uAIS: An Experience of Increasing Performance of NLP Information Extraction Tasks from Legal Documents in an Electronic Document Management System

Marcos Ruiz[1], Cristian Román[1], Ángel Luis Garrido[2][a] and Eduardo Mena[2][b]
[1]*InSynergy Consulting S.A., Madrid, Spain*
[2]*SID Research Group, IIS Department, University of Zaragoza, Zaragoza, Spain*

Keywords: Optimization, Performance, Electronic Document Management System, Message Broker, NLP, Information Extraction.

Abstract: Nowadays, the huge number of documents which are managed through document management systems make their manual processing practically impossible. That is why the use of natural language processing subsystems that help to perform certain tasks begins to be essential for many commercial systems. Although its use is gradually extending to all levels, this type of subsystems presents the problem of its high requirements of resources from CPU and memory that can harm the entire system to which it intends to provide assistance. In this work, we propose and study an architecture based on microservices and message brokers which improves the performance of these NLP subsystems. We have implemented our approach on a real document management system, which performs intensive processes of language analysis on large legal documents. Experimental results show promising results, greatly increasing the productivity of systems based on other approaches.

1 INTRODUCTION

Due to the lowering of storage costs, nowadays private individuals, and especially private and public organizations, possess huge amounts of text-based documents, which are stored on their own computers or on the Web.[1]. This overflow due to the amount of information precludes manual treatment, and as a result, in recent years, the need of automated tools to analyze and process all this big amount of information has been more noticeable than ever. These types of tools fall within the field of *Natural Language Processing* (NLP), the scientific study of how to program computers to process and analyze large amounts of natural language. Challenges in NLP include speech recognition, natural language understanding, and natural language generation (Khurana et al., 2017).

Regarding the task of extracting information from text documents, when they performed by people specialized, it is a high time-consuming activity that additionally requires a high degree of expertise in the field treated by these documents.

[a] https://orcid.org/0000-0001-5600-0008
[b] https://orcid.org/0000-0002-7462-0080
[1] http://www.internetlivestats.com/

In fact, the greatest difficulty is the knowledge of the work context and the rules that lead to locate what are the specific data to be extracted. In addition, the task can also be especially expensive in time if the documents are large and the status of the document is not optimal, as sometimes happens with documents scanned using an optical recognition system (OCR). If we circumscribe to the information extraction task regarding to legal documents (laws, contracts, mortgages, agreements, etc.) we can observe that the language used exhibits a very specific vocabulary and expressions, so it is more difficult to understand and process. It leads to the need for a very specialized type of staff for carrying out this type of work. Moreover, these types of documents can be very long and tedious to read, which further complicates this type of work for a human.

Despite efforts to perform automated information extraction tasks that have been made for many years, there are still no commercial products specialized in these tasks, even more for very specific fields of application (science, history, or law, to name a few). Language is also another important limitation that requires specificity in developments. Therefore, the software that is developed with this aim is often a combination of existing resources with custom programming. In addition, these types of applications, being the extraction of information a very complex task, need many software resources: databases to store linguistic resources, large thesauri, complex libraries, or Internet services. That also leads to requiring many resources at the hardware level: CPU, memory, disk space, cache, bandwidth, etc.

In recent years, the application of automatic information extraction technologies to address these tasks has eased their realization in administrative and business fields. The lack of commercial products has led to the implementation of tailor-made solutions, which often prioritize effectiveness over efficiency.

Further, if we focus on the specific case of automatic processing of legal documents, we find additional difficulties such as the specificity of the vocabulary, the typology of the documents, the particularities of the languages in each region, and the concrete classification needs in each context. As a result, it is very complex to create a system capable of correctly solving particularly specific tasks and also for any use case (van Noortwijk, 2017). Moreover, this kind of documents contains, in many cases, a considerable amount of overlapping elements like stamps and signatures, which further hinders its automated treatment. These elements are usually required to prove the authenticity of the document, so they are hardly avoidable.

The purpose of this work is to study the optimization technologies for boosting NLP systems, specifically information extraction systems applied to legal documentation, with the ultimate goal of optimizing the performance of this kind of processes over that context. For this, we have used the technology of message brokers. Message brokers are elements in telecommunication or computer networks where software applications communicate by exchanging formally-defined messages (Magnoni, 2015).

To carry out this investigation in a rigorous way, on the one hand, we have worked with a real implementation of a system able of extracting information from legal documents using semantic technologies good results. On the other hand, given the specific nature of legal documents, real data sets will be required, an aspect that is usually complicated, since in many cases they are difficult to access private texts for the realization of experiments. To overcome this difficulty, this work has been carried out jointly by the research team of Insynergy (ISYC)[2], a well-known Spanish company dedicated to technology and innovation. The company has among some of its document management products a tool called AIS which processes large amounts of legal documents in order to extract information from them (Buey et al., 2016). Thanks to the participation of the company in this research work, an important set of these legal documents have been used to perform the experiments, both of the training and of the classifications. Despite the fact that the experimental dataset is composed of Spanish legal documents, our approximation is generic enough to be applied to any type of legal documents and regardless of language.

This paper is structured as follows. Section 2 analyzes and describes the state of the art. Section 3 describes the methodology proposed for the automatic classification process. Section 4 show and discusses the preliminary results of our experiments with real legal documents. Finally, Section 5 explains our conclusions and future work.

[2]https://www.isyc.com/

2 RELATED WORK

Since the widespread implementation of Electronic Document Management Systems (EDMS), like described at (Chen et al., 1999), their performance has been analyzed from different points of view (Iacob and Jonkers, 2006; Burtylev et al., 2013). If we focus on the problem at the level of system performance that represents the in-depth treatment of extensive text-based documents, it is presently too in many scientific papers. Systems like the one described in (Srihari and Shi, 2004) already showed the difficulties to optimize the performance of systems dedicated to the interpretation of documents. Some authors focus their study on the optimization of the processes of this type of systems (access, download, delete, etc.) (Cheng et al., 2013), but they do not specifically address the problem of NLP subsystems. Other works base their optimization on analyzing the system in depth: In (Kiedrowicz et al., 2016), the authors analyze a document management system representing them through the use of graph models. Another option is the use of Business Process Management (BPM) tools to model the processes and thus detect the bottlenecks, to later provide these parts of the system with more resources (Djedović et al., 2016). These authors make a good approximation to the analysis part of the problem, but it remains to be specified how to make that optimization.

In regard to the use of NLP in legal document management systems, it can be seen that its use has been increasing in recent years. For example, in (Hachey and Grover, 2006) the authors integrate an automatic summarization tool into a document system for storing judgments of the UK House of Lords. Another system dedicated to manage notary documents is described in (Amato et al., 2008), which transforms them into RDF statements for suitable indexing, retrieval and long term preservation. Eunomos (Boella et al., 2016) is a legal document and knowledge management system, based on legislative XML and ontologies, which uses NLP tools to semi-automate lower-skill tasks. In all of them it is clear the high computational cost that this type of process entails, but its optimization is pending. There are also interesting approaches that study how to serialize or parallelize NLP services (Garrido et al., 2014; McEwan et al., 2016), but they are not directly related to EDMS nor belong to the legal field.

Regarding message brokers, the first references of architectures based on the use of services accessible through messages can be found in the works carried out in the 90s, such as (Oki et al., 1994). These, in turn, are inspired by previous works focused on the design of architectures for distributed systems (Banavar et al., 1999). In these works a message broker is defined as an intermediary software that translates a message from the formal messaging protocol of the sender to the formal messaging protocol of the receiver. In the architecture with traditional broker all the messages go through a static component that will always be separated from the different groups of applications that want to communicate with each other: producers and consumers. This architecture is simple, light, and easy to adapt in other contexts, since they only have to know how to connect to the broker. The broker might be distributed on different machines or not.

The main objective of message queuing systems is to offer a reliable and simple message exchange system for the application developer. For this, different techniques are used as such: 1) disk usage through databases or the operating system itself, 2) replication of messages through various brokers, 3) replication of brokers to ensure high availability, 4) use of transactions, and 5) use of confirmation messages, both in publication and consumption.

Nowadays, message brokers have many uses (Yongguo et al., 2019), as managing workload queues or messages for multiple receivers, routing messages to one or more destination, transforming messages to an alternative representation, invoking services to retrieve data, etc. Even so, although message brokers are applied in multiple scenarios, as far as we have known, its use has not been analyzed yet how it works on an EDMS in combination with natural language processing subsystems.

3 METHODOLOGY

The objective is to optimize the performance of an embedded NLP subsystem within an EDMS. The large computational requirements of the NLP module in charge of processing documents contribute to making the EDMS slower and therefore more inefficient. The proposed solution to improve system performance and scalability is described below.

3.1 Message Oriented Middleware

The main idea is to convert the NLP subsystem into an external service that can be replicated in several nodes so that they can solve different requests in parallel instead of sequentially. For this, the design approach is to use an asynchronous messaging service. There is a wide range of solutions of this nature that allow parallelizing and distributing different computational works. The use of a Message Oriented Middleware (MOM) has been chosen, a type of architecture that offers the following functionalities:

- Sending and receiving messages in a non-blocking and asynchronous manner.
- Message persistence through disk usage (database or operating system included).
- Replication of messages through various brokers to prevent the loss of messages.
- Replication of brokers to ensure high service availability.
- Use of transactions to ensure that all messages are processed.
- Use of confirmation messages, both in publication and in consumption, to ensure that requests are sent and received correctly.

It has been decided to incorporate a message broker like MOM which is responsible for managing the exchange of messages between producers and consumers. The broker can also be distributed on different machines. The communication process is depicted in Figure 1.

In this architecture we find on the one hand a group of consumers who are going to demand information (EDMS nodes), and some workers (NLP module) who are going to provide it. For this, both groups of programs subscribe to the Broker to synchronize with it, so that when a consumer issues a request, that request is put in a queue and finally sent to a worker to solve it. The worker confirms receipt of the request to the Broker, processes it, and sends the answer again to the Broker. This response is put in a queue and finally sent by the Broker to the consumer who has made it, which finally confirms its receipt to the Broker.

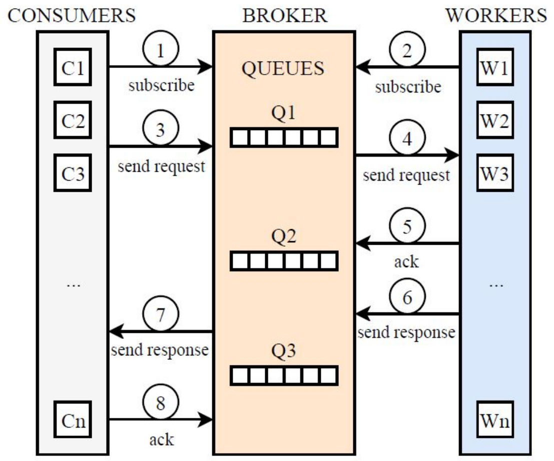

Figure 1: Proposed architecture using a message broker: the consumers are different nodes of the EDMS, and the workers are services that incorporate NLP functionality.

3.2 Design

The design of the proposed architecture is described in greater detail below:

1. EDMS: Responsible for receiving document processing requests. Depending on the number of users to attend, it is assumed that several nodes are deployed to offer a service that satisfies customers. In the case of having more than one node, the clients access the different modules through a load balancer. In addition, it is also assumed that each of these nodes needs to access a file system where the documents are located, and a database to update data on said documents.
2. Microservice: It will contain the NLP functionality and is responsible for processing the documents. Their number can be increased so that service availability can always be guaranteed and adapted to the number of customer requests.
3. Message broker: The message broker is the component that allows communication between the nodes of the EDMS and any microservice. It is a critical component in this system, since there is only one and it also has a status. If this component fails, there may be lost messages and users will not get a response to their requests. To avoid this type of fault, several solutions are proposed:
   - Persistence through saving messages to disk.
   - Replication of messages through the master-slave pattern.
   - Use of transactions.
   - Replication or distribution of message brokers.

4 EVALUATION

To meet the objectives, a solution based on the proposed architecture explained in Section 3 has been developed. Next, the use case used to evaluate the proposed architecture will be discussed.

4.1 Context and Dataset

To evaluate the proposed architecture, we have worked on the Oncostumer system, a CRM (Customer Relationship Management), that belongs to Insynergy (ISYC)[3]. Oncostumer is an EDMS in charge of managing the legal documentation of its clients. It is necessary to extract data from the documents, and for doing this, a subsystem focused on the NLP tasks is used. OnCostumer uses the file system to save customer documents and a database to store customer information and document data.

The tests have been carried out with a certain type of legal document, specifically with sales contracts. These types of contracts, which may have a high number of pages, contain information on the property to be sold, the seller, and the buyer, as well as specific data on the same transaction. These documents are relatively long: between 75 and 100 pages each. The task of NLP that is performed on these documents is a laborious extraction of information that requires a series of highly resource-consuming actions: spell checking, correction by using N-grams, lemmatization, parsing, application of rules, ontology queries and Named Entity recognition (NER) tasks. All this to be able to extract from each document 55 data embedded in the text, among which are the names, surnames and identifying data of the people involved, the information about the property (such as its size in square meters thereof), and the transaction data (such as the price of the transaction).

4.2 Baseline System

To carry out the complex tasks of information extraction, the AIS system is used footnoteAIS is the Spanish acronym for *Semantic Analysis and Interpretation*. (Buey et al., 2016), a system created from the joint work of ISYC and the University of Zaragoza. This system has as input legal documents scanned with an OCR (Optical Character Recognition), and its output is the relevant information automatically extracted using artificial intelligence techniques. The AIS system performs three major tasks: a preprocess, a central extraction task, and a postprocess (Garrido and Peiró, 2018; Buey et al., 2019).

[3]https://www.isyc.com/

This system is embedded within the OnCustomer platform, which, as mentioned above, is a complex CRM solution. AIS is implemented as one of the many resources available for this platform. The hardware on which it works is a high performance machine to service all customers who use this solution. OnCostumer can be cloned into several nodes to support clients in parallel. The AIS system consists of a process that is launched periodically. Check if there are documents in the input directory, process them, and deposit the results in several output directories, depending on the status (correct or erroneous). Oncostumer collects the system results from these directories and performs the following document processing tasks based on them.

The main problems of this architecture are:

- The work done by the library is sequential, so the maximum capacity of the machine cannot be used.
- The scaling is only vertical, so the capabilities of AIS only can be expanded by improving the machine itself.
- In case of having the CRM in several nodes, this library should be included in each of them, making these nodes even heavier than they already are.
- Preprocessing, extraction and postprocessing cannot be used separately.
- The task is executed from time to time, i.e., tasks are not starting immediately and calls are being made to the filesystem unnecessarily.

4.3 Optimized System

The new system is an implementation of the microservice-based architecture proposed in Section 3, and its main objective is to decouple the AIS library from the rest of the CRM. Each of the microservices responsible for NLP processing will be called *uAIS*. Now the communication between the Document Management System and the NLP subsystem is based on the competing-consumer's pattern. This pattern works as follows:

1. Firstly, OnCostumer leaves in the IN folder the legal document from which is wanted to extract the data.
2. Secondly, one of the nodes of the CRM sends a message to the message broker, also indicating the queue to use.
3. The message broker decides to which node uAIS is sending the request through the Round-Robin method. This implies that if we have N uAIS services waiting for messages. The messages will be distributed equally as long as there are no uAIS busy with previous requests.
4. Each of the uAIS nodes are responsible for receiving the messages, leaving the XML file in the OUT folder and sending the reply message to the message broker. If uAIS suffers a shutdown, it will not be able to send the response message, so after a reasonable period of time the message broker will forward the request to the next active uAIS node or will wait for one.
5. Once one of the uAIS services has processed a request and deposited the response XML file in the shared file system, it sends a message indicating the location of the output XML file and if it has been processed correctly. These response messages are sent through the Round-Robin method to the nodes of the CRM to copy the XML file to their memory area and update the database to notify the user that the process has been completed.

4.4 Implementation

In the Figure 2 a scheme of the implementation and operation of the new system is shown. On the one hand, there are the users that interact with each of the nodes of the CRM through a load balancer and these in turn share the data through two databases. On the other hand, there is the message broker that is responsible for communicating the different nodes of the CRM with the different AIS microservices that ensure high service availability. And finally there is the shared file system whereby the CRM nodes and the AIS microservices exchange the documents to be processed and the response XML files processed.

For the implementation of the new system, the use of a RabbitMQ broker [4] has been chosen since it meets each and every one of the following requirements:

- Permissive license, very advantageous for its great compatibility with other types of license, and even compatible with copyleft licenses.
- Guaranteed long-term support and compatibility with software of a certain age. Very interesting also because not always is possible to work with the latest technology, on the contrary, it is very common to have legacy systems.
- Guarantees in delivery of shipment to total processing of the request.
- Disk persistence, which provides greater security against loss of information in case of system crashes.

[4]https://www.rabbitmq.com/

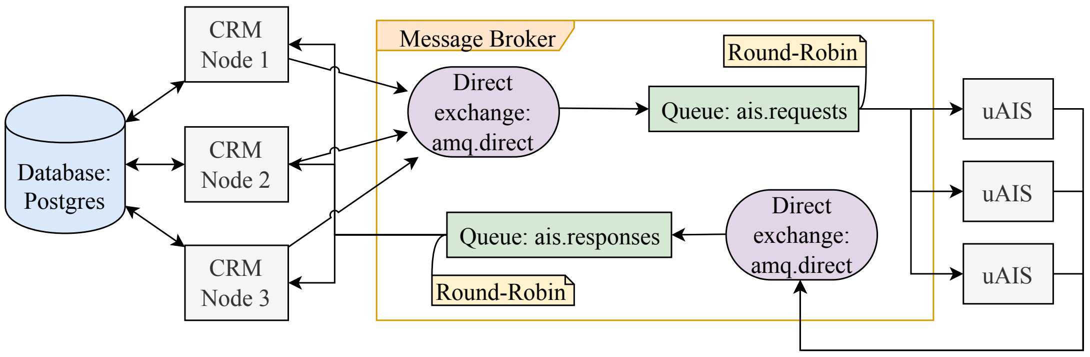

Figure 2: Architecture of the proposed communication system. On the left are the CRM instances that make requests to the uAIS services located on the right side. The management of these requests is done through the Message Broker.

- Monitoring with graphical interface.
- Integration with Spring and Maven repository.
- Authentication and authorization systems.
- Sufficient performance for our requirements.
- Ability to use the pub / sub pattern (publisher / subscriber), consumers competitors and RPC (Remote Procedure Call).
- Compatibility with numerous standard protocols such as AMQP, STOMP and MQTT.
- Great community of users.
- A very low RAM usage compared to ActiveMQ.
- Use of virtual hosts to be able to use the same message broker in different environments at the same time.

The implementation of the uAIS microservice itself was made with the latest stable version of the Springs Framework since it is the same framework used by the CRM, which will simplify its maintenance. This microservice will integrate the previous version of AIS as a library, but adapted to its new service format. One of the objectives of the uAIS development is to make it as simple as possible since it will only be in charge of communicating with the messaging broker and configuring and starting the AIS library.

4.5 Results

To evaluate the performance of both systems, the CPU, memory, and performance consumption of both the base system and the new system are monitored first. To do the tests, eight uAIS microservices have been activated waiting for messages.

As can be seen in Figure 3, the difference in the CPU usage of the CRM without AIS, compared to the old system is very significant. It has gone from practically not using the CPU (since the CRM is waiting for the results and only works when they are already available), to take full advantage of it.

With regard to memory use, in Figure 4 it can be seen how RAM consumption is similar in both systems. This is because, on the one hand, the CRM has a minimum of very high RAM in use due to its complexity. This causes that the differences between both systems are not seen so clearly. On the other hand, the memory consumption of the CRM using AIS as a service has ups and downs during the whole process, each peak being the collection of an AIS result.

Regarding performance, Figure 5 shows an improvement of 77%, from taking almost 450 seconds to less than 100 seconds in processing 200 documents.

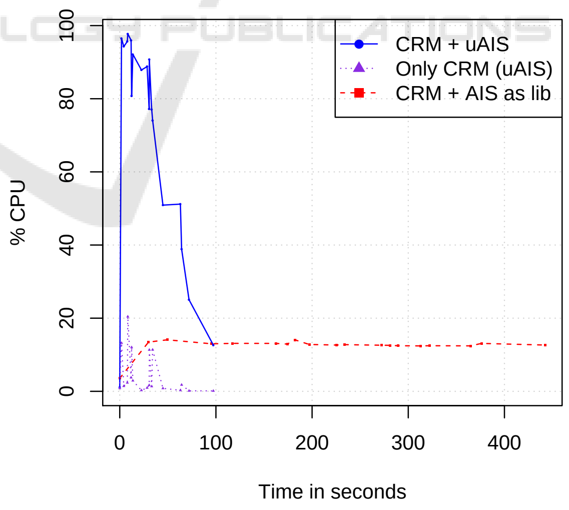

Figure 3: Comparison of the processor usage of the baseline system versus the new system.

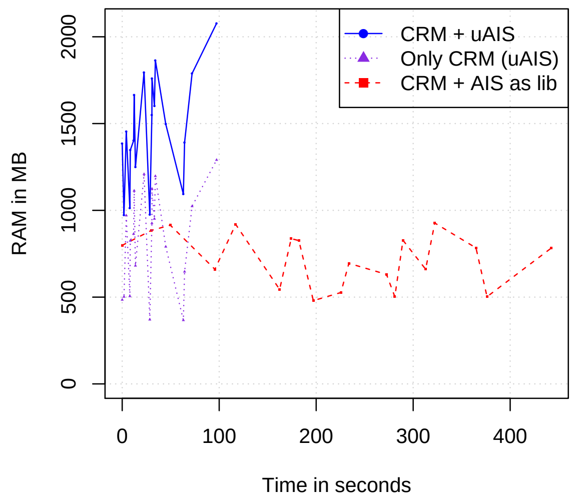

Figure 4: Comparison of the memory usage of the baseline system versus the new system.

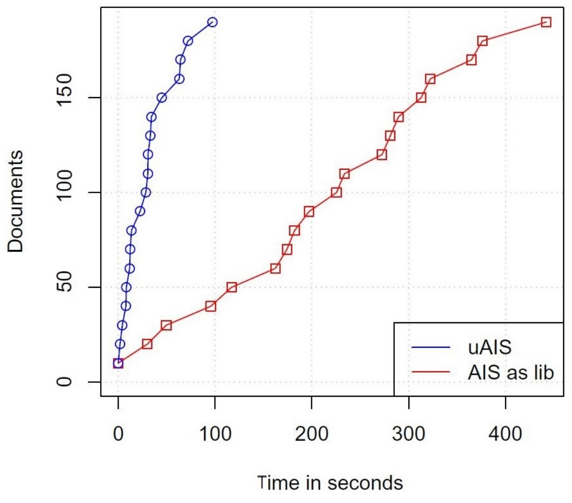

Figure 5: Comparison of the performance of the baseline system versus the new system.

5 CONCLUSIONS AND FUTURE WORK

Notarial acts, sales documents, judicial acts, contracts, etc., are types of legal documents widely used, but there are not too many specialized tools for processing them. Moreover, the automatic information extraction of this type of documents is still done manually or with low performance automatic tools based on monolithic solutions.

In this work, we have proposed an approach based of message brokers to optimize this type of systems. Due to the difficulty in finding suitable datasets in the field of legal documents, the development of this work and the experimental tests have been carried out in collaboration with a company dedicated to the processing of this type of documents.

The main contributions of this work is the design of an architecture based on microservices and message brokers which improves the performance of NLP subsystems, which require many resources from CPU and memory. Our approach offers scalable performance by allowing asynchronous communication between machines. As these are subsystems that can be isolated from the information storage modules (such as databases), their conversion to a queue system offers great advantages without excessive complexities in the developing: 1) allows the maximum capacity to be used. the machine where it is housed, 2) can perform a horizontal scaling at the level of work lots, 3) It makes the main document management systems lighter, and 4) it facilitates maintenance and testing.

In addition, a series of automatic tests have been carried out with a real system for extracting information from legal documents, measuring response times, the use of RAM and the use of the CPU. Regarding the results, the system has achieved an improvement of up to 77% in performance.

There are two main lines of development for future work:

1. Making NLP microservices multifunctional and having independent configurations, which would give the system additional flexibility. This configuration would be embedded in the messages themselves.
2. Subdividing the NLP modules into several microservices, which would make them even lighter and easier to invoke and maintain. In this way, we could add more nodes in the most demanding parts in CPU, thus improving even more the response times of the system.

ACKNOWLEDGEMENTS

This research work has been supported by the OTRI Projects ICIX6 2018/0577 and ICIX7 2019/0628 at the University of Zaragoza, and the CICYT TIN2016-78011-C4-3-R (AEI/FEDER, UE).

REFERENCES

Amato, F., Mazzeo, A., Penta, A., and Picariello, A. (2008). Using nlp and ontologies for notary document management systems. In *2008 19th International Workshop on Database and Expert Systems Applications*, pages 67–71. IEEE.

Banavar, G., Chandra, T., Strom, R., and Sturman, D. (1999). A case for message oriented middleware. In *International Symposium on Distributed Computing*, pages 1–17. Springer.

Boella, G., Di Caro, L., Humphreys, L., Robaldo, L., Rossi, P., and van der Torre, L. (2016). Eunomos, a legal document and knowledge management system for the web to provide relevant, reliable and up-to-date information on the law. *Artificial Intelligence and Law*, 24(3):245–283.

Buey, M. G., Garrido, A. L., Bobed, C., and Ilarri, S. (2016). The AIS project: Boosting information extraction from legal documents by using ontologies. In *Proceedings of the 8th International Conference on Agents and Artificial Intelligence*, pages 438–445. INSTICC, SciTePress.

Buey, M. G., Roman, C., Garrido, A. L., Bobed, C., and Mena, E. (2019). Automatic legal document analysis: Improving the results of information extraction processes using an ontology. In *Intelligent Methods and Big Data in Industrial Applications*, pages 333–351. Springer.

Burtylev, I., Mokhun, K., Bodnya, Y., and Yukhnevich, D. (2013). Development of electronic document management systems: Advantage and efficiency. *Science and Technology*, 3(2A):1–9.

Chen, Y.-J. J., Ferguson, D. R., Hong, A. N., Suleman, D., and Whittemore, G. L. (1999). Computer-based document management system. US Patent 6,009,442.

Cheng, W. Z., Yang, Y., Zhang, L., and Li, L. (2013). Optimization for web-based online document management. In *Advanced Materials Research*, volume 756, pages 1135–1140. Trans Tech Publ.

Djedović, A., Žunić, E., Alić, D., Omanović, S., and Karabegović, A. (2016). Optimization of the business processes via automatic integration with the document management system. In *2016 International Conference on Smart Systems and Technologies (SST)*, pages 117–122. IEEE.

Garrido, A. L., Buey, M. G., Escudero, S., Peiro, A., Ilarri, S., and Mena, E. (2014). The Genie Project - a semantic pipeline for automatic document categorisation. In *Proceedings of the 10th International Conference on Web Information Systems and Technologies*, pages 161–171. INSTICC, SciTePress.

Garrido, A. L. and Peiró, A. (2018). Recovering damaged documents to improve information retrieval processes. *Journal of Integrated OMICS*, 8(3):53–55.

Hachey, B. and Grover, C. (2006). Extractive summarisation of legal texts. *Artificial Intelligence and Law*, 14(4):305–345.

Iacob, M.-E. and Jonkers, H. (2006). Quantitative analysis of enterprise architectures. In *Interoperability of Enterprise Software and Applications*, pages 239–252. Springer.

Khurana, D., Koli, A., Khatter, K., and Singh, S. (2017). Natural language processing: State of the art, current trends and challenges. *arXiv preprint arXiv:1708.05148*.

Kiedrowicz, M., Nowicki, T., Waszkowski, R., Wesołowski, Z., and Worwa, K. (2016). Method for assessing software reliability of the document management system using the rfid technology. In *MATEC Web of Conferences*, volume 76, page 04009. EDP Sciences.

Magnoni, L. (2015). Modern messaging for distributed sytems. In *Journal of Physics: Conference Series*, volume 608, page 012038. IOP Publishing.

McEwan, R., Melton, G. B., Knoll, B. C., Wang, Y., Hultman, G., Dale, J. L., Meyer, T., and Pakhomov, S. V. (2016). Nlp-pier: a scalable natural language processing, indexing, and searching architecture for clinical notes. *AMIA Summits on Translational Science Proceedings*, 2016:150.

Oki, B., Pfluegl, M., Siegel, A., and Skeen, D. (1994). The information bus: an architecture for extensible distributed systems. In *ACM SIGOPS Operating Systems Review*, volume 27, pages 58–68. ACM.

Srihari, S. N. and Shi, Z. (2004). Forensic handwritten document retrieval system. In *First International Workshop on Document Image Analysis for Libraries, 2004. Proceedings.*, pages 188–194. IEEE.

van Noortwijk, K. (2017). Integrated legal information retrieval; new developments and educational challenges. *European Journal of Law and Technology*, 8(1):1–18.

Yongguo, J., Qiang, L., Changshuai, Q., Jian, S., and Qianqian, L. (2019). Message-oriented middleware: A review. In *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*, pages 88–97. IEEE.