Pontifícia Universidade Católica
do Rio de Janeiro

Técnicas Inteligentes para Interpretação de Documentos em uma Plataforma de Extração de Dados

Gabriel Augusto Silva de Aquino

RELATÓRIO DO PROJETO FINAL DE GRADUAÇÃO

Orientação Prof. Marco A. Casanova

CENTRO TÉCNICO CIENTÍFICO - CTC

DEPARTAMENTO DE INFORMÁTICA

Curso de Graduação em Ciência da Computação

Gabriel Augusto Silva de Aquino

Técnicas Inteligentes para Interpretação de Documentos em uma Plataforma de Extração de Dados

Relatório de Projeto Final, apresentado ao programa **Ciência da Computação** da PUC-Rio como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Marco A. Casanova

Rio de Janeiro, 12 de junho de 2021

AGRADECIMENTOS

A minha família, que possibilitou que eu cursasse essa excelente faculdade que é a PUC-RIO e sempre me deram apoio em todos os momentos difíceis, a alguns amigos próximos que fizeram parte importante dessa graduação, Hugo, Michelle, Mariela, Renan, Thiago, Abrahão, a todos os excelentes professores que tive na PUC-Rio que sempre faziam todo o possível para que seu conhecimento fosse absorvido pelos alunos, finalmente a toda a equipe do Tecgraf, Melissa que me acolheram e me deram todas as ferramentas necessárias para crescer como aluno e profissional, e finalmente ao meu orientador Casanova que possibilitou o desenvolvimento desse projeto.

RESUMO

Este projeto trata da incorporação de documentos textuais (por exemplo, com extensões .pdf, .txt e .doc) em um banco de dados, permitindo que as buscas encontrem palavras-chave no conteúdo destes documentos e que o usuário possa encontrar e navegar sobre os relacionamentos destes documentos com as outras entidades do banco de dados. Para tanto, o projeto desenvolve técnicas inteligentes para interpretação de documentos, tratando do reconhecimento de entidades nomeadas em documentos (NER), associando-as às entidades já existentes no banco de dados. Por fim, este projeto desenvolverá um workflow que automatiza esse processo.

Palavras-chave: Reconhecimento de entidades nomeadas, Bancos de Dados, Interpretação de Documentos

ABSTRACT

This project deals with the incorporation of textual documents (for example, with the extensions .pdf, .txt and .doc) in a database, allowing the user to submit keyword-based queries over the documents and to navigate through the relationships of these documents with the other entities in the database. To achieve this goal, the project develops intelligent techniques for interpreting documents, dealing with named entity recognition (NER), associating them with entities already existing in the database, and with document indexing. Lastly, this project also creates a workflow that automates the entire process.

Keywords: Named Entity Recognition, Database, Document Interpretation

Rio de Janeiro, Junho de 2021

SUMÁRIO

ÍNDICE DE FIGURAS

1. INTRODUÇÃO

Danke [1] é uma plataforma para extração de dados e conhecimento, desenvolvida no Instituto Tecgraf [2], PUC-Rio. Ela permite que os usuários realizem busca de palavras-chave sobre bancos de dados relacional e sobre conjuntos de triplas RDF [9]. Danke explora o esquema do banco de dados para compilar uma consulta em SQL ou SPARQL, com o mínimo de cláusulas de junção, que retornam dados que melhor se casam com as palavras-chave. A plataforma permite que os usuários explorem os resultados da busca em uma tabela, com os resultados mais relevantes no topo. Os usuários podem adicionar ou remover colunas desta tabela, adequando o resultado de acordo com seu interesse, e também navegar sobre o banco de dados a partir da resposta da consulta, acessando detalhes de cada resultado e seus relacionamentos com outros dados. Para que o Danke possa ser utilizado em um determinado domínio de aplicação, é necessário definir o esquema para o banco de dados de acordo com tal domínio, além de se realizar um processo de preparação do banco de dados, que envolve a ingestão, indexação e enriquecimento dos dados, de forma a responder com maior desempenho e eficácia as buscas do usuário.

Este projeto trata da incorporação de documentos textuais (por exemplo, com extensões .pdf, .txt e .doc) no banco de dados do Danke [1,8], permitindo que as buscas encontrem palavras-chave no conteúdo destes documentos e que o usuário possa encontrar e navegar sobre os relacionamentos destes documentos com as outras entidades do banco de dados. Para tanto, o projeto desenvolve técnicas inteligentes para interpretação de documentos, tratando do reconhecimento de entidades nomeadas em documentos (NER), e associando-as às entidades já existentes no banco de dados, da indexação dos documentos [6,7]. Como muitos dos documentos envolvidos são do tipo imagem (por exemplo documentos escaneados), este projeto também trata a utilização de técnicas de OCR (Optical Character Recognition) [13] para transformar o conteúdo de um documento do tipo imagem para texto, sempre que possível. Por fim, o projeto desenvolve um workflow [11] que abrange todas as técnicas desenvolvidas. Tal workflow é importante para permitir que essas técnicas sejam usadas em um grande número de documentos em um ambiente de processamento paralelo.

Este documento apresenta as atividades realizadas neste projeto durante o primeiro e o segundo semestres.

2. SITUAÇÃO ATUAL

Extração de dados refere-se à tarefa de extrair entidades, relacionamentos e valores de atributos de documentos ou arquivos semiestruturados. Na literatura, a tarefa específica de identificar entidades nomeadas em textos é conhecida por *named entity recognition* (NER) [3], e a tarefa de extrair relacionamentos entre entidades por *relation extraction* (RE) [4].

Extração de dados adquiriu grande importância pois existe um volume considerável de informação disponível sob forma de texto em linguagem natural, que é conveniente para consumo de usuários humanos, mas não para uso por algoritmos de análise de dados. Este fato exigiu o desenvolvimento de métodos e ferramentas para interpretação, mesmo que superficial, de texto em linguagem natural. Como na tarefa de identificação de entidades, pode-se tratar extração de dados sob duas suposições: “hipótese do mundo fechado”, quando os documentos ou arquivos semiestruturados referem-se a um domínio de aplicação bem conhecido; e “hipótese do mundo aberto”, em caso contrário.

As técnicas mais antigas para extração de dados baseiam-se em regras e usam um dicionário específico para o domínio de aplicação em questão para identificar termos e frases no texto. Em particular, enfoques de linguística computacional baseados em regras exploram a estrutura sintática das sentenças para melhorar o processo de extração. Métodos mais recentes baseiam-se em aprendizagem de máquina [5] e, em particular, redes neurais profundas (notadamente *recurrent neural networks* - RNNs e *convolutional neural networks* - CNNs). Os métodos para identificar relacionamentos entre entidades mais bem sucedidos aplicam aprendizagem supervisionada para construir classificadores, que utilizam características (*features*) extraídas de sentenças anotadas manualmente em um corpus de treinamento. Supervisão a distância (*distant supervision*) endereça o problema de gerar automaticamente exemplos, em número suficiente para treinamento dos algoritmos, com a ajuda de bancos de dados do domínio da aplicação.

Workflow pode ser definido como um conjunto de tarefas padronizadas que devem ser feitas para se alcançar um objetivo. Em muitos casos ferramentas disponíveis de workflow auxiliam também processamento paralelo de dados, como por exemplo o Apache NIFI [10].

Este projeto abordará o problema de reconhecimento de entidades nomeadas (NER) sob a “hipótese do mundo fechado”, assumindo a existência de um banco de dados específico para o domínio com as principais entidades e relacionamentos encontrados nos documentos. Além

disso, o projeto contempla o desenvolvimento de um workflow que cuida de todas as tarefas que são necessárias para reconhecer entidades nomeadas em um conjunto de documentos, que estão disponibilizados no file system, e armazenar o resultado do reconhecimento em um banco de dados.

Mais especificamente, este projeto tratará de incorporar técnicas de reconhecimento de entidades nomeadas (NER) nos documentos da plataforma Danke, baseando-se nos dados existentes no banco de dados do Danke. Não é uma interpretação profunda, mas sim o reconhecimento das entidades dentro da *Hipótese do Mundo Fechado*, o que significa que o objetivo não é reconhecer qualquer entidade, mas sim reconhecer, no documento, as entidades que estão representadas no banco de dados do Danke, a ideia é utilizar o banco de dados do Danke como referência para as entidades e os relacionamentos.

3. OBJETIVO E PROPOSTA DO TRABALHO

A proposta deste projeto final consiste em: (1) fazer um levantamento dos requisitos e as funcionalidades do Danke para incorporar documentos textuais, permitindo que as buscas encontrem palavras-chave no conteúdo destes documentos e que o usuário possa encontrar e navegar sobre os relacionamentos destes documentos com as outras entidades do banco de dados; (2) estudar técnicas já existentes para realizar interpretação de documentos, através de reconhecimento de entidades nomeadas em documentos (NER) e de extração de relacionamentos (RE); (3) adotar técnicas já existentes ou propor novas técnicas que sejam mais adequadas para serem incorporadas na plataforma Danke; (4) propor uma evolução da arquitetura do Danke para incorporar estas técnicas; (5) implementar as técnicas propostas; (5) realizar experimentos destas técnicas no Danke com um banco de dados e documentos para algum domínio de aplicação específico; (6) Desenvolver um workflow que envolve todo o processo desde a recuperação dos documentos até o resultado final.

Espera-se que a nova arquitetura do Danke possa aumentar o poder da busca e do acesso aos dados.

4. ATIVIDADES REALIZADAS

Neste projeto, foi feita a especificação da técnica de reconhecimento de entidades em documentos, o desenvolvimento da técnica, a realização de experimentos com a técnica, e o desenvolvimento de um workflow.

Este projeto concentra-se em técnicas de reconhecimento de entidades nomeadas sob a "hipótese do mundo fechado", ou seja, não possuem o objetivo de reconhecer qualquer entidade, mas sim reconhecer, no documento, as entidades que estão representadas no banco de dados.

A técnica desenvolvida neste projeto foi empregada durante um pré-processamento de documentos do Danke, registrando no repositório do sistema os relacionamentos entre os documentos e as entidades reconhecidas.

O Workflow foi criado usando a ferramenta Apache NIFI e envolve as tarefas de extração de texto com e sem OCR e aplicação da técnica NER desenvolvida.

5. PROJETO E ESPECIFICAÇÃO DO SISTEMA

A técnica desenvolvida pode ser dividida em quatro passos principais, descritos abaixo.

A classe *nerAlgorithm*, implementação da interface *inerAlgorithm,* representa o algoritmo NER, responsável por executar estes quatro passos.

Ela é inicializada a partir da classe *NERParameter*, que possui uma lista de *documentCollections*, objeto que contém a entrada do algoritmo, o que inclui os documentos, os indexadores, as entidades e os atributos, conforme será explicado mais à frente.

Para maiores detalhes dessas classes, consulte o Diagrama de Classes [1] no final desta seção.

Os quatro passos principais do algoritmo estão ilustrados no diagrama de sequência a seguir.

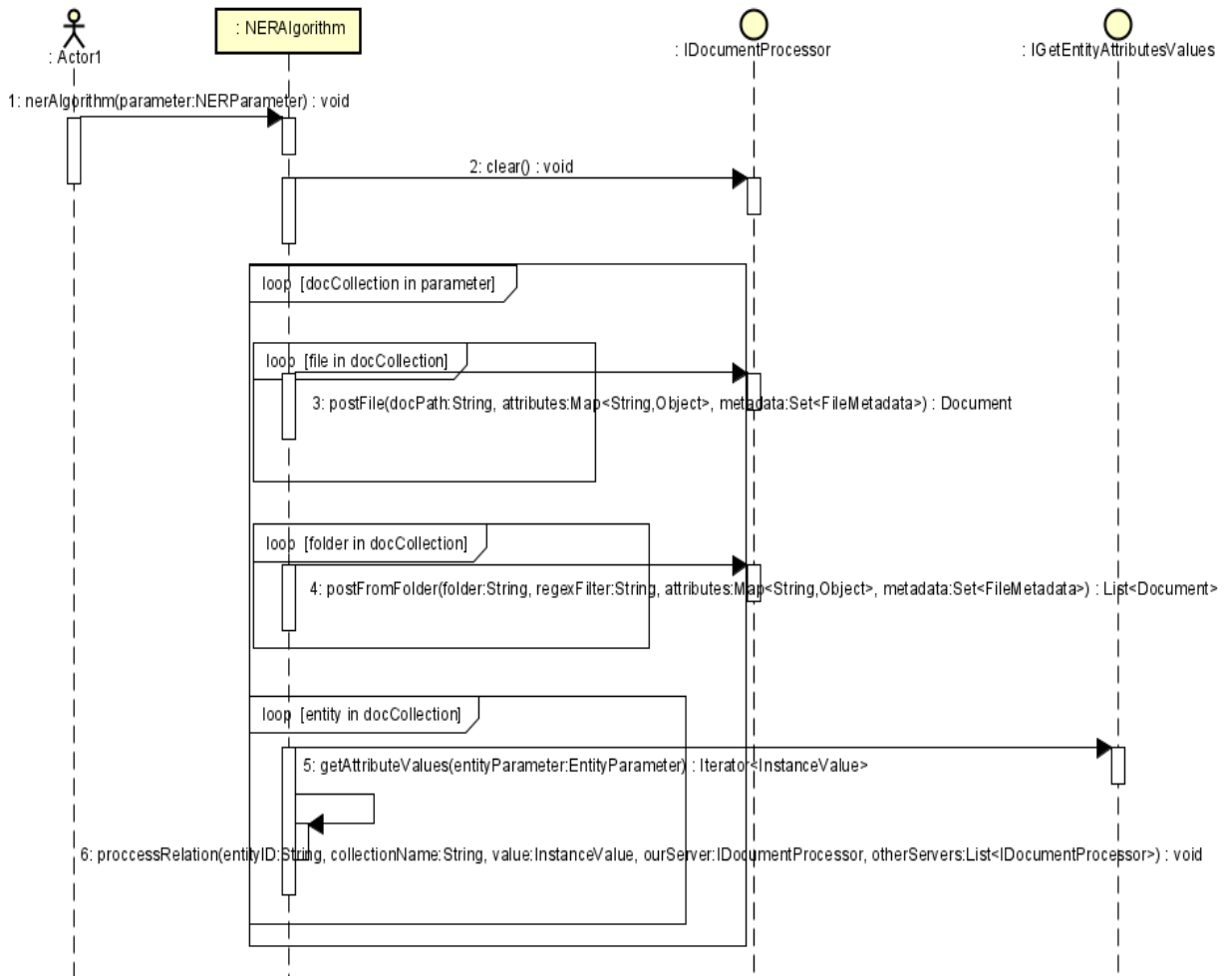

Figura 1. Diagrama de sequência para NERAlgorithm

5.1 PASSO 1: Obtenção dos dados de entrada

Os dados de entrada do algoritmo são: a lista de entidades que precisam ser reconhecidas, um conjunto de documentos (que podem ser disponibilizados como arquivos independentes ou organizados em uma pasta) e uma lista de servidores de indexação. A indexação será realizada utilizando servidores de indexação, como o Apache Solr (Apache Solr 2020) ou Elasticsearch [6] Eles podem já estar acoplados ao Danke ou serem externos. Todos devem ser informados neste primeiro passo.

Abaixo segue o pseudo-código que ilustra a obtenção destes dados de entrada.

```
List-Entities = Get list of entities
List-Files = Get list of all files
List-Folder = Get list of all folders
List-Servers = Get list of servers
```

A classe *nerAlgorithm* utiliza a classe *DocumentsCollections* para fazer chamadas aos métodos *getEntities*, *getFiles*, *getFolder*, *getServers* responsáveis pelas tarefas acima, as entidades são adicionadas ao *DocumentCollections* na forma de um objeto chamado *EntityParameter*, ele tem um identificador e um nome para cada entidade.

Considere um exemplo em que existam relatórios técnicos de uma indústria de óleo e gás, e que o algoritmo precise reconhecer plataformas e equipamentos nestes relatórios. Neste caso, a lista de documentos conterá os arquivos dos relatórios técnicos e a lista de entidades será composta por duas entidades: (1) plataforma; (2) equipamentos. Serão dois objetos *EntityParameter*, um para reconhecer as plataformas e o outro para reconhecer os equipamentos. Ambos terão a mesma lista de documentos.

Para maiores detalhes destas classes, consulte os Diagramas de Classes [1] e [2] no final desta seção.

5.2 PASSO 2: Indexação dos Documento

Todos os documentos recebidos no passo anterior serão indexados utilizando o método *postFile*. Para indexar uma pasta de documentos, o método *postFromFolder* varre todos os documentos da pasta e chama o *postFile* para cada um, indexando, consequentemente, cada documento.

```
FOR each file from List-Files
  postFile(path, attributes, metadata)
END FOR
FOR each folder List-Folder
  postFromFolder(folderPath, attributes, metadata)
END FOR
```

Os métodos *postFile* e *postFromFolder* pertencem à interface *iDocumentProcessor*. Neste projeto, essa interface foi implementada pela classe *DocumentProcessorSolr,* que utiliza o *Solr* como servidor de indexação.

Feita a indexação, este passo gera como saída, para cada documento processado, um objeto da classe *Document* contendo os seus respectivos atributos e metadados.

De forma mais detalhada, a indexação realizada pelo método *postFile* é ilustrada no diagrama de sequência a seguir.

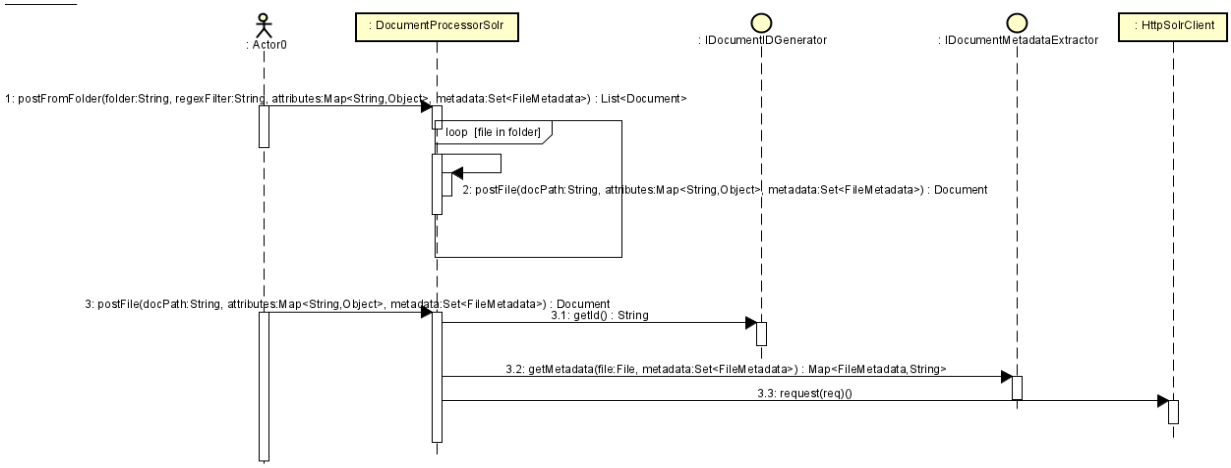

Figura 2. Diagrama de sequência para DocumentProcessorSolr

Conforme pode ser observado pelo pseudocódigo a seguir, o método *postFile* inicia com a definição de um identificador a ser atribuído ao documento e prossegue extraindo os metadados do arquivo, para então efetivamente indexar o documento.

```
postFile(path, attributes, metadata){

ID = getId()
IF metadata is not empty
     THEN getMetadata(file, metadata)
END IF

Post to the indexer a document with the attributes, the
metadata and the ID generated

IF Post was successful
     THEN  Instantiate a Document object with the
     attributes, metadata and the ID generated
END IF
```

```
RETURN Document Object

}
```

Para maiores detalhes, consulte o Diagrama de Classes [3] no final desta seção.

5.3 PASSO 3: Reconhecimento das entidades nomeadas nos documentos

Neste passo, todos os documentos serão analisados em busca do reconhecimento das entidades. O método *processRelation* da classe *nerAlgorithm* é responsável por encontrar e salvar os relacionamentos entre as entidades e os documentos.

Como pode ser observado no pseudocódigo desse passo, o método *processRelation* é chamado para cada instância de entidade que precisa ser reconhecida. Suponha o exemplo citado anteriormente da indústria de óleo e gás, onde deseja-se reconhecer plataformas nos relatórios técnicos. Neste caso, cada plataforma (p_1, p_2 ... p_n) é uma instância da entidade *Plataforma*.

Como entrada, o método *processRelation* recebe o identificador de entidade (*entityID*), uma instância da entidade (*instanceValue*), o nome da coleção de documentos (*collectionName*), e a lista de servidores que possuem tais documentos indexados.

```
FOR each entity remaining
      attributeValues = getEntityAttribute(entity)
      FOR each instanceValue in attributeValues
                              processRelation(entityID,
                              instanceValue , collectionName,
                              index server)
      END FOR
END FOR
```

O método *getEntityAttribute* deve recuperar todas as instâncias de uma entidade e colocá-los num objeto do tipo *iterator* de *instanceValues*.

Para maiores detalhes, veja o Diagrama de Classes [4] no final desta seção.

O método *processRelation* chama o método *search* da interface *iDocumentProcessor* para fazer uma busca de uma entidade nos documentos, indexados previamente pelo servidor de indexação. O método *search* produz como saída uma lista de documentos onde foi reconhecida uma determinada entidade. Mais precisamente, cada documento é um objeto da classe *Document*.

Para maiores detalhes, veja o Diagrama de Classes [2] no final desta seção.

5.4 PASSO 4: Geração dos dados de saída

Após descobrir a associação de uma determinada entidade com uma lista de documentos (no passo anterior), o algoritmo salva tal relacionamento. O método *processRelation* chama o método *storageRelation*, da interface *IStorageEntityRelation*, responsável por salvar os relacionamentos encontrados entre um documento e uma entidade. A saída deste método *processRelation* é um *boolean* que indica *true* caso tenha obtido sucesso em sua tarefa. No final do método *processRelation* será gerado um arquivo CSV ou mesmo como uma tabela de um banco de dados com o registro de todos os relacionamentos entre documentos e entidades encontrados.

Para maiores detalhes, veja o Diagrama de Classes [5] no final desta seção.

5.5 Diagramas de Classes

Diagrama de Classes [1]: Parâmetros para o Algoritmo de Named Entity Recognition

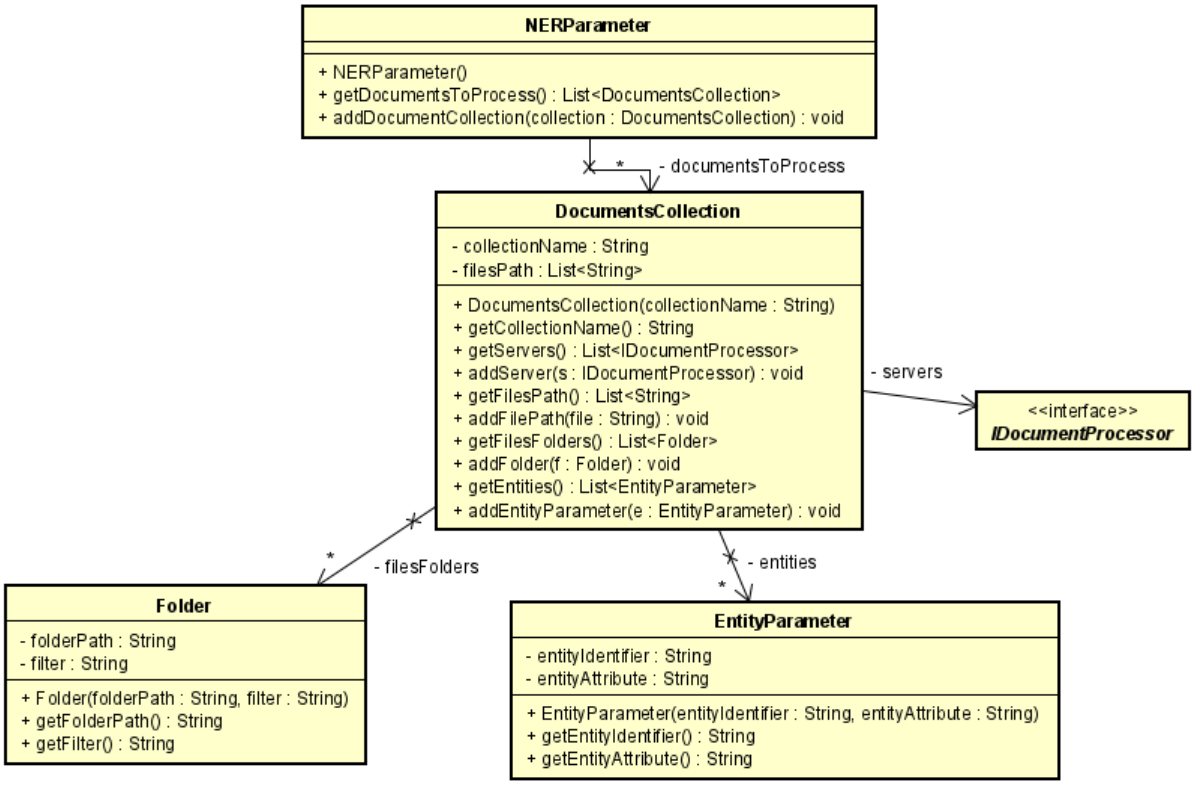

Figura 3. Diagrama de Classe do NERParameter

Diagrama de Classes [2]: Algoritmo Named Entity Recognition

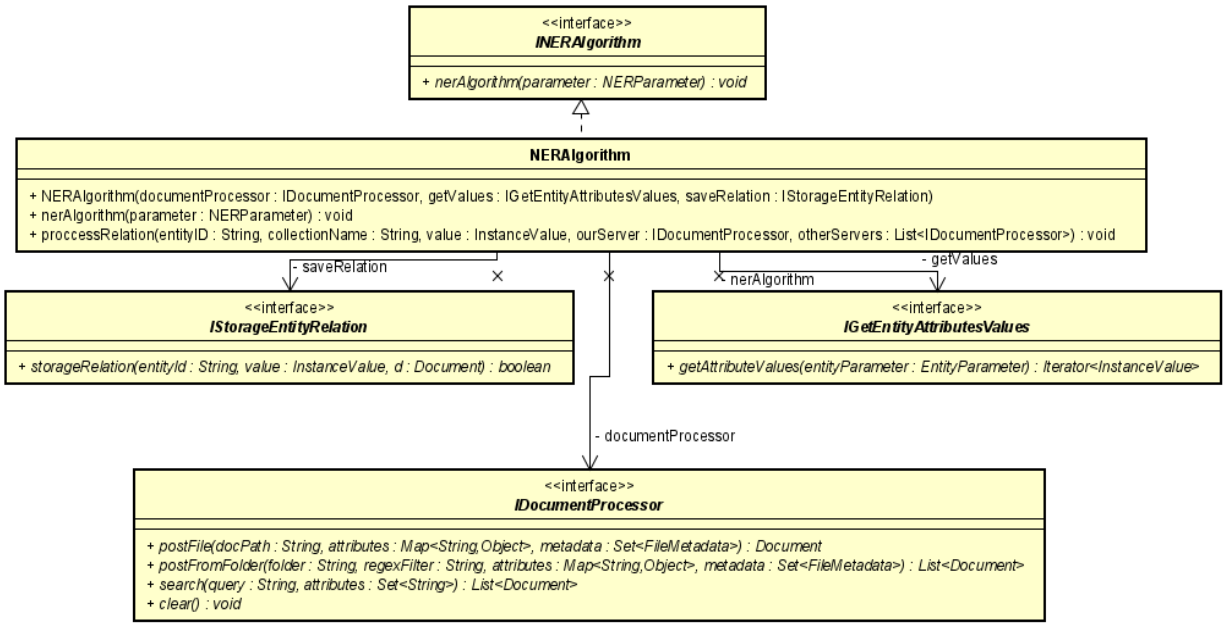

Figura 4. Diagrama de classe do NERAlgorithm

Diagrama de Classes [3]: Processamento dos Documentos

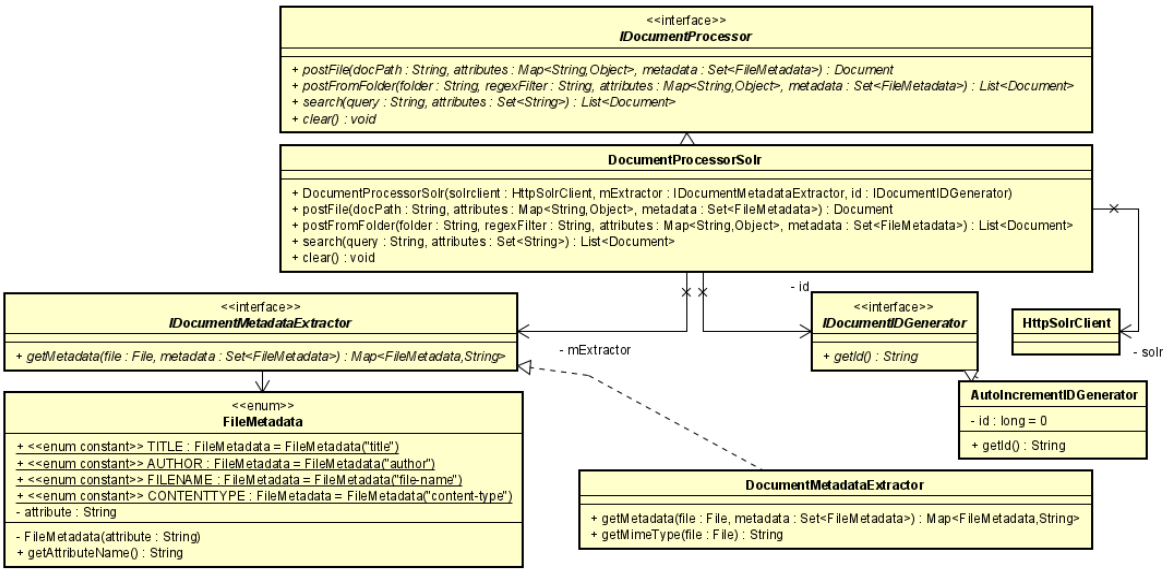

Figura 5. Diagrama de classe do DocumentProcessor

Diagrama de Classes [4]: Entidades e Valores

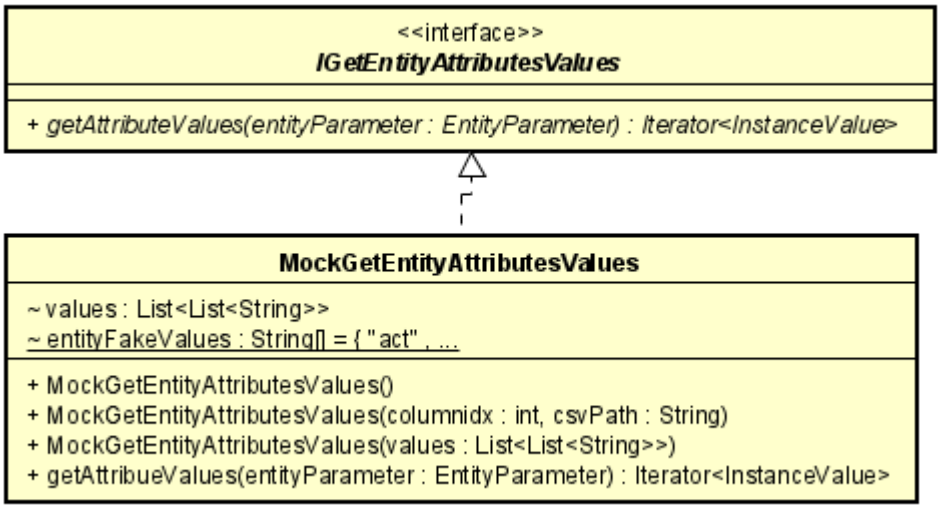

Figura 6. Diagrama de classe do GetEntityAttributesValues

Diagrama de Classes [5]: Armazenamento dos Relacionamentos

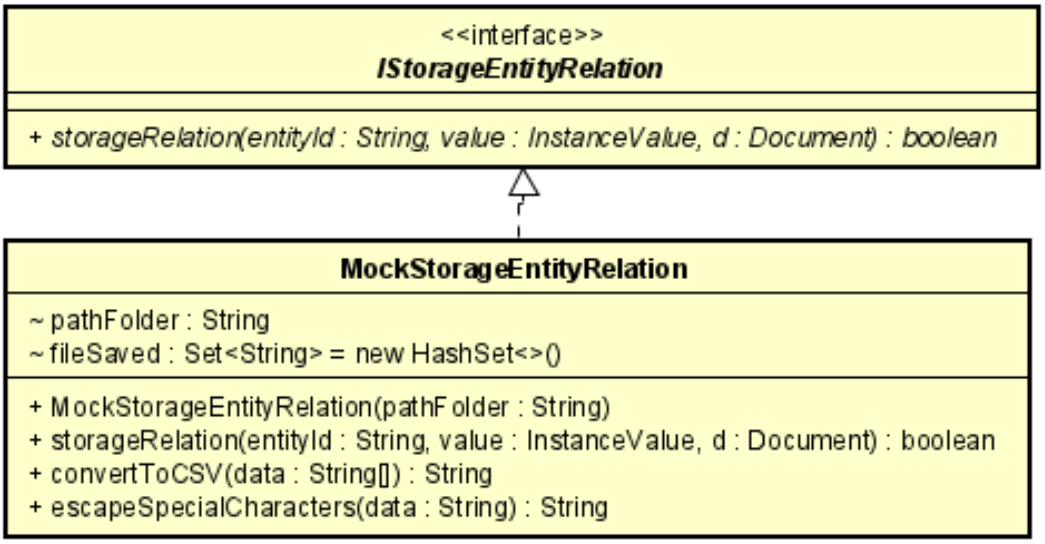

Figura 7. Diagrama de classe do StorageEntityRelation

5.6 Teste e protótipo

Para realizar testes com o algoritmo de reconhecimento de entidades, foram usados dados de uma indústria de óleo e gás, com o objetivo de reconhecer as entidades plataformas e equipamentos em relatórios técnicos em PDF.

Foram implementadas as classes *MockGetEntityAttributeValues* e *MockStorageEntityRelation*, que podem ser observadas nos Diagramas de Classes [4] e [5].

A classe *MockGetEntityAttributeValues* tem o objetivo de receber a lista de entidades a serem reconhecidas. No caso, foram criados dois arquivos CSV com informações sobre as plataformas (ex. plataforma.csv) e sobre os equipamentos (ex. equipamento.csv). Como estes dados são confidenciais, não foi possível apresentar neste documento.

A classe *MOCKStorageEntityRelation* tem o objetivo de salvar em um arquivo CSV os relacionamentos entre os relatórios técnicos e as entidades acima.

Como os resultados dos testes não podem ser mostrados devido a confidencialidade dos dados e da ferramenta, foi construído um mockup replicando os pontos chaves desses resultados.

Os dois próximos mockups mostram o resultado da consulta pelas palavras-chave “relatório equipamentos”, sendo que o primeiro ilustra o resultado antes de aplicar a técnica de NER desenvolvida neste projeto e o segundo já mostra após o resultado do NER ser aplicado. Vemos no lado direito um grafo que representa o caminho realizado pelo Danke para conseguir entregar o resultado. Nota-se que no grafo sem a aplicação do NER, para chegar nos Relatórios precisamos passar por Procedimento, Plataforma e Serviço, enquanto que depois da aplicação do NER o grafo ficou muito mais simples, tendo um relacionamento direto entre os relatórios e equipamentos.

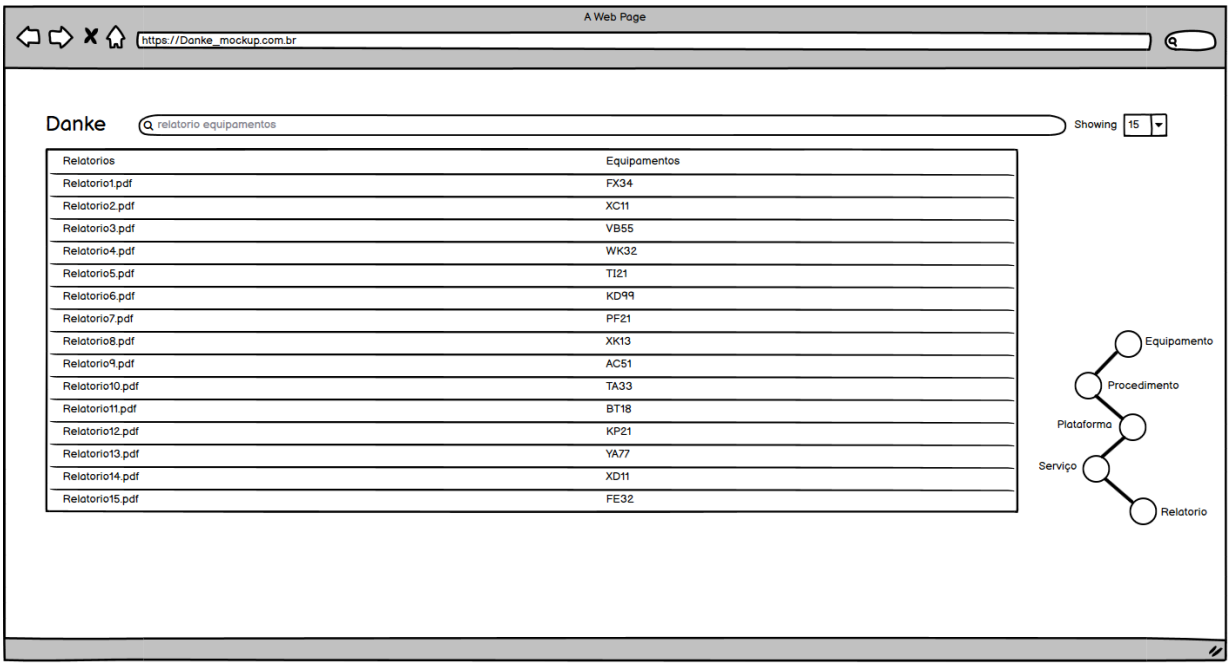

Figura 8. Consulta "relatório equipamentos" sem o NER

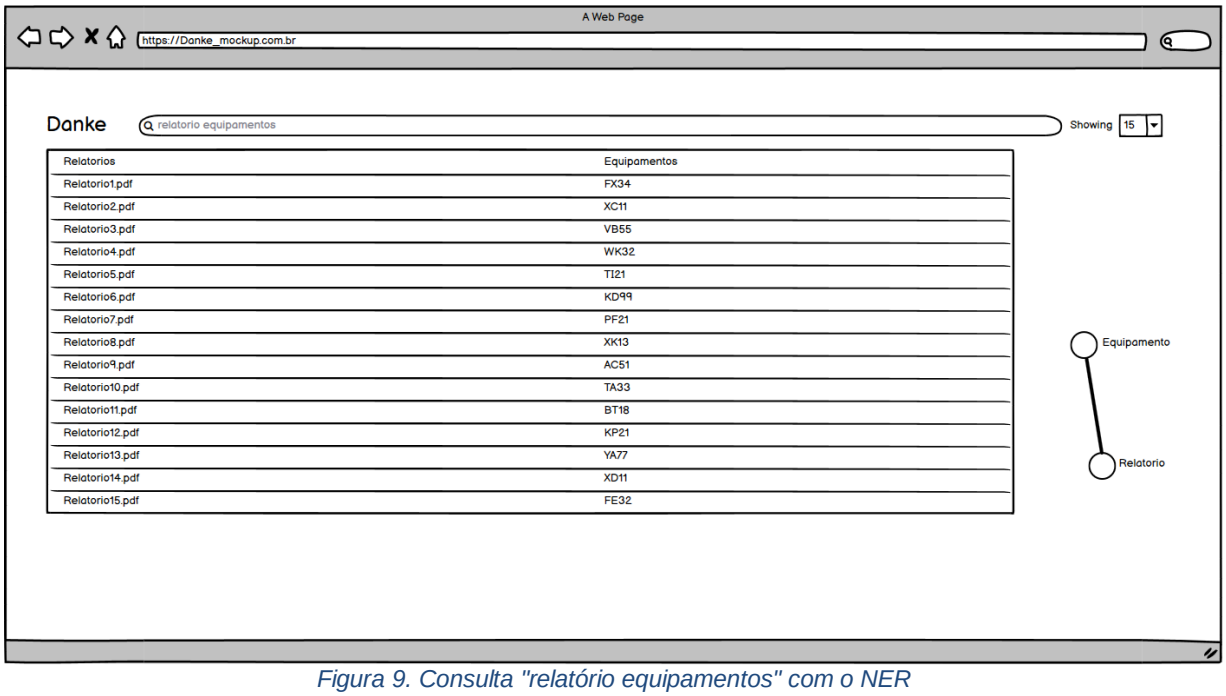

Figura 9. Consulta "relatório equipamentos" com o NER

Além disso, podemos notar nas imagens abaixo que, quando o NER foi aplicado em uma consulta por um equipamento específico, é possível visualizar todos os relatórios que possuem esse equipamento, devido ao relacionamento criado pelo NER.

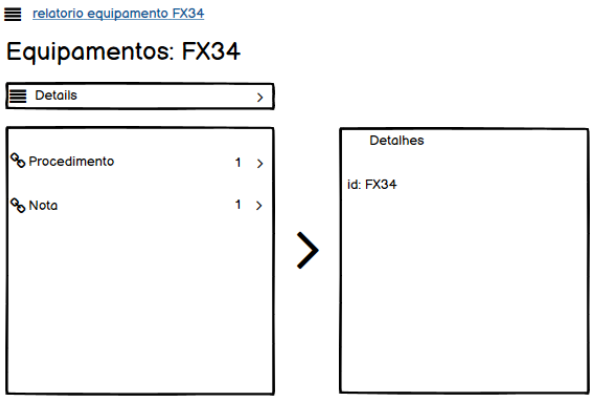

Figura 10. Detalhes de uma consulta por equipamento sem o NER

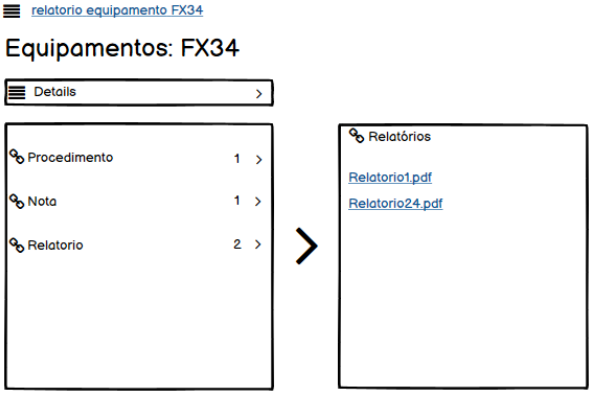

Figura 11. Detalhes de uma consulta por equipamento com o NER

Também pode-se notar nos mockups abaixo que, no caso de uma consulta para um relatório específico, ao aplicar o NER, o sistema passa a apresentar uma lista com todos os equipamentos presentes no relatório.

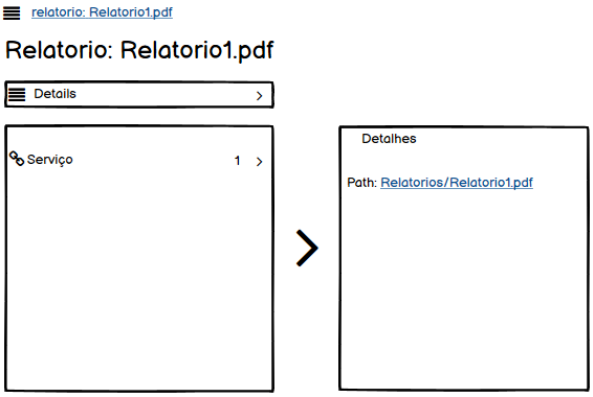

Figura 12. Detalhes de uma consulta por relatório sem o NER

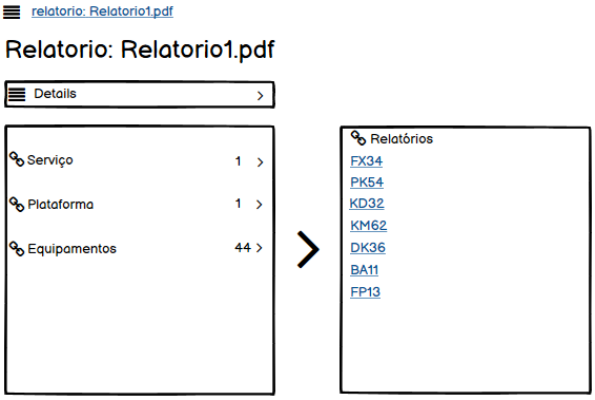

Figura 13. Detalhes de uma consulta por relatório com o NER

5.7 Desenvolvimento do Workflow

Para aplicar as técnicas de NER em um conjunto de documentos PDFs presentes em ambientes reais das indústrias, observou-se que é necessário lidar com um grande volume de documentos

e que eles, em alguns casos, são digitalizados, ou seja, não são sempre textuais. Desta forma tornou-se importante considerar tanto o processo de extração de texto de imagens e a paralelização da execução do NER, otimizando a interpretação de um grande número de documentos. O processo de extração de texto será feito utilizando a ferramenta Apache Tika [12], em conjunto com Tesseract [13]. Para isso, foi adotado um modelo pré-treinado para o português, a ser utilizado pelo Tesseract.

O diagrama básico a seguir ilustra o que é necessário para o workflow de interpretação de documentos, considerando os pontos destacados acima: (1) Obtenção dos documentos; (2) Reconhecimento do tipo do arquivo (Imagem ou texto); (3) Extração do texto dependendo do tipo do arquivo, ou seja, se o documento for do tipo imagem, utiliza-se OCR; (4) Aplicação do algoritmo do NER.

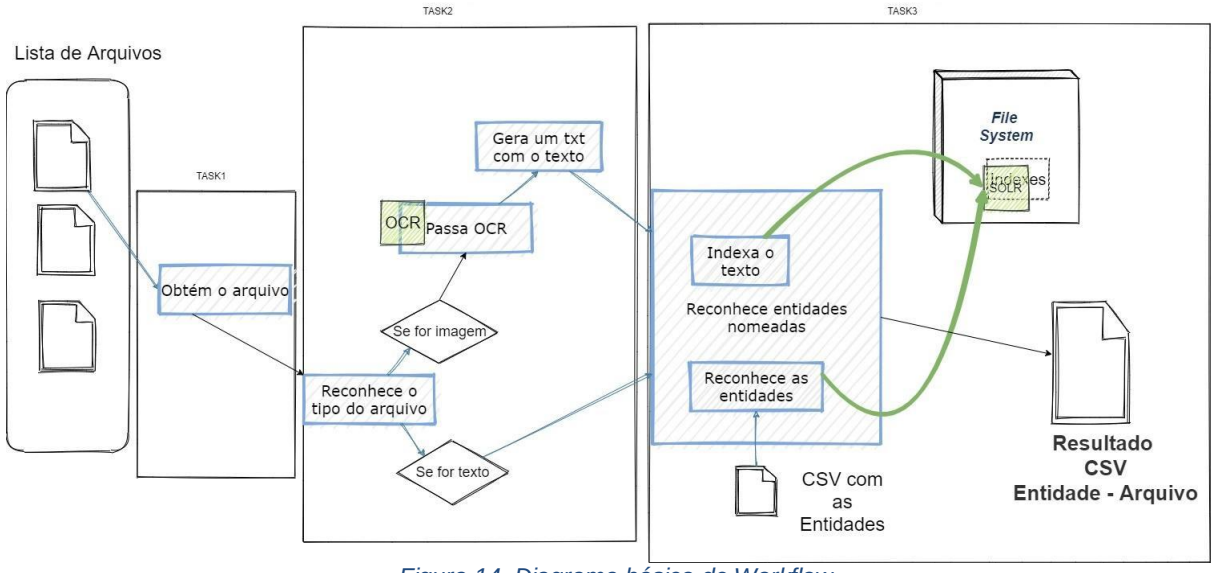

Figura 14. Diagrama básico de Workflow

A implementação deste workflow foi feita utilizando Apache NIFI, que oferece formas de controlar todo o fluxo de dados, recuperar documentos, execução do NER e a manipulação do output gerado.

O Apache NIFI faz uso de 3 importantes estruturas:

- ***Processor***: Este é um elemento que faz uma tarefa simples, o NIFI possui uma grande quantidade de *processadores* configuráveis como por exemplo o "*GetFile*", que é capaz de recuperar vários arquivos de uma pasta e passar seus dados adiante.
- ***FlowFile***: É uma abstração do dado no ambiente, um *processador* pode gerar um *Flowfile* a partir de uma fonte de dados ou ele pode transformar um *Flowfile* em outro.
- ***Connection***: É a conexão entre dois *processadores*, cada *connection* gera uma fila e nela é possível ver os *Flowfiles* enfileirados esperando para ser processados e também seu tamanho em bytes.

A imagem abaixo apresenta a implementação do workflow, ilustrado anteriormente, feita no Apache NIFI. Cada uma dessas caixas é um *processor* e entre elas com uma seta estão as *Connections*.

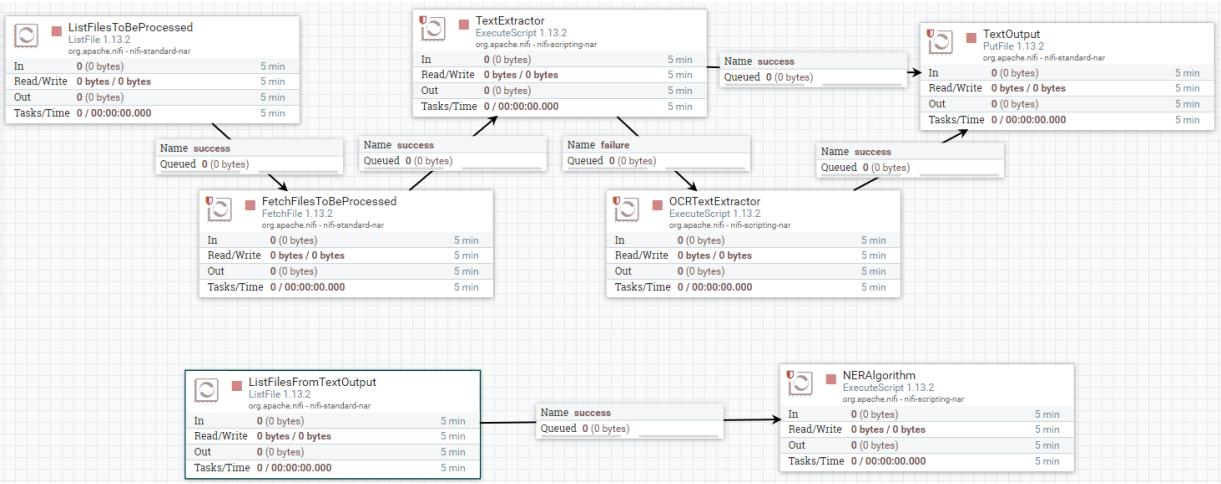

Figura 15. Visão geral da implementação do Workflow no NIFI

Os dois primeiros processadores, "*ListFilesToBeProcessed*" e "*FetchFilesToBeProcessed*" são responsáveis por recuperar os documentos que irão ser processados de uma pasta, vale ressaltar que eles poderiam ser trocados por processadores que recuperam documentos de um banco de dados, ou até mesmo de um FTP.

O processador "*TextExtractor*" extrai o texto de um documento e faz uma contagem de seus caracteres, se esse número for muito baixo determinamos que devemos passar o OCR pois ele possui grandes chances de ser do tipo imagem, nesse caso o documento original é passado para o "*OCRTextExtractor*".

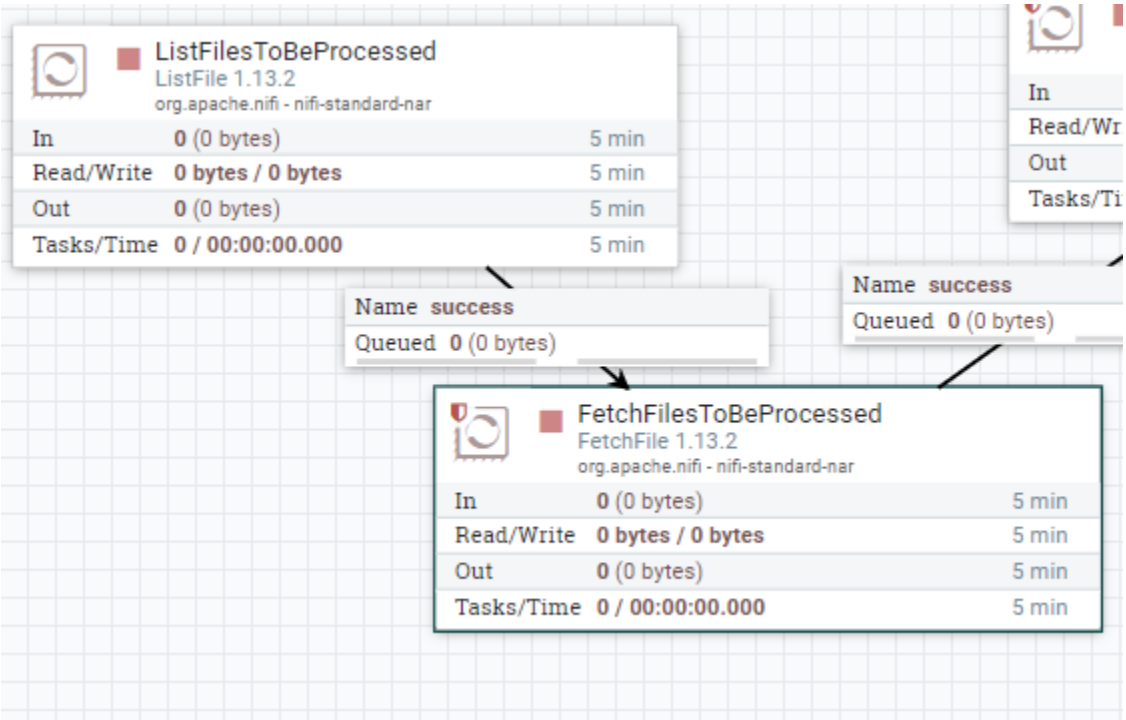

Figura 16. Processadores de recuperação dos documentos

No caso de sucesso para os *Processors* "*TextExtractor*" e "*OCRTextExtractor*" ambos irão gerar uma saída txt e passar para o processador "*TextOutput*", que irá guardar todos esses resultados em uma pasta para ser consumidos em breve.

Finalmente, na parte de baixo da imagem, temos um processador que vai listar todos os nomes dos arquivos contidos na pasta do "*TextOutput*" e passá-los para o processador responsável pelo NER, este faz uso de um script que importa um *.jar* do NER e assim faz sua execução.

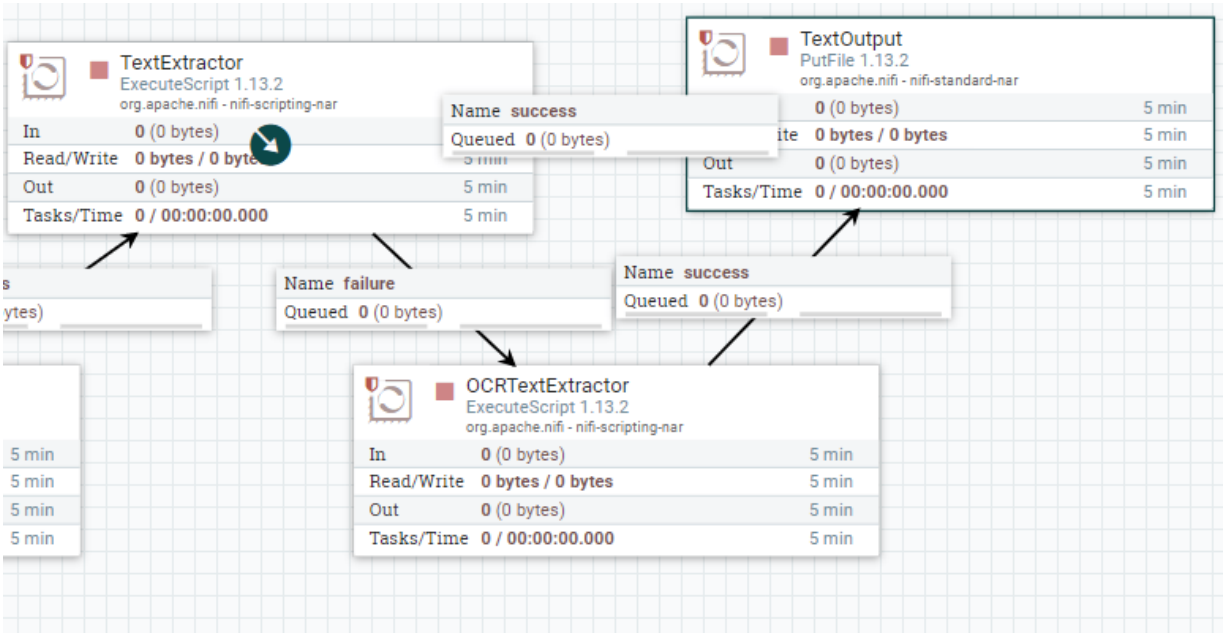

Figura 17. Processadores de extração de texto e output

Algumas configurações necessárias para o NER, como a conexão com o Solr e as listas com as entidades (CSVs) foram programadas dentro desse processador e assim podem ser recuperadas pelo script.

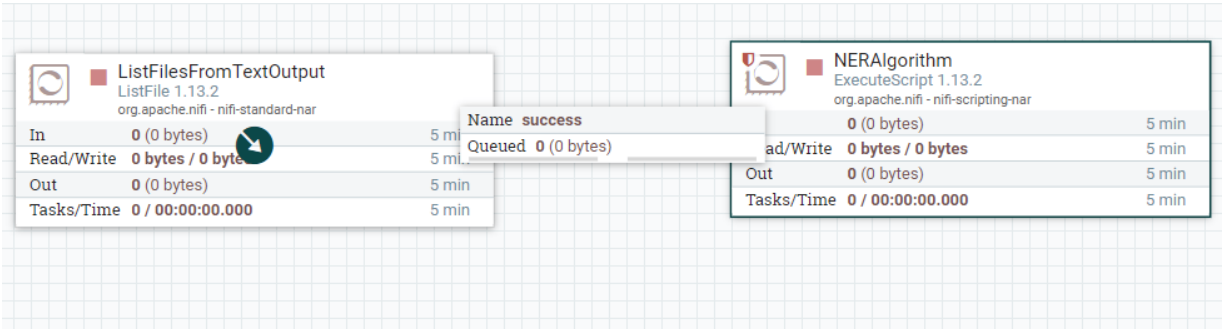

Figura 18. Processadores de listagem de arquivos e algoritmo NER

6. ETAPAS DO PROJETO

Este projeto dividiu-se nas seguintes etapas:

1) Levantar requisitos e funcionalidades do Danke para incorporar documentos textuais, permitindo que as buscas encontrem palavras-chave no conteúdo destes documentos e que o usuário possa encontrar e navegar sobre os relacionamentos destes documentos com as outras entidades do banco de dados.
2) Registrar o levantamento realizado em um documento.
3) Estudar tecnologias e técnicas já existentes para realizar interpretação de documentos, através de reconhecimento de entidades nomeadas em documentos (NER).
4) Registrar o estudo realizado em um documento.
5) Incorporar técnicas para NER ao Danke:
    a) Propor ou criar novas técnicas NER que sejam mais adequadas para serem incorporadas na plataforma Danke.
    b) De acordo com o nível de complexidade, definir etapas e o que será efetivamente implementado destas técnicas na plataforma Danke durante este projeto final.
    c) Propor e documentar uma evolução da arquitetura do Danke para incorporar estas técnicas NER.
    d) Implementar as técnicas NER definidas.
    e) Realizar experimentos destas técnicas NER no Danke com um banco de dados e documentos para algum domínio de aplicação específico.
    f) Criar um mockup da aplicação do Danke para poder apresentar os resultados sem quebrar o sigilo dos dados.
6) Criar um Workflow para aplicar as técnicas produzidas, considerando um ambiente real da indústria, que precisa paralelizar o processamento de interpretação de documentos, devido ao grande volume de documentos, e que precisa considerar que em alguns casos os documentos são digitalizados, necessitando então extração de texto de imagens.
    a) Propor um diagrama representativo para o workflow, separando as tarefas necessárias para o processo completo de interpretação de documentos.
    b) Pesquisar opções de ferramentas para o desenvolvimento do Workflow.
    c) Implementar e testar o Workflow.
7) Criação de um documento detalhado de todo o projeto.

Cabe ressaltar que o plano original deste projeto final contemplava estender as técnicas de NER para reconhecer relacionamentos entre as entidades. Entretanto, ao analisar as necessidades do mundo real, percebemos que seria mais relevante tratar questões do volume de dados e da diversidade dos dados dos documentos. Sendo assim, optou-se por investir na pesquisa e testes de ferramentas de criação de workflow, que possibilitasse encadear todas as tarefas de interpretação de documentos, além de estar pronto para uma futura paralelização de todo o seu processamento para melhorar seu desempenho e escalabilidade.

Finalmente, no projeto final 1, foi feito uma série de testes, que não foram apresentados neste documento devido à confidencialidade dos dados envolvidos. Portanto, neste documento foi apresentado uma série de mockups da plataforma Danke ilustrando os resultados obtidos com a aplicação da técnica desenvolvida neste projeto.

Cronograma

Etapas	Set	Out	Nov	Dez	Jan	Fev	Mar	Abr	Mai
Etapa 1	x	x							
Etapa 2		x							
Etapa 3			x	x					
Etapa 4				x					
Etapa 5				x	x	x			
Etapa 6						x	x	x	x
Etapa 7		x	x	x	x	x	x	x	x

7. CONSIDERAÇÕES FINAIS

A técnica para NER implementada apresentou bons resultados, melhorando a experiência dos usuários do Danke.

O Workflow implementado apresentou um desempenho bastante lento pois o algoritmo do NER não foi implementado para ser chamado uma vez para cada documento e, sim, uma vez

para um grande conjunto de documentos. Portanto, o algoritmo de NER executa consultas para todas as entidades de uma lista, multiplicadas pelo número de documentos, aumentando a ordem da complexidade computacional do algoritmo.

Os testes realizados foram todos feitos através da geração e a leitura de arquivos do tipo CSV. Além disso, não foram realizados testes com processamento paralelo e o uso de clusters, que é um dos motivos pela escolha do Apache NIFI. Em um trabalho futuro, pretende-se investir na otimização deste algoritmo, novos testes que utilizam um banco de dados e um ambiente computacional que permita paralelizar os processos.

8. REFERÊNCIAS BIBLIOGRÁFICAS

[1] Danke: Data and Knowledge Retrieval. *http://danke.tecgraf.puc-rio.br*. Acesso em Outubro/2020.

[2] Instituto Tecgraf: Instituto de desenvolvimento e pesquisa da PUC-Rio. *http://www.tecgraf.puc-rio.br*. Acesso em outubro/2020.

[3] Nadeau, David; Sekine, Satoshi. *A survey of named entity recognition and classification*. National Research Council Canada / New York University. Acesso em Outubro/2020

[4] Herman, Andreas. *Different ways of doing Relation Extraction from text*. Acesso em Outubro/2020.

[5] Dong, X.L and Rekatsinas, T. "Data Integration and Machine Learning: A Natural Synergy". VLDB 2018. Acesso em Outubro/ 2020.

[6] Gormley, Clinton; Tong, Zachary. *Elasticsearch: The Definitive Guide*. Acesso em Outubro/2020.

[7] Smiley, David; Pugh, Eric; Parisa, Kranti; Mitchell, Matt. *Apache Solr Enterprise Search Server*. Acesso em Outubro/2020.

[8] Izquierdo, Y. T.; García, G. M.; Cavaliere, M. L.; Novello, A.; Novelli, B. A.; Damasceno, C. O.; Leme, L.A.P.P.; Casanova, M. A. . Comparing and Recommending Conferences. In: Brazilian Symposium on Databases - SBBD, 2020 (To be published). Acesso em Outubro/2020.

[9] García, G.M. A Keyword-based Query Processing Method for Datasets with Schemas. Thesis presented to the Graduate Program in Informatics, PUC-Rio (March 2020). DOI: *https://doi.org/10.17771/PUCRio.acad.48728*. Acesso em Outubro/2020.

[10] [online] Disponível em: *https://nifi.apache.org*. Acesso em Maio/2021.

[11] [online] Disponível em: *https://en.wikipedia.org/wiki/Workflow*. Acesso em Maio/2021.

[12] [online] Disponível em: *https://tika.apache.org*. Acesso em Junho/2021.

[13] C. Patel; A. Patel; D. Patel. Optical Character Recognition by Open Source OCR Tool Tesseract: A Case Study. Disponível em: *International Journal of Computer Applications (0975 – 8887) Volume 55– No.10, October 2012*. Acesso em Junho/2021.