorithm handle_tokens (list init_tokens):
     results = [];
     if tokens not empty then:
          stopwords = get_stopwords_from_db();
          for token in tokens do:
                if token in stopwords than:
                   ignore_it;
                else:
                   if "ة" in token than: #ة
                       remove_taa_marbota(results);
                   else:
                       if "+" in token then:
                            ignore_it;
                       else:
                            add(results, token);
                       endif;
                   endif;
               endif;
           endfor;
     endif;

     return results;
end.
```

c- Get synonyms

This algorithm processes the handled tokens generated by the previous algorithm by adding synonyms if exists.

```
Algorithm get_synonyms (list init_tokens):
      results = [];
      for token in tokens then:
           add(results, token);
           add(results, arabicLT.main.get_synonyms(token));
      endfor;
      return results;
end.
```

3.2.2. Algorithms of JA_ResponseHandler

a- Make response

This algorithm responsible to give answer to request of user via SQLite queries on ontology file named “`judicial_ontology.owl`”.

```
algorithm make_response(list tokens):
  with onto:

     q = make_sparql_query("""
     SELECT ?art ?art_title ?art_content (group_concat(?kw_name;separator="،") as
?kws) WHERE {
           ?art a jonto:Article ;
                jonto:art_title ?art_title ;
                jonto:art_content ?art_content .

      OPTIONAL { ?art jonto:has_keyword ?kw . ?kw jonto:named ?kw_name }
      FILTER (""" + " || ".join('("{a}" in ?art_content)' for a in tokens) + """)}
      GROUP BY ?art ?art_title ?art_content"""))
   endwith;
   return q
end.
```

4. Ontology conception

There are a multitude of ontology engineering methods. However, there is no consensus on the principles that should guide ontological modeling. Most of these methods aim to identify concepts and relationships from documents in the field, or from questions asked to experts.

The method followed is based on the one developed at Stanford University, as presented in the previous chapter, because it offers clear, straightforward, and easily understandable steps..

4.1. Construction of the domain ontology

- **Specification of needs**

Consists of determining the domain and scope of the ontology. The domain covered by our ontology is the domain of legal texts.

- **Conceptualization**

The construction of the conceptual ontology will be accomplished through expert interviews and the utilization of available documentation.:

4.2. Definition of classes

Classes	Description

Keyword	Keywords that exist in documents
Sector	The basic sectors of the state, such as the financial and agricultural sectors …etc.
Authority	State institutions.
OfficialJournal	Official Journal may refer to the public journal of several nations and other political organizations.
JudicialLaw	Laws
Article	Articles of laws
LegalRule	Collection of legal rule that composite JudicialLaw

Table 01: Definition of classes

4.3. Definition of the data properties and object properties

a- Data properties

Data property	**Class**	**Type**
About	LegalRule	string
art_content	Article	string
art_title	Article	string
Canceled	Article, LegalRule	boolean
Dated	OfficialJournal, LegalRule	date
Issue	OfficialJournal	integer
Named	Keyword, JudicialLaw, Sector, Authority	string
No	LegalRule	string

Table 02: Description of data properties

b- Object properties

Object property	**Domain**	**Range**
belong_to	Authority	Sector
Cancel	LegalRule	JudicialLaw, LegalRule, Article
canceled_by	JudicialLaw, LegalRule, Article	LegalRule
complete	LegalRule	LegalRule
completed_by	LegalRule	LegalRule
composed_by	JudicialLaw	Article
part_of	Article	JudicialLaw
Contain	LegalRule	JudicialLaw
contained_in	JudicialLaw	LegalRule
has_keyword	LegalRule, Article	Keyword
Modify	LegalRule	LegalRule

modified_by	LegalRule	LegalRule
Publish	Authority	LegalRule
published_by	LegalRule	Authority
published_in	LegalRule	JournalOfficial
referenced_to	LegalRule	LegalRule
referenced_by	LegalRule	LegalRule

Table 03: Description of object properties

5. System modeling using UML

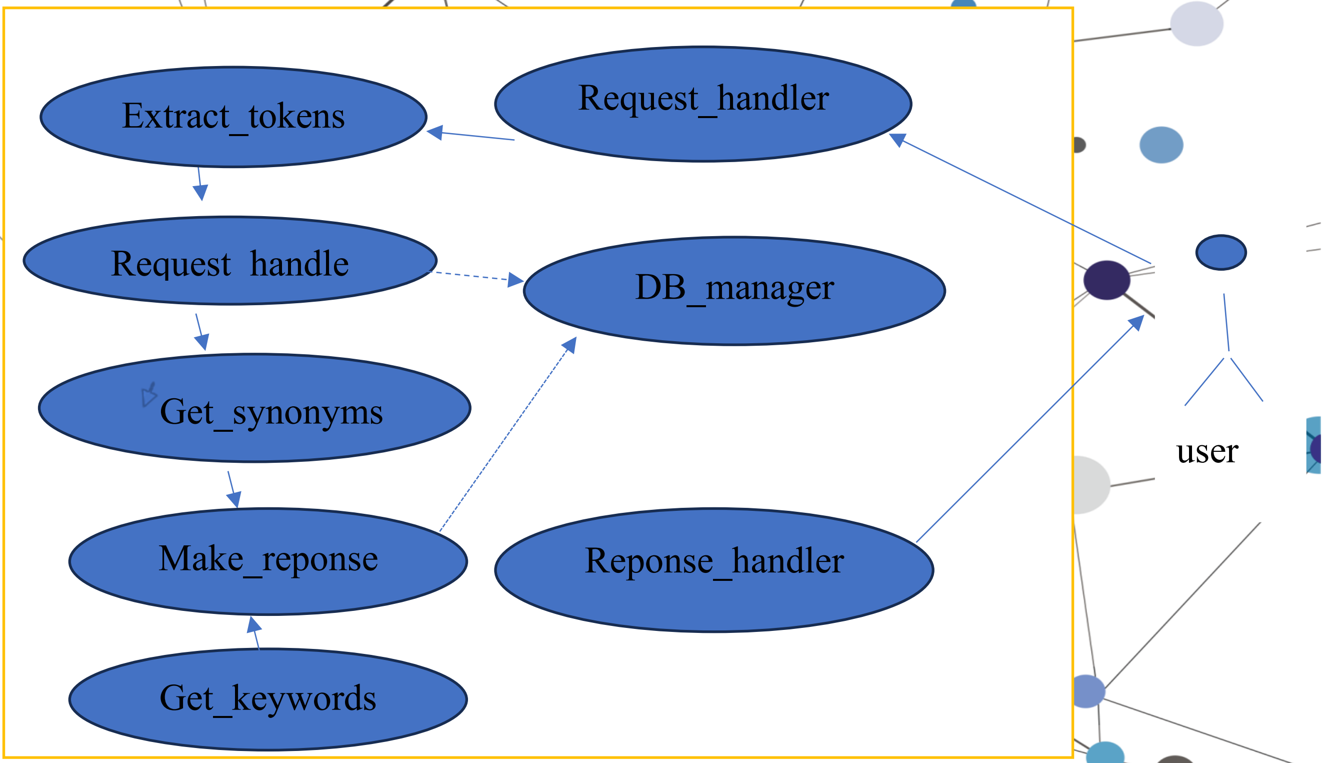
UML, short for Unified Modeling Language, is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing object oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

5.1. UML Diagrams

Among the diagrams proposed by UML, we use: the Use Case, Sequence, Class and Activity diagrams.

5.1.1. Use Case

A use-case model describes a system's functional requirements in terms of use cases. It represents the system's intended functionality (use cases) and its environment (actors). Use cases facilitate the connection between the requirements from a system and how the system fulfills those needs.



5.1.2. *Sequence diagram*

The Sequence Diagram models the collaboration of objects based on a time sequence. It shows how the objects interact with others in a particular scenario of a use case. We have two diagrams for User and Administrator.

a- User sequence diagram

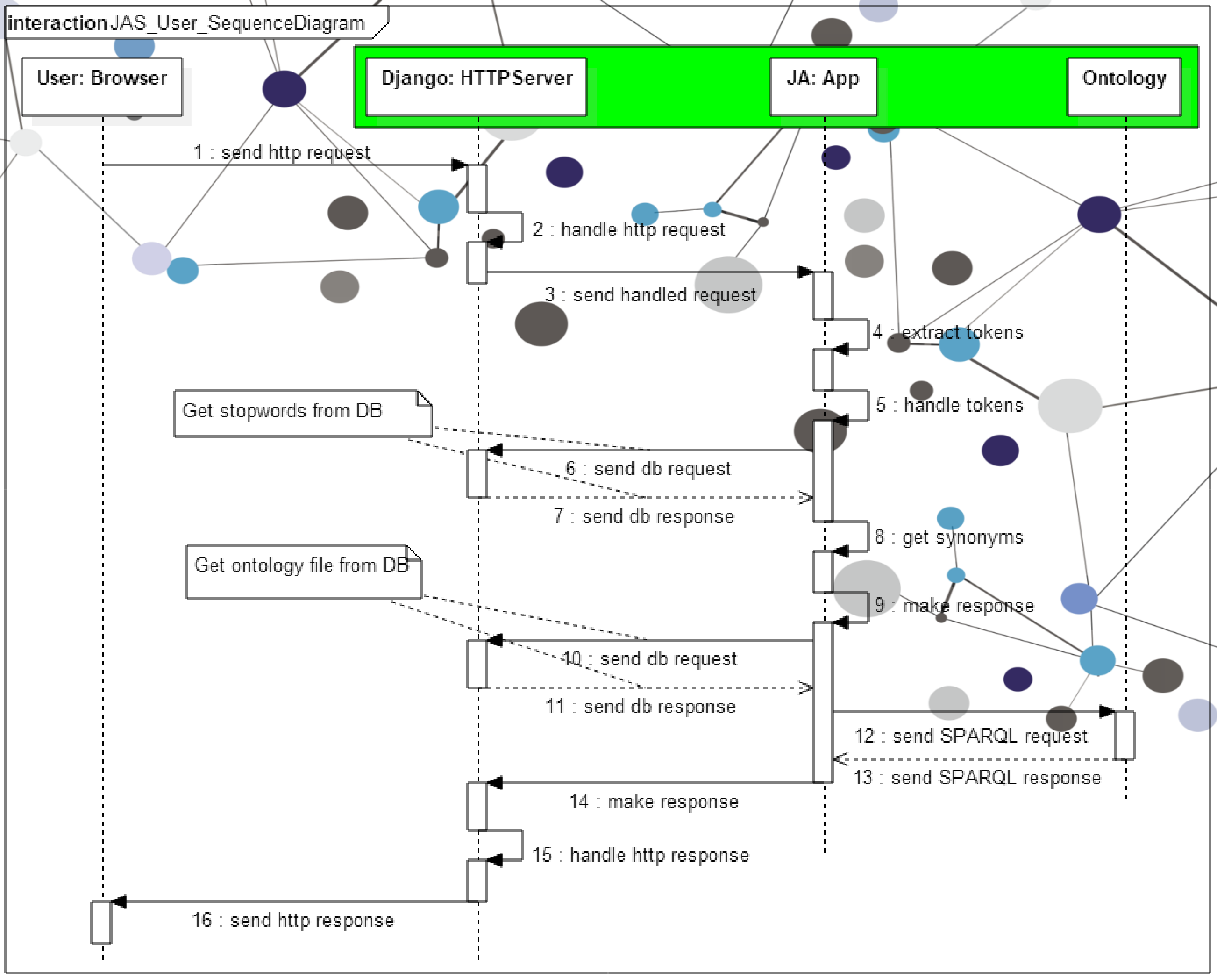


Figure 07: User Sequence Diagram of the system

b- Administrator sequence diagram

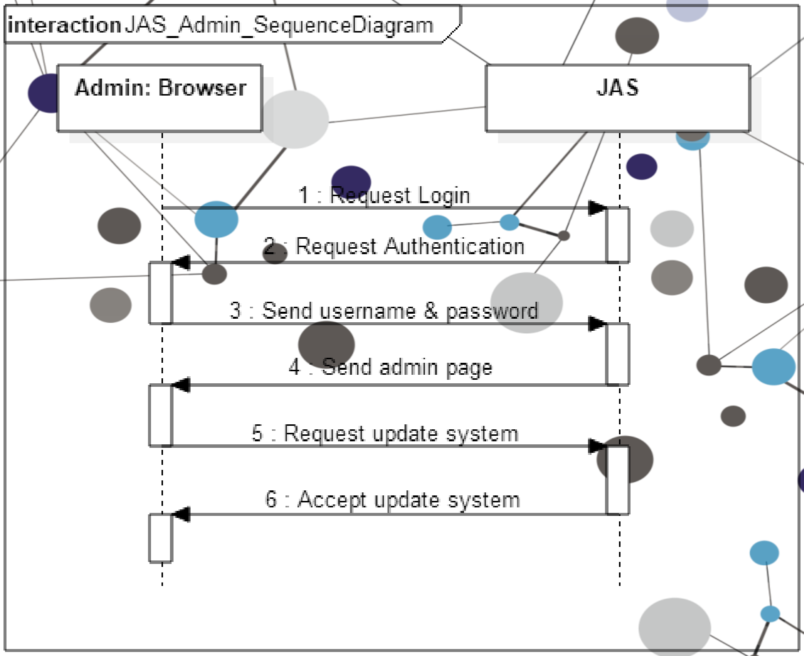

interaction JAS_Admin_SequenceDiagram
Admin: Browser
JAS
1 : Request Login
2 : Request Authentication
3 : Send username & password
4 : Send admin page
5 : Request update system
6 : Accept update system

Figure 08: Administrator Sequence Diagram of the system

5.1.3. Class diagram

The class diagram is a central modeling technique that runs through nearly all object-oriented methods. This diagram describes the types of objects in the system and various kinds of static relationships which exist between them. We have two diagrams: App class diagram and Ontology class diagram.

a- App class diagram

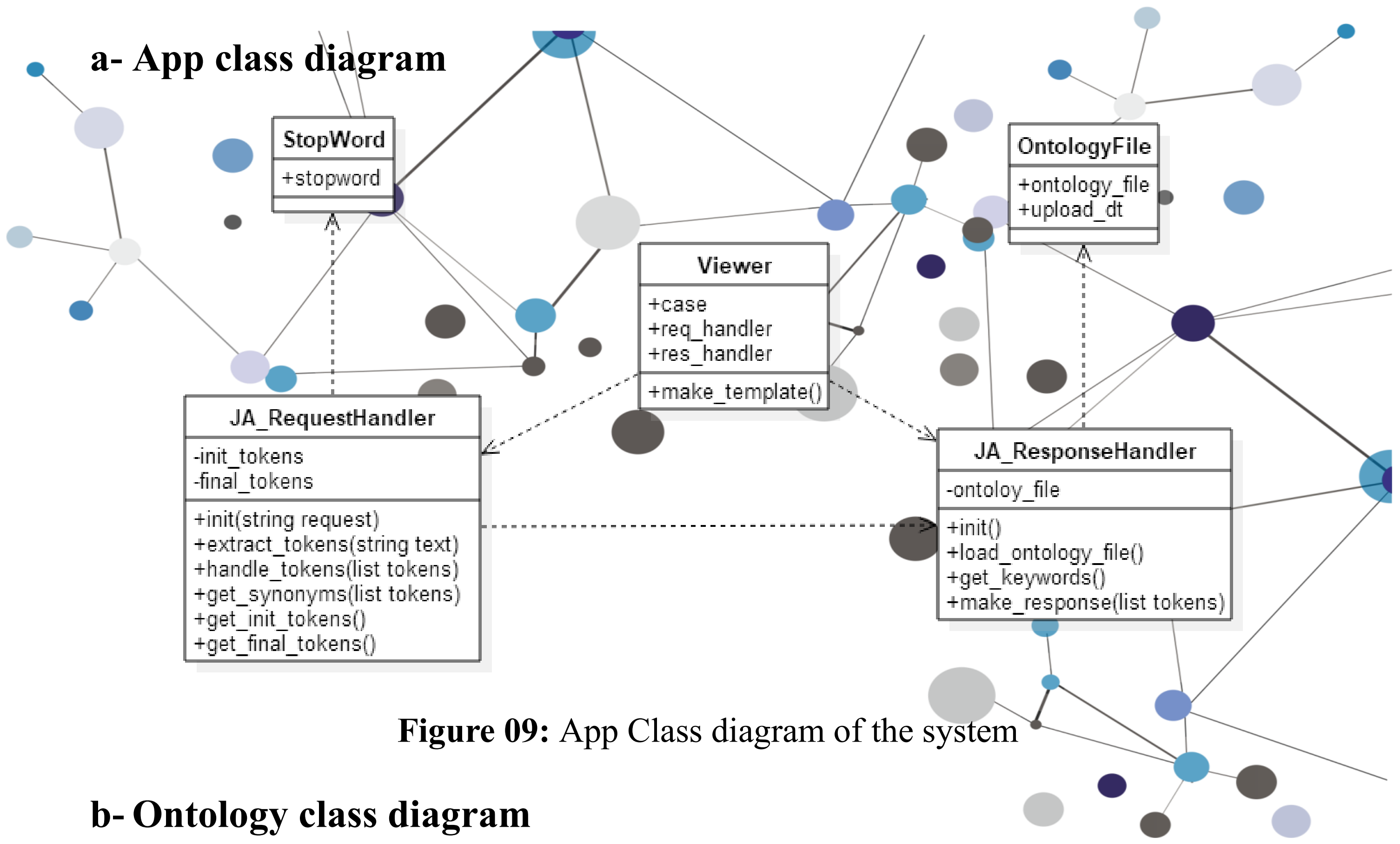


Figure 09: App Class diagram of the system

b- Ontology class diagram

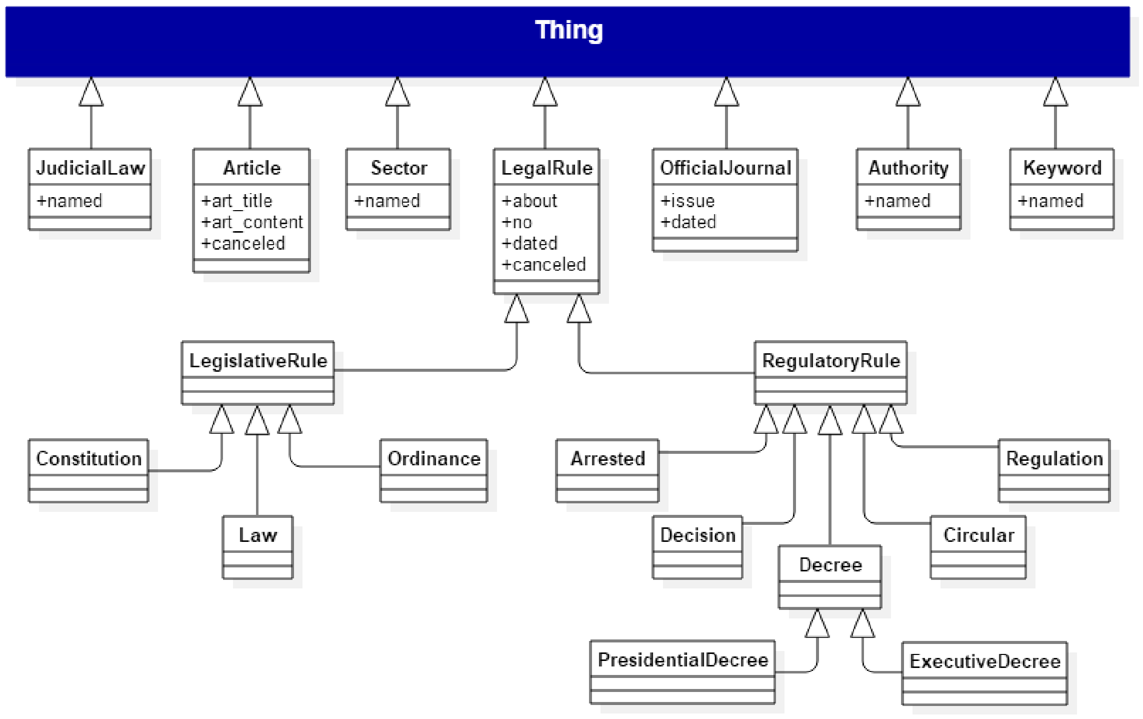


Figure 10: Class diagram of ontology

5.1.4. Activity diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. It describes the flow of control of the target system, such as the exploring complex business rules and operations, describing the use case also the business process. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e. workflows).

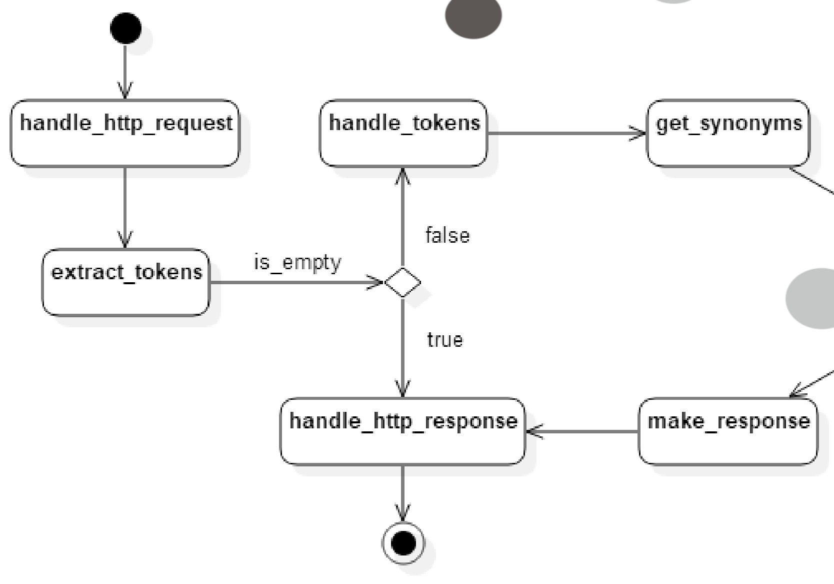


Figure 11: Activity Diagram of the system

6. Conclusion

In this chapter we presented the design of our system through the different UML diagrams. We started with the process of building our ontology. Then, we presented the different sequence and use case diagrams as well as the architecture of our system.

CHAPTER 03

Realization of Judicial Assistant System (JAS)

1. Introduction

After having designed our ontology and how to exploit its knowledge, we will now begin the implementation of the ontology and our regulatory information search system. We will then present, during this chapter:

- Development environment and tools,
- The implementation of our ontology,
- The development process of our system,
- The main interfaces of our system through screenshots.

2. Development environment and tools

Before starting the implementation of our system architecture, we chose a set of tools that can meet the development requirements of our system in view of the possibilities and advantages they offer.

2.1. Visual Studio Code (Code editor)

Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add functionality. [19]

2.2. Protégé (Ontology editor)

Protégé is a free, open source ontology editor and a knowledge management system. The Protégé meta-tool was first built by Mark Musen in 1987 and has since been developed by a team at Stanford University. The software is the most popular and widely used ontology editor in the world. [20]

2.3. Django (web framework)

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, to allow the developer to focus on writing his app without needing to reinvent the wheel. It's free and open source. [21]

2.4. SQLite

SQLite is a database engine written in the C programming language. It is not a standalone app; rather, it is a library that software developers embed in their apps. As such, it belongs to the family of embedded databases. It is the most widely deployed database engine, as it is used by several of the top web browsers, operating systems, mobile phones, and other embedded systems. [22]

2.5. Owlready2 (Python package for ontology)

Owlready2 is a package for manipulating OWL 2.0 ontologies in Python. It can load, modify, save ontologies, and it supports reasoning via HermiT (included). Owlready allows a transparent access to OWL ontologies. [23]

2.6. Camel-tools (Python package for NLP)

CAMeL Tools, is a collection of open-source tools for Arabic natural language processing in Python. CAMeL Tools currently provides utilities for pre-processing, morphological modeling, dialect identification, named entity recognition and sentiment analysis. In this paper, we describe the design of CAMeL Tools and the functionalities it provides.

3. Creation of Our Ontology by Protégé

3.1. Creating a project:

To create ontology, we must first create a project by choosing the desired language (or format).

3.2. Creation of classes:

Once the project has been created and named, we can proceed to create the classes. To do this, we would need to activate the “Classes” tab. And in the left pane, click on the root named “THING” with the right mouse button, this will display a context menu. Clicking on the "Add subclass" command will display a text box in which we will enter the name of the class. This is placed directly under the root.

We give, in the following figure, an overview of the "`judicial_ontology`" ontology for searching regulatory texts published under Protégé.

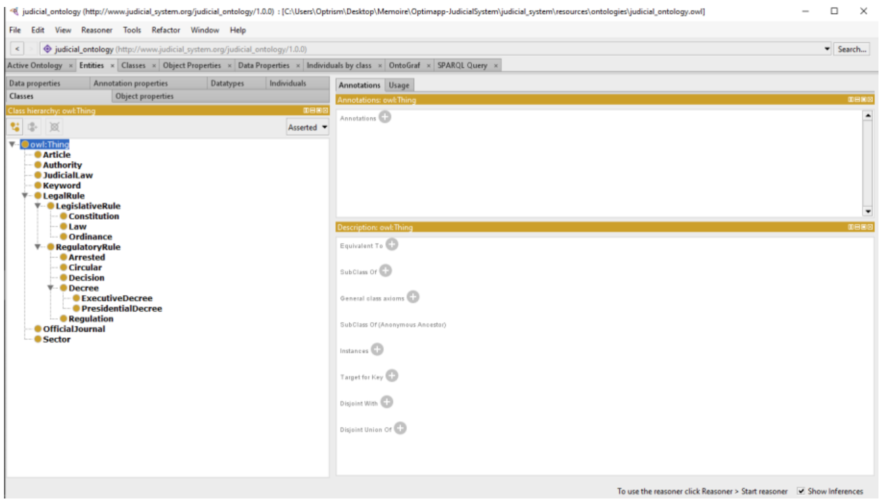


Figure 12: Creation of classes

3.3. Creation of data properties

Properties are created by selecting the class concerned in the left pane of the tree, then clicking on the icon located in the "DatatypeProperty" area, in the right pane. A dialog box is displayed allowing us to enter the different attributes:

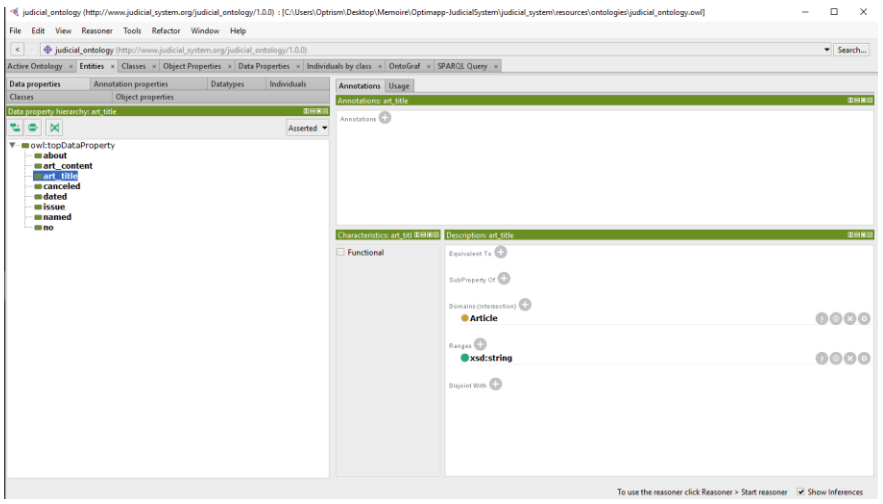
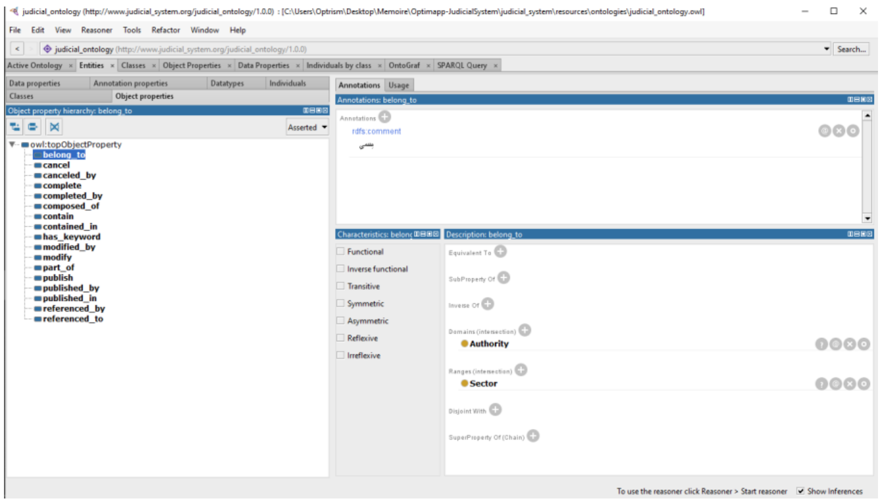


Figure 13: Creation of data properties

3.4. Creation of object properties (relationships)

The relationship creation is done by selecting the class in the left pane of the tree, then clicking on the icon located in the "Objectproperty" area, in the right pane. A dialog box is displayed allowing us to enter the different relationships:



judicial_ontology (http://www.judicial_system.org/judicial_ontology/1.0.0) : [C:\Users\Optrism\Desktop\Memoire\Optimapp-JudicialSystem\judicial_system\resources\ontologies\judicial_ontology.owl]

Object property hierarchy: belong_to

- owl:topObjectProperty
  - belong_to
  - cancel
  - canceled_by
  - complete
  - completed_by
  - composed_of
  - contain
  - contained_in
  - has_keyword
  - modified_by
  - modify
  - part_of
  - publish
  - published_by
  - published_in
  - referenced_by
  - referenced_to

Annotations: belong_to — rdfs:comment

Characteristics: belong_to — ☐ Functional ☐ Inverse functional ☐ Transitive ☐ Symmetric ☐ Asymmetric ☐ Reflexive ☐ Irreflexive

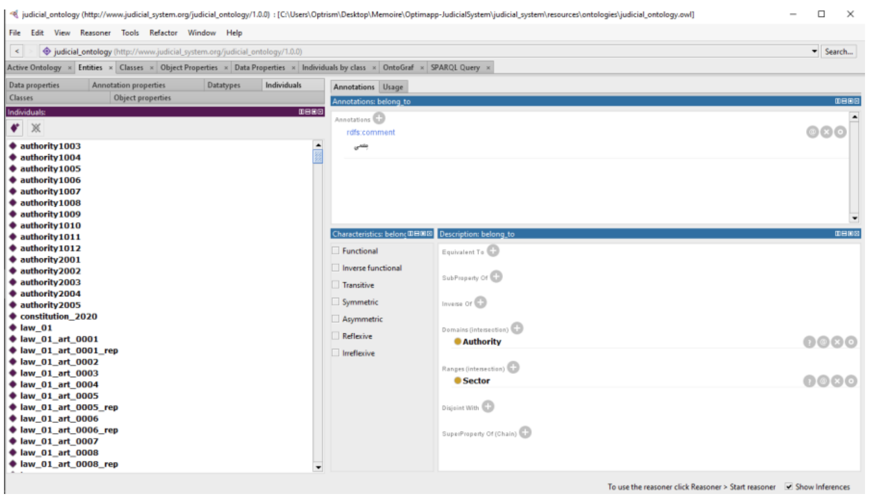
Description: belong_to — Domains (intersection): Authority; Ranges (intersection): Sector

To use the reasoner click Reasoner > Start reasoner ☑ Show Inferences

Figure 14: Creation of object properties (relationships)

3.5. Reservation of instances (Individuals)

Instances are entered by first activating the "Individuals" tab. Once this is done, we must select the class in the left pane of the project, then click on in the right pane of the project. In the class attributes boxes, enter the attribute values.



judicial_ontology (http://www.judicial_system.org/judicial_ontology/1.0.0) : [C:\Users\Optrism\Desktop\Memoire\Optimapp-JudicialSystem\judicial_system\resources\ontologies\judicial_ontology.owl]

Individuals:

- authority1003
- authority1004
- authority1005
- authority1006
- authority1007
- authority1008
- authority1009
- authority1010
- authority1011
- authority1012
- authority2001
- authority2002
- authority2003
- authority2004
- authority2005
- constitution_2020
- law_01
- law_01_art_0001
- law_01_art_0001_rep
- law_01_art_0002
- law_01_art_0003
- law_01_art_0004
- law_01_art_0005
- law_01_art_0005_rep
- law_01_art_0006
- law_01_art_0006_rep
- law_01_art_0007
- law_01_art_0008
- law_01_art_0008_rep

Annotations: belong_to — rdfs:comment

Characteristics: belong_to — ☐ Functional ☐ Inverse functional ☐ Transitive ☐ Symmetric ☐ Asymmetric ☐ Reflexive ☐ Irreflexive

Description: belong_to — Domains (intersection): Authority; Ranges (intersection): Sector

To use the reasoner click Reasoner > Start reasoner ☑ Show Inferences

Figure 15: Reservation of instances (Individuals)

3.6. Ontology graph

The figure shown below represent out ontology graph, obtained from the protégé, after applied the previous steps.

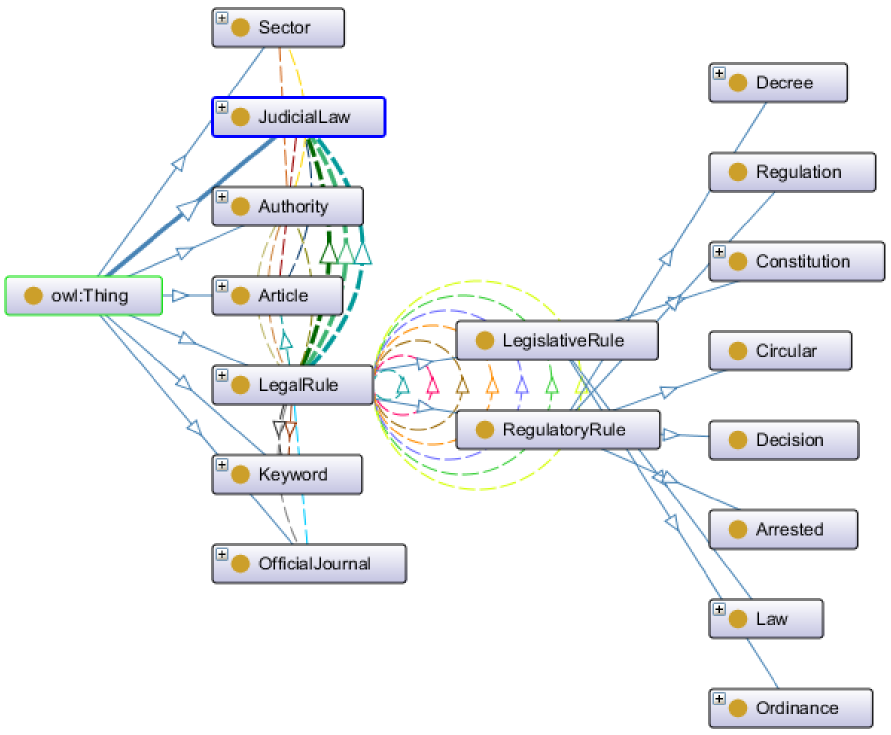
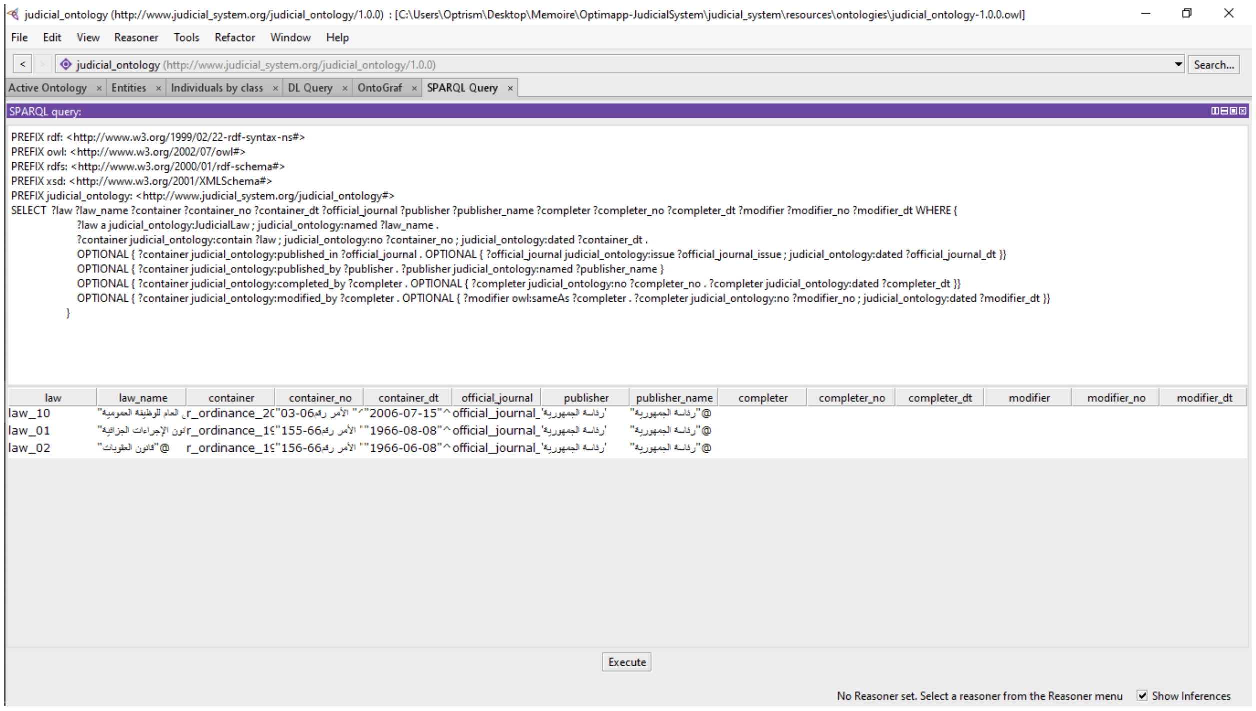


Figure 16: Ontology graph of our system

3.7. Make SPARQL query on Ontology


judicial_ontology (http://www.judicial_system.org/judicial_ontology/1.0.0) : [C:\Users\Optrism\Desktop\Memoire\Optimapp-JudicialSystem\judicial_system\resources\ontologies\judicial_ontology-1.0.0.owl]
File Edit View Reasoner Tools Refactor Window Help
judicial_ontology (http://www.judicial_system.org/judicial_ontology/1.0.0)
Search...
Active Ontology
Entities
Individuals by class
DL Query
OntoGraf
SPARQL Query
SPARQL query:
Execute
No Reasoner set. Select a reasoner from the Reasoner menu
Show Inferences

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX judicial_ontology: <http://www.judicial_system.org/judicial_ontology#>
SELECT  ?law ?law_name ?container ?container_no ?container_dt ?official_journal ?publisher ?publisher_name ?completer ?completer_no ?completer_dt ?modifier ?modifier_no ?modifier_dt WHERE {
            ?law a judicial_ontology:JudicialLaw ; judicial_ontology:named ?law_name .
            ?container judicial_ontology:contain ?law ; judicial_ontology:no ?container_no ; judicial_ontology:dated ?container_dt .
            OPTIONAL { ?container judicial_ontology:published_in ?official_journal . OPTIONAL { ?official_journal judicial_ontology:issue ?official_journal_issue ; judicial_ontology:dated ?official_journal_dt }}
            OPTIONAL { ?container judicial_ontology:published_by ?publisher . ?publisher judicial_ontology:named ?publisher_name }
            OPTIONAL { ?container judicial_ontology:completed_by ?completer . OPTIONAL { ?completer judicial_ontology:no ?completer_no . ?completer judicial_ontology:dated ?completer_dt }}
            OPTIONAL { ?container judicial_ontology:modified_by ?completer . OPTIONAL { ?modifier owl:sameAs ?completer . ?completer judicial_ontology:no ?modifier_no ; judicial_ontology:dated ?modifier_dt }}
        }
```

law	law_name	container	container_no	container_dt	official_journal	publisher	publisher_name	completer	completer_no	completer_dt	modifier	modifier_no	modifier_dt
law_10	"ن العام للوظيفة العمومية	r_ordinance_2(	"03-06الأمر رقم"	"2006-07-15"^	official_journal_	'رئاسة الجمهورية'	"رئاسة الجمهورية"@						
law_01	"انون الإجراءات الجزائية	r_ordinance_1(	"155-66الأمر رقم"	"1966-08-08"^	official_journal_	'رئاسة الجمهورية'	"رئاسة الجمهورية"@						
law_02	"قانون العقوبات"@	r_ordinance_1(	"156-66الأمر رقم"	"1966-06-08"^	official_journal_	'رئاسة الجمهورية'	"رئاسة الجمهورية"@						

Figure 17: Some results after making SPARQL query on Ontology

4. GUI

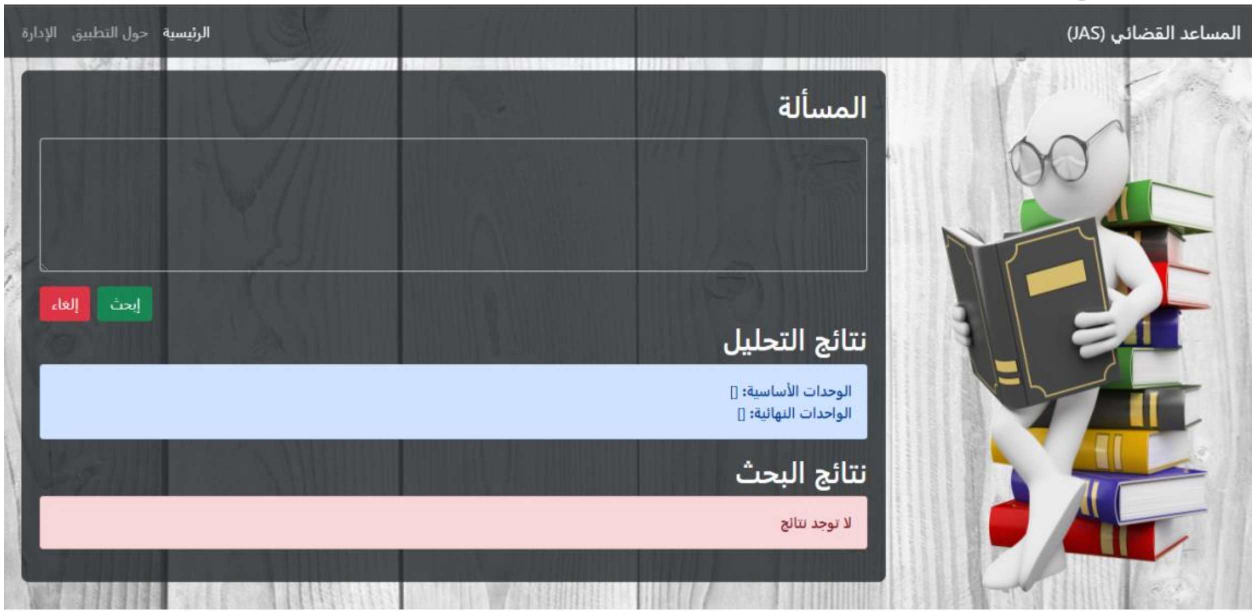
We designed our system as a web application, which we called Judicial Assistant (JAS). The application was divided into two parts: the first is for "general access part", and the other is for "administrator access part" to managing and updating the system.

4.1. General access part

We divided this part of system into two web pages: "Home page" and "About" as follow:

4.1.1. Home page

The interface shown in the figure below represents the home page of our system.



المسألة

نتائج التحليل

الوحدات الأساسية: []
الواحدات النهائية: []

نتائج البحث

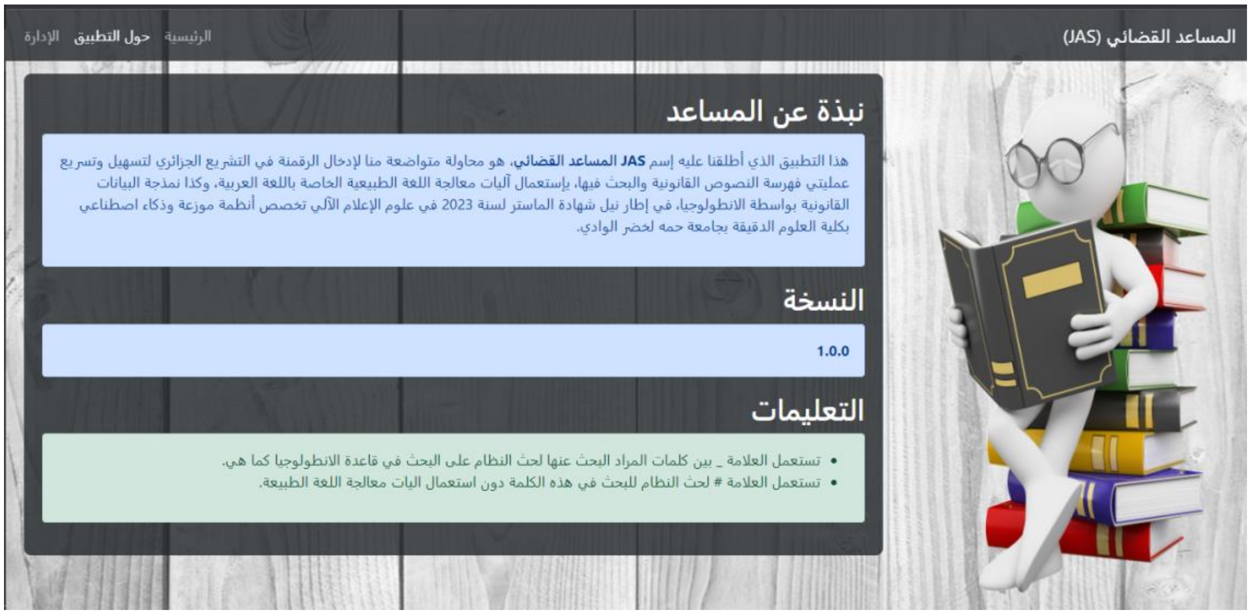
لا توجد نتائج

Figure 18: Home page of Judicial Assistant System (JAS)

This page allows the user to write the question in the text area designated for that purpose, and by pressing the "Search" button, he gets the answer consisting of two parts: "Analysis results" and "Research results".

4.1.2. About page

The interface shown in the figure below represents the about page of our system.



نبذة عن المساعد

هذا التطبيق الذي أطلقنا عليه إسم **JAS المساعد القضائي**، هو محاولة متواضعة منا لإدخال الرقمنة في التشريع الجزائري لتسهيل وتسريع عمليتي فهرسة النصوص القانونية والبحث فيها، بإستعمال آليات معالجة اللغة الطبيعية الخاصة باللغة العربية، وكذا نمذجة البيانات القانونية بواسطة الانطولوجيا، في إطار نيل شهادة الماستر لسنة 2023 في علوم الإعلام الآلي تخصص أنظمة موزعة وذكاء اصطناعي بكلية العلوم الدقيقة بجامعة حمه لخضر الوادي.

النسخة

1.0.0

التعليمات

- تستعمل العلامة _ بين كلمات المراد البحث عنها لحث النظام على البحث في قاعدة الانطولوجيا كما هي.
- تستعمل العلامة # لحث النظام للبحث في هذه الكلمة دون استعمال اليات معالجة اللغة الطبيعة.

Figure 19: About page of Judicial Assistant System (JAS)

This page shows the user some information about the application, the developers, its purpose, as well as some important instructions for searching.

4.2. Administrator access part

This part of system contains two web pages: “Login page” and “Administrator page”, as shown below:

4.2.1. Login page

The interface shown in the figure below represents the Login page of admin access part of our system.

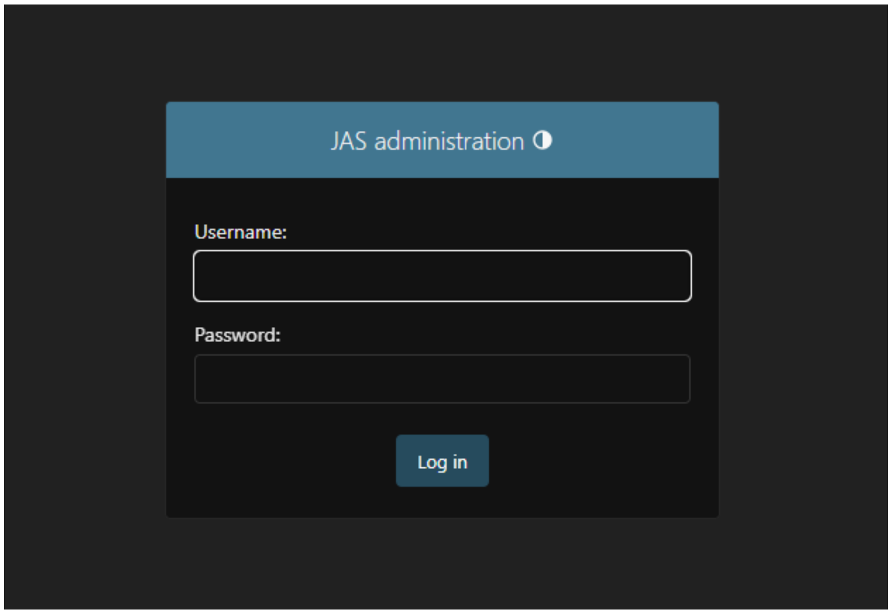
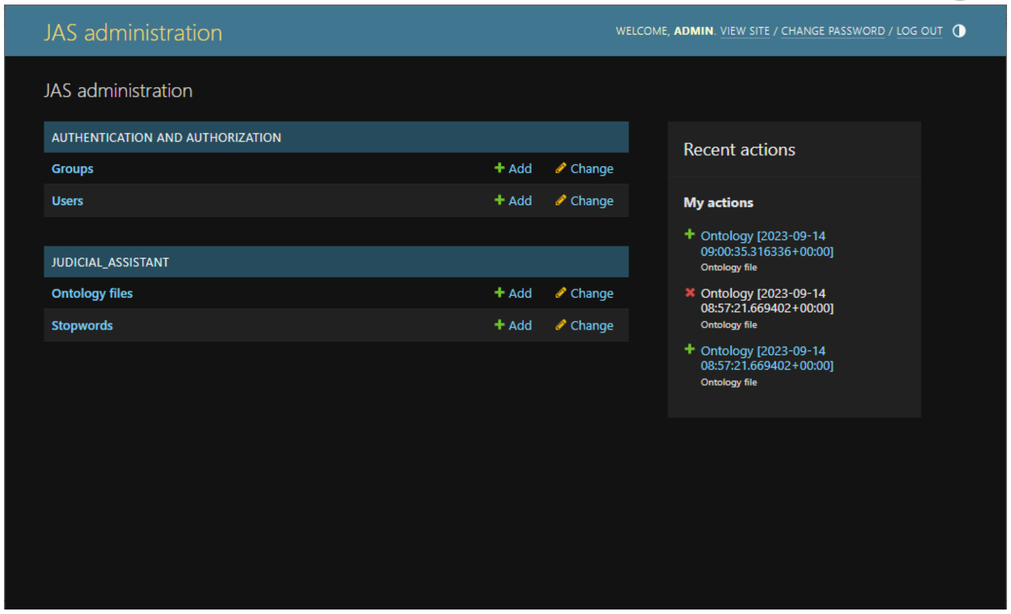


Figure 20: Login page of Judicial Assistant System (JAS) Administrator

4.2.2. Administrator page

The interface shown in the figure below represents the administrator page for our system, which authorizes him to manage and update its data, by modifying stop words and ontology files by adding, updating, or deleting.



JAS administration

WELCOME, ADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT

JAS administration

AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change
Users	+ Add	Change

JUDICIAL_ASSISTANT		
Ontology files	+ Add	Change
Stopwords	+ Add	Change

Recent actions

My actions

- + Ontology [2023-09-14 09:00:35.316336+00:00] Ontology file
- ✗ Ontology [2023-09-14 08:57:21.669402+00:00] Ontology file
- + Ontology [2023-09-14 08:57:21.669402+00:00] Ontology file

Figure 21: Administrator page of Judicial Assistant System (JAS)

5. Conclusion

In this chapter we presented the tools used and the steps followed for the implementation of our system which is based on an ontology. We also presented the different screenshots showing the different windows of our system. The research results have been encouraging. We hope that our system will be useful in solving the problem of searching for legal texts. Thus a user can quickly find a target legal text.

General conclusion and perspectives

The work carried out during this dissertation is part of ontological engineering, and essentially intended to be a contribution to the problem of the exploitation of knowledge in the field of Algerian regulations.

Ontology is a top-down approach to artificial intelligence that allows formal representation and semantic exploitation of knowledge in the field of modeling.

In this context, we first started by presenting the notion of information system, Natural Language Processing (NLP) and ontology. After that, we designed an ontologically based knowledge search system, using UML. And we created our system with the following tools: Protégé for editing the ontology, the Python language using owlready2 for working with OWL under Visual Studio Code, and use Django for creating the user interface via Browse (web app). We also used SQLite as database.

Following the work carried out, several points remain to be developed and improved. Among which we cite:

- ✓ Evaluate the ontology created by experts in the field, their point of view has a very significant impact to verify its completeness;
- ✓ Subsequently, develop it within a concrete application, for its validation

Bibliography

01- Harper, A. T., Antill, L., & Avison, D. E. (1985). Information systems definition: The multiview approach. Blackwell Scientific Publications, Ltd..

02- Moorthy, S., Ratchford, B. T., & Talukdar, D. (1997). Consumer information search revisited: Theory and empirical analysis. Journal of consumer research, 23(4), 263-277.

03- Ajallouda, L., Fagroud, F. Z., & Zellou, A. (2022). A Systematic Literature Review of Keyphrases Extraction Approaches. *International Journal of Interactive Mobile Technologies*, *16*(16).

04- Castells, P., Fernandez, M., & Vallet, D. (2006). An adaptation of the vector-space model for ontology-based information retrieval. *IEEE transactions on knowledge and data engineering*, *19*(2), 261-272.

05- Ozertem, U., Chapelle, O., Donmez, P., & Velipasaoglu, E. (2012, August). Learning to suggest: a machine learning framework for ranking query suggestions. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval* (pp. 25-34).

06- Mäkelä, E. (2005). Survey of semantic search research. In *Proceedings of the seminar on knowledge management on the semantic web*. Department of Computer Science, University of Helsinki, Helsinki.

07- Krama Ishak , THÈSE pour Conception et réalisation d’un système intelligent de recherche d’information réglementaire Master , UNIVERSITE KASDI MERBAH OUARGLA 2014.

08- Nadkarni, P. M., Ohno-Machado, L., & Chapman, W. W. (2011). Natural language processing: an introduction. Journal of the American Medical Informatics Association, 18(5), 544-551.

09- Gudivada, V. N. (2018). Natural language core tasks and applications. In Handbook of statistics (Vol. 38, pp. 403-428). Elsevier.

10- Jusoh, S. (2018). A study on NLP applications and ambiguity problems. Journal of Theoretical & Applied Information Technology, 96(6).

11- Aghdam, Z. N., Rahmani, A. M., & Hosseinzadeh, M. (2021). The role of the Internet of Things in healthcare: Future trends and challenges. Computer methods and programs in biomedicine, 199, 105903.

12- Raina, V., Krishnamurthy, S., Raina, V., & Krishnamurthy, S. (2022). Natural language processing. Building an Effective Data Science Practice: A Framework to Bootstrap and Manage a Successful Data Science Practice, 63-73.
13- Naçima MELLAL , THÈSE pour obtenir le grade de DOCTEUR ,UNIVERSITÉ DE SAVOIE27 Réalisation de l'interopérabilité sémantique des systèmes, basée sur les ontologies et les flux d'information
14- Gilles FALQUET , Mémoire de DEA en Management et Technologies des Systèmes d'Information (MATIS)
15- https://en.wikipedia.org/wiki/Ontology_components (18/09/2023-17:55)
16- HABIB-ELLAH GUERGOUR , « Construction d'une ontologie d'application dans le cadre de l'EAI », Magister en informatique, Université de Constantine
17- Thèse pour obtenir le grade de docteur L'Institut National des Sciences Appliquées de Lyon 27 Conception Coopérative d'Ontologies Pré-Consensuelles : Application au domaine de l'Urbanisme
18- Mémoire de Master Recherche Informatique Université des Sciences et Technologies de LilleDe l'apport des ontologies pour la conception de systèmes multi-agents ouverts
19- https://en.wikipedia.org/wiki/Visual_Studio_Code (10/09/2023 - 21:14)
20- https://en.wikipedia.org/wiki/Prot%C3%A9g%C3%A9_(software) (19/09/2023 – 19:00)
21- https://www.djangoproject.com/ (19/09/2023 – 19:05)
22- https://en.wikipedia.org/wiki/SQLite (19/09/2023 - 19:11)
23- https://owlready2.readthedocs.io/en/v0.42/intro.html (20/07/2023 - 10:02)