Universidade do Minho
Escola de Engenharia

Alexandre Carrizo Gomes

Processo de Digitalização de Contratos Legais: Criação de Smart Templates

Outubro de 2022

Processo de Digitalização de Contratos Legais: Criação de Smart Templates

Alexandre Carrizo Gomes

UMinho |

Universidade do Minho
Escola de Engenharia

Alexandre Carrizo Gomes
(84188)

Processo de Digitalização de Contratos Legais: Criação de Smart Templates

Dissertação de Mestrado
Mestrado [integrado] em Engenharia e Gestão de Sistemas de Informação

Trabalho efetuado sob a orientação do
Professor Doutor Miguel Abrunhosa de Brito

mês de 20X1

DIREITOS DE AUTOR

AGRADECIMENTOS

Esta dissertação é o produto de um caminho longo e àrduo que contou com o contributo não só de seu autor, mas também de várias pessoas que o influenciaram. Agradeço às seguintes pessoas pelo seu contributo para a materialização desta investigação:

Ao meu orientador, o Professor Doutor Miguel Abrunhosa de Brito pelas críticas construtivas e pela sua disponibilidade,

à minha familia, especialmente aos meus pais e irmã, Virgílio Gomes, Maria José Carrizo, e Diana Carrizo Gomes,

à Paula Marques,

ao João Miguel Soares,

ao Bernard Georges, Pedro Dantas e Sofia Neto,

à Ana Paula Amorim e ao Serafim Gomes,

aos meus colegas de curso,

e por último ao meu adorável gato Max

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio, nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

RESUMO

Processo de Digitalização de Contratos Legais: Criação de Smart Templates

O *Smart Legal Contract* é uma tecnologia recente utilizada para representar e gerir, através de um ambiente digital, um contrato legal. Estes SLCs, para além de legalmente válidos, apresentam a capacidade de autoexecução e autogestão, seguindo um conjunto de regras que permite executar programaticamente as cláusulas legais que este apresenta. No entanto, na atualidade ainda não existem tecnologias de apoio para a transformação de documentos em texto simples para um formato de *Smart Legal Contract.*

Recorrendo ao *Accord Project*, uma iniciativa que tem como objetivo a estandardização de um formato específico de *Smart Legal Contracts*, este trabalho pretende criar um processo/algoritmo, ou conjunto destes, que permita digitalizar um contrato legal em formato de *word/pdf* para o formato de SLC estabelecido pela iniciativa.

Para atingir este objetivo, serão explorados diversos métodos de extração e análise de dados de documentos como *Natural Language Processing* e vários algoritmos de *Machine Learning* com ênfase em *Document Understanding*, que por si, dão a origem à criação de *Smart Templates*, que guardam a informação e conjuntos de regras que definem a estrutura de um tipo de contrato. Os documentos de contratos legais serão posteriormente adaptados para o formato de SLC, recorrendo aos *Smart Templates* previamente criados como base e a uma biblioteca de cláusulas legais executáveis, através de métodos de geração de código automáticos.

Palavras chave: Smart Legal Contracts, Smart Templates, Document Processing, Accord Project.

ABSTRACT

Digitalization Process for Legal Contracts: Smart Template Creation

The Smart Legal Contract is a recent technology used to represent and manage, through a digital environment, a legal contract. These SLC, besides being legally valid, have the ability to self-execute and self-manage, following a set of rules that allow the contract to programmatically execute the legal clauses that it presents. However, at present time, there are still no supporting technologies for the transformation of plain text documents into a Smart Legal Contract format.

Using the Accord Project, an initiative that aims to standardize a specific format for Smart Legal Contracts, this work intends to create a process/algorithm, or a set of these, that allows the digitization of a legal contract in a word/pdf format into the SLC format established by the initiative.

To achieve this goal, several methods of extracting and analyzing data from documents will be explored, such as Natural Language Processing and several Machine Learning algorithms with emphasis on Document Understanding, which in themselves give rise to the creation of Smart Templates, which store the information and sets of rules that define the structure of a contract type. The legal contract documents will later be adapted to the SLC format, using the Smart Templates previously created as a basis and a library of executable legal clauses, through automatic code generation methods.

Keywords: Smart Legal Contracts, Smart Templates, Document Processing, Accord Project.

ÍNDICE

LISTA DE ABREVIATURAS/SIGLAS

SLC Smart Legal Contracts

OCR Optical Character Recognition

RPA Robotic Process Automation

LISTA DE FIGURAS

LISTA DE TABELAS

1. INTRODUÇÃO

1.1 Apresentação da Entidade de Acolhimento

A empresa acolhedora foi a DigitalSign, uma empresa de certificados digitais com sede em Guimarães, Portugal. O projeto foi sugerido inicialmente pela empresa, sendo que existe um interesse intrínseco, por parte da organização, nas funcionalidades desenvolvidas neste trabalho. Foi lançado o desafio inicial com uma proposta de investigação e aplicação do *Accord Project*.

1.2 Enquadramento

Apesar da transição gradual do papel para um ambiente digital, todos os dias milhares de contratos legais ainda são escritos, partilhados, assinados e geridos de forma manual através de documentos em texto simples. Existem inúmeras ferramentas e aplicações hoje em dia que nos permitem facilitar o processo de criação destes documentos de texto, através do preenchimento automático de campos num documento base. Os últimos anos têm sido marcados por um interesse crescente na aplicação de técnicas modernas aos problemas de Processamento da Linguagem Natural presentes no sector jurídico. As aplicações de automatização de documentos jurídicos estão indiscutivelmente entre os primeiros sistemas comerciais de geração de linguagem natural, que sempre funcionaram com base em sistemas baseados em modelos, nos quais um utilizador preenche um formulário simples para gerar automaticamente novos documentos contratuais. Foi relatado pela Deloitte em 2016 que cerca de 40% dos empregos no sector jurídico poderiam ser automatizados nos dez anos seguintes, e as estimativas confirmam que em geral, 22%-35% de um emprego no sector jurídico poderia ser automatizado (Dale, 2019).

Verifica-se que, a informação de gestão existente nos contratos legais perde-se no próprio meio de criação destes documentos. Não há maneira fácil de aceder aos dados referentes ao conteúdo de um contrato através do próprio documento, visto que estes estão escritos em texto simples que não é facilmente legível e manipulável por uma máquina. Os trabalhadores

do domínio jurídico estão habituados a trabalhar em contratos armazenados principalmente em formato *Word* e *PDF*, que precisam de ser digitalizados e integrados no seu sistema de gestão documental existente. A partir de 2019, não existe nenhum sistema que possa extrair automaticamente conteúdo e cláusulas destes documentos (Rao & Havewala, 2019).

Este processo de criação de contratos causa então, por um lado, uma dependência em *software* especializado exterior aos próprios documentos com limitações, e por outro, uma necessidade de inserção manual destes dados, obrigando a reinserção de informação, que por sua vez aumenta o tempo e esforço requeridos para a digitalização e gestão de um simples contrato.

Esta necessidade é respondida pelos *Smart Legal Contracts*: contratos digitais autónomos e autoexecutáveis. Este novo tipo de documento digitalizado tem várias aplicações práticas e desafios que terão o potencial de revolucionar o status quo dos contratos tradicionais em papel. A questão com os *Smart Contracts* das *Blockhains* foi que não eram legalmente reconhecidos. Tendo isto em mente, qualquer contrato que seguisse um conjunto de regras e princípios definidos pelo sistema legal, seria rotulado como *Smart Legal Contract*, representando a sua autoridade dentro da lei (Pereira, 2019).

Com o surgimento desta nova tecnologia surge também um problema: não existe nenhum processo automatizado que permita uma digitalização de contratos legais em documentos de texto para o formato executável dos *Smart Legal Contracts*.

1.3 Objetivos da dissertação

Clarificando a questão, o principal objetivo deste projeto é então a idealização e criação de um processo de digitalização de contratos legais, visto que, sendo uma tecnologia recente e com falta de exploração no mercado de trabalho, há uma demanda para estes algoritmos que não está a ser atendida. Também se deve ter em conta a optimização monetária e temporal da solução criada, devido à perspetiva empresarial presente.

Para uma melhor compreensão do trabalho, este divide-se em subobjetivos. Em primeiro lugar, será necessário investigar e analisar as tecnologias de extração de dados de documentos existentes no mercado atual, originando as seguintes tarefas:

- Investigar tecnologias atuais para estruturar documentos legais em formatos digitais executáveis.

- Investigar e analisar bibliotecas open source de *Natural Language Processing, Document Understanding* e de outros tipos de compreensão de texto.

- Idealizar e planear um ou mais processos de digitalização de fácil utilização por parte do utilizador final.

- Manter o custo da solução, quer este seja monetário ou temporal, o mais baixo possível.

Numa segunda fase, idealiza-se a construção de *Smart Templates*, que contêm todos os dados referentes ao conteúdo do contrato e meta-dados gerados em função destes para apoiar a gestão dos documentos finais. Estes *Smart Templates* servirão como uma base computarizada para a criação de novas instâncias de um contrato, facilitando a digitalização de novos documentos legais. Para tal, será necessário:

- Construção de uma prova de conceito que demonstre um ou mais processos de digitalização previamente projetados.

- Validação do processo e prova de conceito desenvolvidos com as organizações interessadas.

Na última fase, serão concepcionados melhoramentos futuros ao processo e apontados os problemas e resoluções destes que ocorreram durante o desenvolvimento do projeto.

1.4 Metodologia de Investigação

Inicialmente, o modo de investigação abordado foi a investigação aplicada, pois este projeto utiliza conhecimento já existente para atingir um resultado pragmático, a criação de um processo de digitalização de contratos legais. Foram estudados vários artigos científicos e tecnológicos que fundamentam as várias decisões realizadas durante o decorrer do projeto, especialmente no capítulo 5. Também foi realizada uma investigação simples do mercado tecnológico atual e como estes impactam o projeto nos capítulos 2, 3 e 4.

O projeto utiliza uma metodologia denominada de *Design Science Research*, devido ao seu foco em desenvolver artefactos em forma de modelos, métodos e processos que servem como soluções para problemas empresariais. Neste caso, será usado para resolver o problema definido anteriormente no capítulo 1.3. Estes artefactos estão representados nos capítulos 5, 6 e 7.

2. SMART LEGAL CONTRACTS E O PROJETO ACCORD

2.1 Smart Legal Contracts

Os *Smart Legal Contracts* ou *SLC* são representações digitais executáveis de contratos legais, e como tal, é importante distingui-los dos *Smart Contracts.* Os *Smart Contracts* são documentos legíveis por máquina inseridos em tecnologia de *blockchain*, que implementam uma linguagem de contratação comum de alto nível tornando-os tanto legíveis por máquinas como por humanos, mas que não têm valor como contratos legais (Norta, 2018). Estes *SLC* têm como objetivo fornecer um formato digital que seja juridicamente vinculativo, incorporando algumas ou todas as suas cláusulas como código legível por máquinas e executando-as automáticamente. Após uma exploração dos efeitos da ambiguidade contratual no seu relativo código legível por máquina, verificou-se que as interpretações mais genéricas eram mais complexas quando escritas num formato de contrato inteligente, o que significa que quanto menos ambíguo for um contrato legal, mais simples será a sua contrapartida de contrato digital (Upadhyay et al., 2020). A empresa acolhedora propôs investigar o *Accord Project* como ponto de partida para a gestão de *SLC*, pois o projeto oferece um formato comum globalizado para contratos legais digitais, simplificando o código e reduzindo a ambíguidade.

2.2 O que é o Projeto Accord?

O *Accord Project* é um projeto *open source* criado em 2017, cujo objetivo estabelecer e manter uma fundação tecnologica e legal para a gestão e execução digital de *Smart Legal Contracts*. O projeto contém todas as funções de software necessárias para a criação, edição, gestão e funcionamento destes contratos digitais, permitindo a desenvolvedores e utilizadores criar documentos legalmente válidos que podem ser executados, reusados e partilhados em plataformas digitais. Para a representação computacional dos contratos legais, a *Accord Project* desenvolveu 3 sistemas que em conjunto criam um denominado *Smart Template*, o qual contém todos os dados e lógica de operação de um *Smart Agreement,* visualizados na *figura 1*. Com este capítulo cumpre-se o objetivo de investigação de tecnologias atuais para estruturar documentos legais em formatos digitais executáveis.

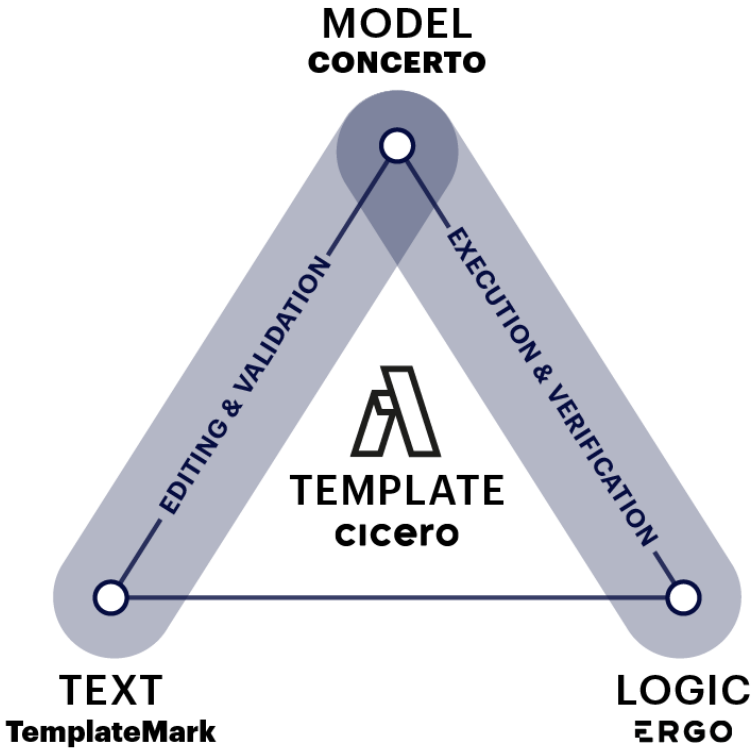


Figura 1 – Componentes de um SLC no Accord Project

2.2.1 Cicero

O primeiro elemento e o mais básico de um *Smart Legal Contract*, é o *Cicero*, um formato de texto e *markup* que indica à máquina a estrutura de um *template*. Ficheiros no formato determinado pelo *Cicero,* que se identificam como *Grammar*, contêm então o texto do documento e uma representação em markup das variáveis que compõem o template, e podem até incluir de modo limitado um pouco de lógica executável, exemplificado na *figura 2*.

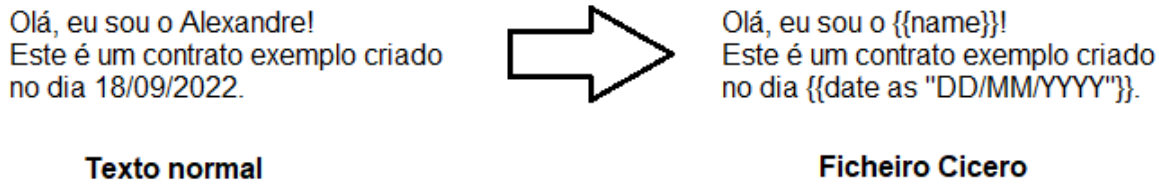


Figura 2 - Exemplo da transformação de texto corrido normal para formato Cicero.

2.2.2 Concerto

Em segundo lugar, o sistema *Concerto* serve como uma representação programática das variáveis de um contrato. Estes ficheiros, denominados *Model*, contêm conteúdo legível pelo computador, que é usado para a definição de conceitos, transações, recursos e até participantes utilizando uma linguagem de programação, com um estilo semelhante a *Java* e *C#*, sendo adicionalmente facilmente extensível. Este componente também realiza a validação e verificação da estrutura do contrato e variáveis definidas no documento de markup em *Cicero*, e serve como a base para a definição de tipos e estrutura de classes usadas pelo sistema lógico, *Ergo*. A *figura 3* exemplifica a estrutura de um *Model.*

```
namespace org.accordproject.examplecontract

import org.accordproject.cicero.contract.* from
https://models.accordproject.org/cicero/contract.cto

import org.accordproject.cicero.contract.* from
https://models.accordproject.org/cicero/contract.cto

/**
* Data Model for the ExampleContract template
*/

asset ExampleContract extends Contract {
o String name // Name of the person
o DateTime date // Date of creation
}
```

Figura 3 - Exemplo de um Model no formato Concerto.

2.2.3 Ergo

Ergo, é uma linguagem de programação específica aos *Smart Templates* do *Accord Project* que captura a execução de contratos legais digitais. Os contratos criados nesta linguagem têm uma estrutura de classes, que utiliza cláusulas como métodos e funções. Os ficheiros *Ergo*, também conhecidos como *Logic*, verificam os conteúdos dos *Models* e as definições das variáveis nestes para a execução funcional das cláusulas. A linguagem é determinista e limitada e tem assinaturas de métodos bem definidas, sendo que cada cláusula deve conter uma definição de um *input* e um *output*, definidos como *transactions* no *Model*. Os ficheiros *Logic,* exemplificados na *figura 4* e *5,* são compilados e executados em conjunto com os ficheiros de *Grammar* e *Model*, completando então os requerimentos para um *Smart Template* funcional e executável.

```
// Defines the input data for the "Example" clause
transaction exampleRequest extends Request {
o String input
}


// Defines the output data for the "Example" clause
transaction exampleResponse extends Response {
o String output
}
```

Figura 4 - Definição das transações da cláusula no ficheiro Model.

```
namespace org.accordproject.examplecontract

contract ContractExample over ExampleContract {

  //Return the name variable
  clause Example(request : exampleRequest) : exampleResponse {
        return exampleResponse {
           output: "The name is " + contract.name
    }
  }

}
```

Figura 5 - Exemplo da definição de uma cláusula exemplo num ficheiro Logic em Ergo.

3. TECNOLOGIAS DE DOCUMENT PROCESSING

Para um melhor entendimento das tecnologias de *Document Processing*, realizou-se um estudo que analisa as maiores plataformas que utilizam estas técnicas. Como tal, as plataformas selecionadas foram a *Amazon Comprehend* e a *Microsoft Azure AI.* Os inícios da extração de dados passam por paradigmas de aprendizagem de máquinas, tais como o *Graph Transformer Networks,* o qual estendeu a aplicabilidade de algoritmos de aprendizagem baseados em gradientes a sistemas que têm principalmente grafos como entrada e saída. Aplicando este método ao processamento de documentos, foi possível representar o conhecimento e o estado da informação que compõem um documento através dos grafos, extraindo essencialmente os dados de um documento (Bottou et al., 1997). Em 2003, foi criado o sistema de extração de informação Artequakt que procurava não só extrair dados de entidades de documentos *web*, mas também registar e criar ligações entre os conhecimentos recolhidos. Isto permitiu ao sistema pintar um quadro mais completo e compreender a informação recolhida através da análise das relações entre entidades de dados (Alani et al., 2003). As tecnologias modernas de extração de dados utilizam conjuntos de dados de documentos etiquetados e não etiquetados para treinar os algoritmos de aprendizagem de máquinas. Durante o seu processo de desenvolvimento nota-se que, ao comparar diferentes métodos de extração, tais como a utilização de regras escritas manualmente e classificadores lineares, os melhores resultados provinham de um método híbrido (Chalkidis et al., 2017). Mais recentemente, utilizando um conjunto de dados de documentos físicos que contém estruturas tabulares complexas, foi investigada a capacidade de extração de informação que existe nestes documentos através da utilização de redes neuronais convolucionais e segmentação condicional de campos aleatórios (Zhang & Dell, 2019). Estas tecnologias comerciais utilizam vários destes métodos para atingir os seus resultados de extração de dados. Neste capítulo serão também exploradas bibliotecas *open source* de *Natural Language Processing, Document Understanding* e de outros tipos de compreensão de texto.

3.1 Amazon Comprehend

A *Amazon Comprehend* é uma plataforma que utiliza modelos treinados previamente para analisar e examinar documentos, extraindo dados e informação destes. Visto que este modelo está a ser continuamente treinado pela própria Amazon, não há necessidade de apresentar ao algoritmo um *dataset* de treino. A plataforma é capaz de reconhecer múltiplas línguas e consegue também indicar a língua dominante de cada documento.

3.1.1 Amazon Console

A *Amazon Console* é a plataforma pela qual se podem efetuar várias operações nos documentos que lhe são apresentados. Existe uma variedade de funcionalidades desde deteção de língua a modelação por tópicos. De seguida estão descritas as funcionalidades mais relevantes a este projeto:

- Detect Entities: Deteção de entidades através da análise do texto tais como, nomes de pessoas, localizações, organizações, objetos e consegue até analisar datas e valores numéricos. A *figura 6* exemplifica esta funcionalidade.

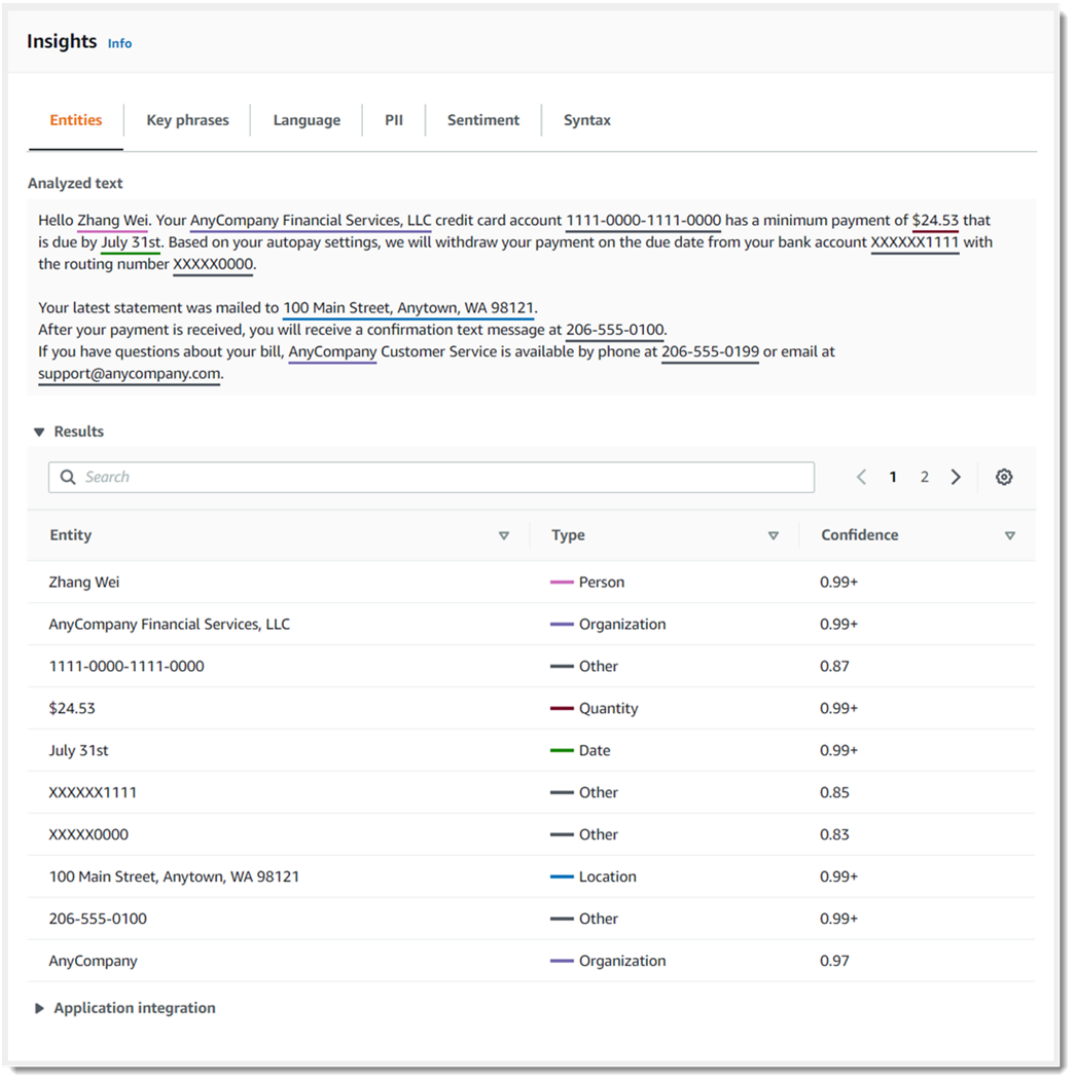


Figura 6 - Exemplo da funcionalidade "Detect Entities" da Amazon Comprehend.

- **Detect Key phrases:** O algoritmo consegue detetar e extrair frases/palavras chave que se encontram presentes no documento, como demonstrado pela *figura 7*.

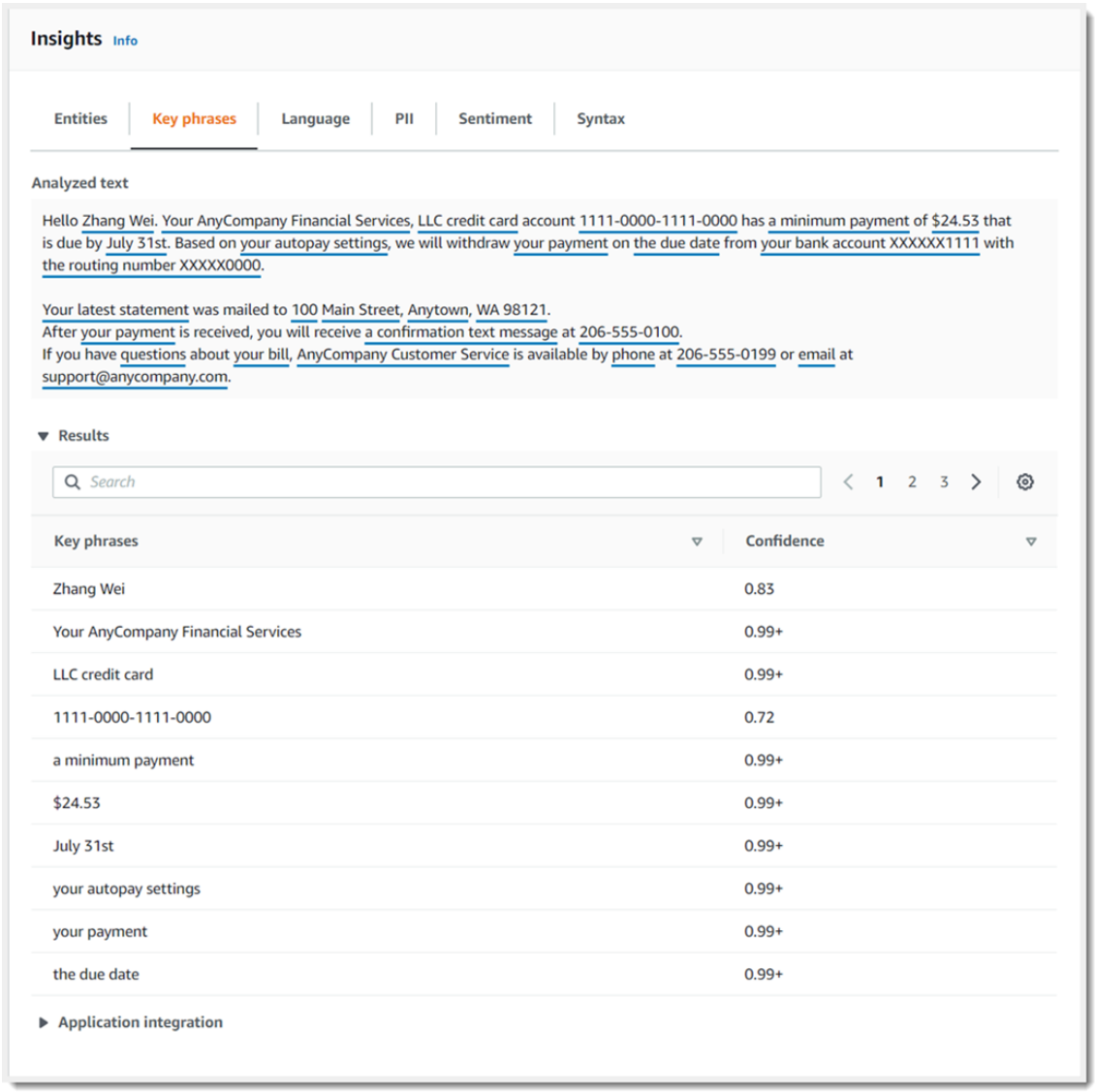


Figura 7 - Exemplo da funcionalidade "Detect Key Phrases" da Amazon Comprehend.

No âmbito do projeto, ambas estas funcionalidades podem ser utilizadas em conjunto para detetar as entidades presentes num contrato e para extrair frases-chave que podem ser úteis para identificar e extrair as cláusulas que formam o contrato.

A Amazon também disponibiliza a utilização de todas as suas funcionalidades através de uma API *web*.

3.1.2 Custom Entity Recognizers

Para além das funcionalidades básicas, a Amazon Comprehend disponibiliza também a opção de criar classificadores de entidades customizados. Estes classificadores são capazes de extrair e classificar informação com base no seu contexto dentro do documento. Para tal, é

necessário apresentar um *dataset* de documentos com as classificações atribuídas manualmente para o algoritmo ser treinado devidamente. Existem duas métodos para classificar as entidades nos documentos:

- Annotations: Os documentos são submetidos com anotações que definem o contexto das entidades requeridas sem ambiguidade. É o método mais trabalhoso e o que consome mais tempo.

- Entity Lists: É submetida uma lista de entidades e a Amazon Comprehend utiliza um algoritmo inteligente que tenta associar às entidades uma classificação. É um método mais simples e rápido, mas está sujeito a erro quando existem entidades diferentes que são identificadas pelos mesmos nomes.

3.2 Microsoft Azure AI

A plataforma *Azure AI* da *Microsoft* disponibiliza funcionalidades relacionadas a automatização do processamento de documentos. Nesta secção estão detalhadas as funcionalidades principais. Todas as ferramentas e funcionalidades apresentadas podem ser usadas através de chamadas às APIs destas ou através de plataformas desenvolvidas pela *Microsoft*.

3.2.1 Optical Character Recognition (OCR)

Esta funcionalidade tem como objetivo utilizar tecnologias de *computer vision* para reconhecer e extrair texto vindo de documentos escritos manualmente, imprimidos ou até das suas imagens. Apoia de base as linguagens C#, Java, Javascript e Python. Texto imprimido poderá ser extraído em várias linguagens, enquanto que texto manuscrito funciona apenas com o inglês. Consegue também reconhecer dígitos, símbolos de moedas e mais. Está otimizada para documentos pesados em imagens e para PDFs com múltiplas páginas. O input pode ser uma grande variedade de formatos de ficheiro e o output é em formato JSON, como retratado na *figura 8*.

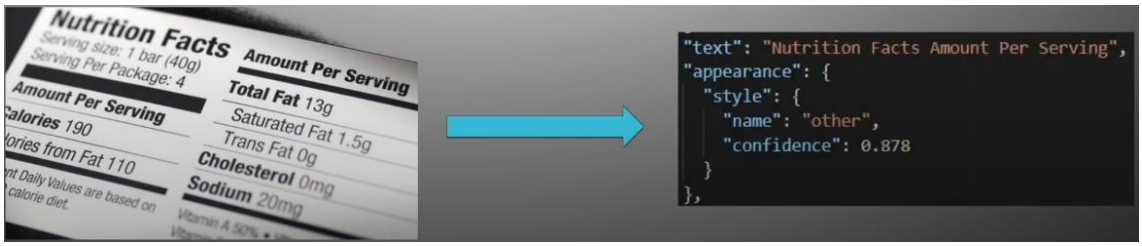


Figura 8 - Exemplo de extração de texto de uma imagem para um objeto JSON.

3.2.2 Form Recognizer

O *form recognizer* consegue rapidamente extrair de documentos texto, pares de chave-valor e tabelas com grande precisão, transformando formulários em dados fácilmente utilizáveis. Diferenciando-se do OCR, o *form recognizer* compreende a estrutura dos documentos e os seus campos e valores. Esta funcionalidade está dividida em 3 partes:

- Layout API: Dado um formulário, são reconhecidas cada célula das tabelas e também valores numéricos, símbolos de moedas e carateres.

- Prebuilt Models: De momento existem apenas 2 *prebuilt models*. Recibos de vendas e cartões de negócios. Cada modelo está otimizado para extrair os dados dos seus respetivos layouts.

- Custom Models: É possível treinar *custom models* com *datasets* à escolha do utilizador para otimizar o modelo na extração de dados de formulários específicos ao negócio em questão.

Existe também uma ferramenta que permite ao utilizador marcar manualmente os campos que devem ser captados pelo *form recognizer* no caso de que a utilização não supervisionada da funcionalidade falhe a reconhecer todos os campos e valores que o utilizador quer reconhecer e processar.

3.2.3 Cognitive Search

Esta funcionalidade disponibiliza uma procura fácil e eficiente dos dados importantes que foram previamente retirados e processados dos diversos documentos em questão. Para tal, é necessário seguir 3 passos para indexar os dados não estruturados que obtivemos. O primeiro passo, denominado "*Ingest*", é a simples inserção de dados na plataforma. O segundo, chamado "*Enrich*", é o processo de melhoramento dos dados através da utilização de skills

cognitivos, tais como classificação de imagens ou deteção de línguas, que permite produzir e inserir novos campos nos dados. O algoritmo é também capaz de detetar e extrair contextualmente nomes de pessoas, de organizações, de localizações e até frases chave. O último passo "*Explore*", consiste na exploração facilitada destes dados através de filtros e palavras-chaves associados a estes.

3.3 Ferramentas Open Source

Foram encontradas várias bibliotecas e módulos para *Python*, *Javascript*, e *C* que utilizam tecnologias de *Natural Language Processing* para extrair, analisar e classificar informação de documentos em texto simples. Neste caso, foram testadas apenas as seguintes bibliotecas de Node.js: Compromise.js e Natural.js, visto que as bibliotecas de outras linguagens são equivalentes e paralelas em termos funcionais.

3.3.1 Compromise.js

O principal objetivo desta biblioteca é a extração direta de informação de documentos, isto deve-se à sua capacidade de *Entity Recognition* que coloca etiquetas nas diversas entidades que encontra no documento, permitindo diferenciar entre Pessoas, Organizações, Locais e até entre Determinantes, Pronomes, Preposições, etc.

Um ponto fraco desta biblioteca é a sua limitação de linguagem, sendo que apenas funciona em inglês, mas, por outro lado, oferece uma quantidade enorme de costumização sendo possível criar os nossos próprios léxicos/dicionários em *Json* e até etiquetas costumizadas.

Foi realizado um teste inicial para determinar que resultados seriam obtidos através da utilização do algoritmo sem quaisquer alterações. O *input* é um template de um contrato de prestação de serviços fornecido pela *DigitalSign*. Os *outputs* deste teste foram determinados como sendo Pessoas (*People*), Organizações (*Organizations*) e Locais (*Places*).

Figura 9 - Output "People" sem alterações ao léxico da biblioteca Compromise.js

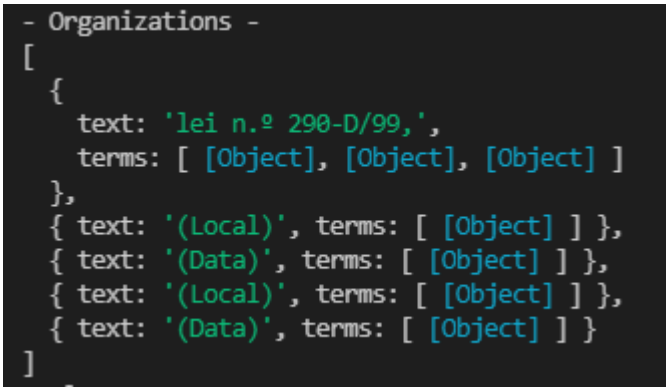


Figura 10 - Output "Organizations" sem alterações ao léxico da biblioteca Compromise.js

Figura 11 - Output "Places" sem alterações ao léxico da biblioteca Compromise.js

Observa-se, nos resultados dos testes iniciais demonstrados nas *figuras 9, 10 e 11,* que de facto o algoritmo da biblioteca não compreende a língua Portuguesa, o que cria uma mescla de palavras nos *outputs* que teoricamente não pertencem às categorias que desejamos. Outro dos problemas que observamos é que o algoritmo confunde nomes de ruas com nomes de pessoas (isto deve-se à natureza dos nomes de ruas em Portugal), o que poderá ser bastante complicado de resolver.

Para combater este problema foram encontrados datasets no portal europeu de dados abertos no formato CSV. A seguinte *tabela 1* descreve estes dados e de que modo foram etiquetados.

Tabela 1 – Datasets extraidos do portal Europeu de dados.

Dataset	**Descrição**	**Etiqueta Atribuida**
nomesmasculino.up.csv	Nomes Masculinos registados em Portugal no ano de 2017	Person
nomesfeminino.up.csv	Nomes Femininos registados em Portugal no ano de 2017	Person
mnindustria.csv	Lista de Estabelecimentos Industriais em Portugal no ano de 2017	Organization
rua.csv	Lista de Ruas em Portugal no ano de 2017 (não especifica se são todas as ruas existentes)	Place

De todos estes datasets apenas o *mnindustria.csv* necessitou de transformações adicionais devido a um erro de formatação presente no documento. Foi, por isso, simples e rápido de resolver.

As informações relevantes nestes CSVs foram então usadas para criar um novo léxico em *Json* para ajudar o algoritmo a compreender a língua. Adicionalmente, foram registadas também no léxico múltiplos determinantes, pronomes, preposições e algumas palavras-chave em Português. Este léxico é posteriormente extraído para um ficheiro *Json* que será usado no algoritmo para correr os testes.

Figura 12 - Output "People" com alterações ao léxico da biblioteca Compromise.js

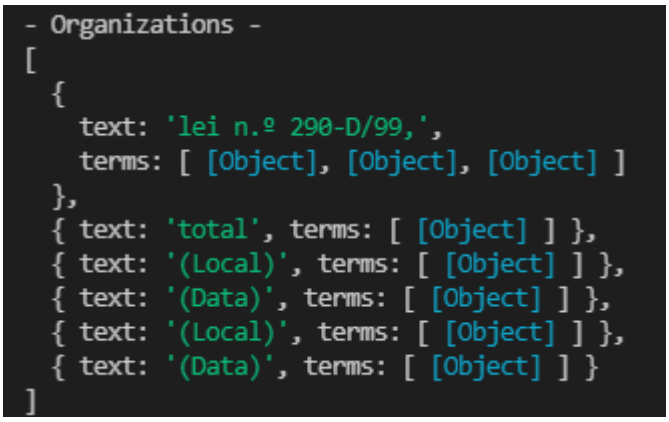


Figura 13 - Output "Organizations" com alterações ao léxico da biblioteca Compromise.js

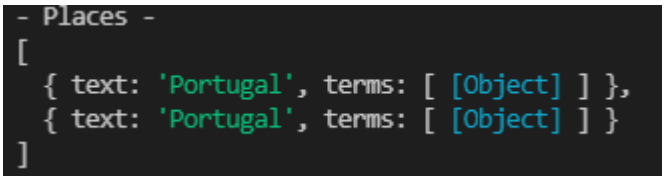


Figura 14 - Output "Places" com alterações ao léxico da biblioteca Compromise.js

Observa-se então, nas *figuras 12, 13 e 14,* uma grande diferença, especialmente nos *outputs* "*Organizations*" e "*Places*" quando utilizamos o léxico em Português. No caso *output* "*People*" ainda se mantém o problema das ruas, mas, muitas das palavras que antes apareciam e não eram pertencentes a este grupo já não são incluidas, e, por outro lado, o algoritmo reconhece "contraente" e "contratante" como "*People*" com sucesso, que foram palavras-chave adicionadas ao léxico. As organizações não estão a ser devidamente reconhecidas, isto deve-se a que o *dataset* utilizado para obter as organizações pertence a um setor específico o que limita a lista de organizações existentes atualmente.

3.3.2 Natural.js

Após uma análise da documentação da biblioteca *Natural.js,* conclui-se que este módulo não contém as funcionalidades necessárias para obter o objetivo do projeto, visto que esta se foca mais numa compreensão linguística e gramatical do documento. Esta biblioteca funciona maioritariamente em Inglês, e as funcionalidades que poderiam ser úteis como *Logistic Regression Classifiers* e *Bayesian Classifiers* não têm suporte para Português. Adicionalmente, apesar de também apresentar customização dos seus classificadores, estes têm de ser alterados e treinados de um modo semelhante, ou mais árduo, ao que foi realizado nos testes com *Compromise.js*.

3.4 Algoritmo Diff

O algoritmo *diff*, é capaz de detetar e relatar diferenças entre dois ficheiros. É um algoritmo eficiente que analisa os conteúdos de documentos linha a linha, ou palavra a palavra, com o objetivo inicial de documentar as diferenças entre versões diferentes de um documento, indicando que linhas de um ficheiro devem ser alteradas para tornar as duas versões idênticas (Hunt & Mcilroy, 1976). Ainda hoje este algoritmo é usado em programas e plataformas como o *github* e *bitbucket,* para controlo de versões. Com as evoluções mais recentes, a operação é utilizada também em conjunto com outros algoritmos, tais como o *Kolmogorov Complexity,* para medir não só as diferenças entre documentos, mas também a semelhança entre eles. Isto permite utilizar estes emparelhamentos de processos computarizados para detetar plágio em contextos universitários, fraudes, ou para assistir na escrita de simples textos, tal como muitos programas de e-mail fazem na atualidade (Del & Jr, 2019).

No contexto de *Smart Legal Contracts* este algoritmo pode ser usado, em conjunto com outros métodos, para extrair e classificar dados com o fim de gerar um *Smart Template*. Para tal, devem ser utilizados dois contratos exemplo com uma estrutura de documento igual, em que o único texto que varia é o que será extraído para processamento.

4. PROCESSO DE DIGITALIZAÇÃO

Um dos tópicos mais desafiantes encontrados nas tecnologias-chave relacionadas com a engenharia de dados e aquisição de conhecimentos é a aquisição automática de dados. A aquisição manual de conhecimentos de documentos tais como relatórios técnicos, ficheiros governamentais, jornais e muito mais, envolvia uma grande quantidade de trabalhos manuais por parte do engenheiro, o que era desgastante e limitativo. Foi desenvolvida uma técnica de processamento de documentos por computador que envolve a análise e compreensão de documentos através do uso de inteligência artificial. Para gerar conhecimento a partir de um documento, é muito importante detectar a sua estrutura. Assim, foram identificados dois tipos de estruturas nesta técnica, a estrutura geométrica que é extraída pela análise do documento e a estrutura lógica, que utiliza o processo de compreensão de documentos para mapear a estrutura geométrica sobre a estrutura lógica. Acabou por se concluir que, é difícil encontrar um mapeamento correcto para transformar uma estrutura geométrica numa estrutura lógica, porque não existe um mapeamento um-a-um entre estes dois espaços. Uma vez que as regras baseadas no conhecimento podem variar de documento para documento, este assunto foi deixado para futuras pesquisas (Tang et al., 1994). Esta diferença entre documentos continuou a causar problemas, especialmente no departamento legal, em que os contratos eletrónicos podem ser compostos por um grande número de documentos contratuais colaterais, cada um destes documentos pode ter a sua própria estrutura e formatos originais, tornando a extracção automatizada de dados muito difícil. (Kwok & Nguyen, 2006). Neste capítulo idealiza-se, através de uma perspetiva de resolução de um problema empresarial, a criação de processos de digitalização de contratos legais, com o objetivo de criar *Smart Templates,* como definidos pelo *Accord Project,* a partir destes, tendo em conta as carateristicas diferenças entre documentos. Para tal foram definidas as seguintes clarificações e restrições:

- No final do processo de digitalização, é obtido um *Smart Template* composto por uma estrutura e lógica executável específica ao contrato digitalizado.

- A estrutura de cada *Smart Template* é rígida. Para instânciar um contrato com base num *template*, este deve seguir a mesma estrutura.

- Caso não exista um *template* adequado a um contrato, este deve passar pelo processo de digitalização para ser devidamente executado.

Devido a uma falha de organização e planeamento de horários, passou-se um período de tempo alargado a explorar e a fazer *brainstorm* de várias facetas do projeto que acabaram por não ser utilizadas. Documentação e artigos sobre *Machine Learning* e *Neural Networks* foram explorados, e também modos diferentes de estruturar os dados dos contratos e dos *templates*. No entanto, a investigação acabou por sair do âmbito do projeto, pelo que foi necessário reencontrar o foco e realizar um planeamento mais otimizado do que restava do projeto. Com as informações novas recolhidas sobre estas tecnologias, realizou-se então uma revisão destas.

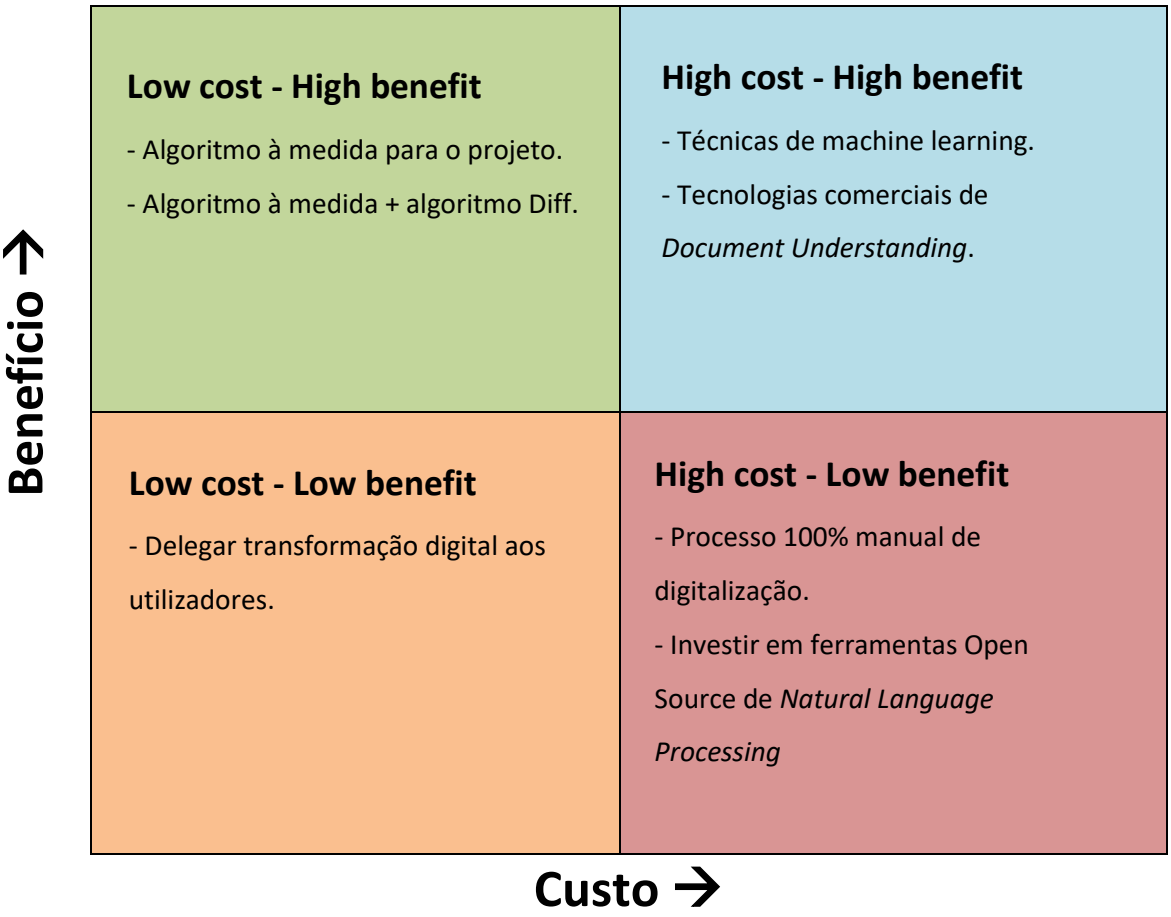
4.1 Escolha das Tecnologias

Considerando a perspetiva empresarial, para avaliar e escolher quais tecnologias, ou métodos de desenvolvimento, a usar no projeto, foi feita uma análise custo-benefício. Este tipo de análise é um dos métodos de avaliação económica mais compreensivos, apesar de limitada, sendo uma boa forma de entender o que um indivíduo ou empresa está "disposto a pagar" por determinados beneficios (Koopmans & Mouter, 2020). Sendo o objetivo manter o custo o mais baixo possível, enquanto se atinge em simultâneo o objetivo do projeto, as tecnologias e métodos a avaliar são os seguintes:

- Desenvolver algoritmo à medida
- Utilizar algoritmo *Diff*
- Utilizar técnicas de machine learning
- Recorrer a tecnologias comerciais de *Document Understanding*
- Delegar transformação digital aos utilizadores
- Recorrer a um processo 100% manual de digitalização
- Investir em ferramentas *Open Source* de *Natural Language Processing*

A *tabela 2*, classifica esta listagem de acordo com a análise custo-benefício efectuada.

Tabela 2 - Análise de custo/benefício da utilização de diferentes abordagens tecnológicas.



4.1.1 Low cost – Low Benefit

Começando a análise pelo nível mais baixo, a proposta com menos custo e menor benifício será então a delegação da transformação digital aos utilizadores de uma plataforma. Com este método, cada utilizador deve gerar, da forma que lhe é mais conveniente, o *Smart Template* de cada contrato do qual quiser fazer *upload* para uma plataforma que os execute. O custo para a empresa é baixo, pois não terá o esforço de criar estes *templates* mas também terá pouco benefício, pois o problema da digitalização dos contratos legais não é resolvido, o que iria dissuadir possíveis clientes.

4.1.2 High cost – Low Benefit

Este é o pior caso em termos de custo-benifício, pois indica que o retorno de uma grande quantidade de esforço seria muito baixo. Aqui encaixa o processo de transformação digital 100% automatizado, o que requereria uma quantidade enorme de mão de obra, aumentando por margens inaceitáveis a despesa da organização. Também entra aqui o investimento em ferramentas *Open Source* de *Natural Language Processing*, pois como foi previamente investigado, seria necessário demasiado esforço para tornar a tecnologia compatível com português, criar regras customizadas para deteção de entidades, construir ou transformar datasets para os propósitos do projeto, e realizar uma abundância de testes para certificar que o sistema seja viável.

4.1.3 High cost – High Benefit

As tecnologias neste quadrante já foram previamente provadas e testadas por outros sistemas. No entanto, relativamente a outras soluções, são demasiadas dispendiosas para o âmbito do projeto. Apesar do seu custo, os seus benefícios são óbvios especialmente na extração de cláusulas legais de um documento contratual, pois comparando a detecção convencional de parágrafos com abordagens baseadas na aprendizagem de máquinas, conclui-se que os algoritmos baseados em regras e de detecção de frases não são tão adequados para conjuntos de dados em grande escala como os algoritmos de classificação de aprendizagem de máquinas (Shah et al., 2018). Analisando as tecnologias como *Machine Learning*, Redes Neurais Convolucionais, Transformadores e BERT, descobriu-se que existia uma falta de melhoria do desempenho de características como a incorporação de forma simbólica e etiquetas POS, que contradizem as conclusões de trabalhos anteriores no campo da extracção de elementos contratuais. Isto indica que a extracção de elementos requer escolhas específicas da tarefa para uma extracção precisa dos dados contratuais (Chalkidis et al., 2019). Pelo que, apesar de serem tecnologias avançadas e que trazem grandes benifícios, para criar um sistema destes seria necessário uma equipa de desenvolvimento e uma

quantidade de tempo adicional para análises, criação e manipulação de *datasets* e testes de viabilidade, que cai fora do escopo deste projeto. No caso das tecnologias comerciais, para além dos custos de integração, existe também uma despensa monetária para o uso destas plataformas que não é o ideal para a empresa acolhedora.

4.1.4 Low cost – High Benefit

A solução ideal para o projeto foi então determinada como sendo o desenvolvimento de algoritmos customizados à solução, tal como a grande parte de soluções empresariais dos dias de hoje, recorrendo potencialmente a algoritmos como *Diff* ou semelhantes para apoio às diferentes fases do projeto. O custo permanece baixo pois não requer mão-de-obra adicional, nem há um custo monetário associado a nenhuma das tecnologias utilizadas, tendo como outros pontos positivos o controlo alargado sobre os dados e a facilidade de integração com a tecnologia desenvolvida pelo *Accord Project*.

4.2 Extração de Dados e Criação de Cláusulas

Com a escolha das tecnologias realizadas, pode-se agora pensar num subprocesso de extração de dados, tendo em conta o desafio das diferenças de conteúdo e estruturais entre documentos. A extração de dados idealizada foi dividida em dois sub-processos:

- **Processo de extração Manual**: O utilizador cria variáveis juntamente com o seu tipo e seleciona o texto do contrato, associando-o à variável.
- **Processo de extração Semimanual**: O utilizador faz upload de dois contratos com a mesma estrutura e, através do algoritmo *Diff,* são detetadas as diferenças de forma automática, criando as variáveis juntamente com o seu tipo. Num passo seguinte, estas variáveis devem ser validadas pelo utilizador.

Para a criação das cláusulas existe apenas um sub-processo, em que o utilizador seleciona que cláusulas quer incluir no seu contrato a partir de uma lista de cláusulas executáveis. Algumas

destas cláusulas podem requerer uma referência a uma variável com um tipo específico. Por exemplo, a cláusula “data limite” requer que lhe seja associada uma variável do tipo “data”, que deve existir no contrato.

4.3 Processo de criação de um Smart Template Manual

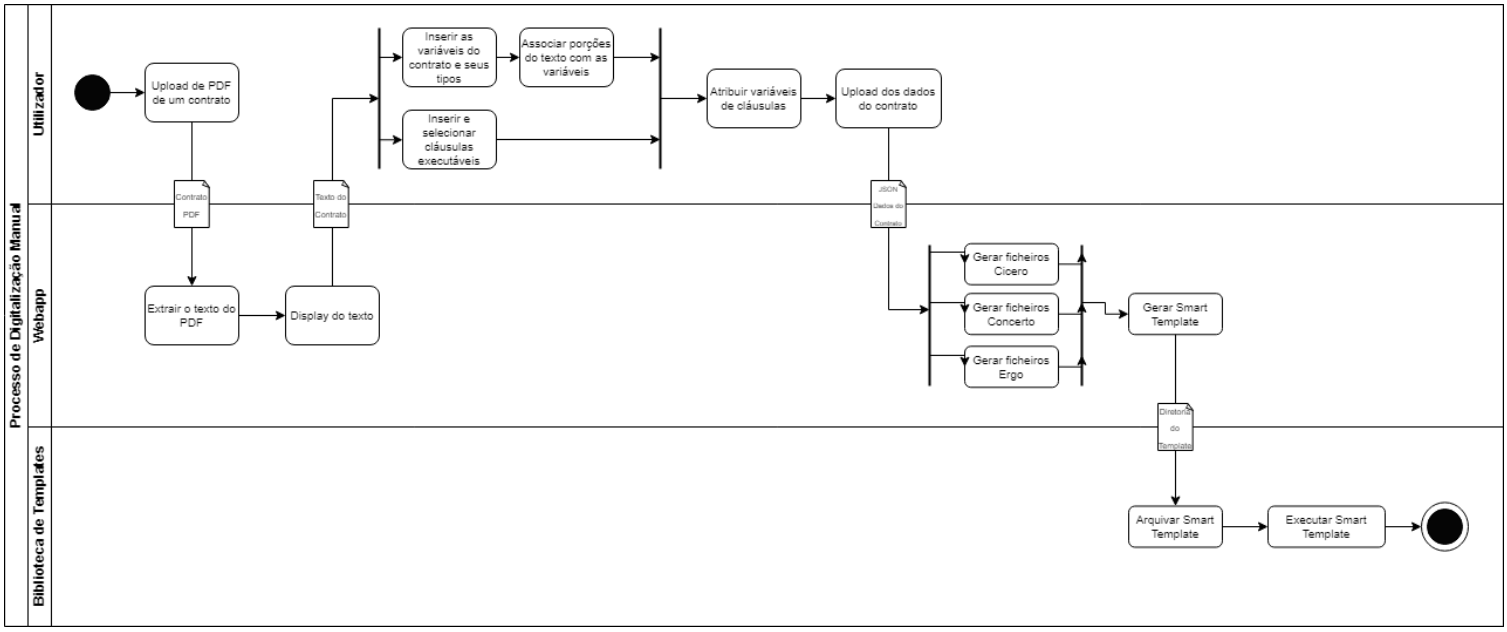


Figura 15 - Processo de Digitalização Manual.

O processo de digitalização de contratos final foi simplificado, tal como demonstra a *Figura 15*. O utilizador inicia o processo ao efetuar um *upload* de um contrato exemplo, no formato PDF, numa aplicação web, através de uma interface visual. De seguida, o texto, e apenas o texto, do contrato é extraído do ficheiro e a aplicação envia este para a visualização do utilizador.

Após o upload, na interface gráfica, o utilizador insere as variáveis do contrato, determinando o seu nome e tipo (String, Numérico, Data, Monetário, ...) e atribui porções do texto do contrato, selecionando-as, às variáveis. O utilizador insere também as cláusulas executáveis, incluindo o seu nome e o tipo de cláusula, de uma lista em que cada cláusula terá funcionalidades diferentes. Após estes dois passos, o utilizador atribui determinadas variáveis às cláusulas e faz upload dos dados inseridos para a aplicação.

De seguida, a aplicação web, gera automáticamente o código, nas linguagens *Cicero, Concerto e Ergo* definidas pelo *Accord Project*, e a estrutura das diretorias que compõem o template de um *Smart Legal Contract*.

No final deste processo, é gerado um *Smart Template* do contrato e arquivado numa biblioteca digital onde este é instanciado e executado pelo motor do *Accord Project*.

4.4 Processo de criação de um Smart Template Semimanual

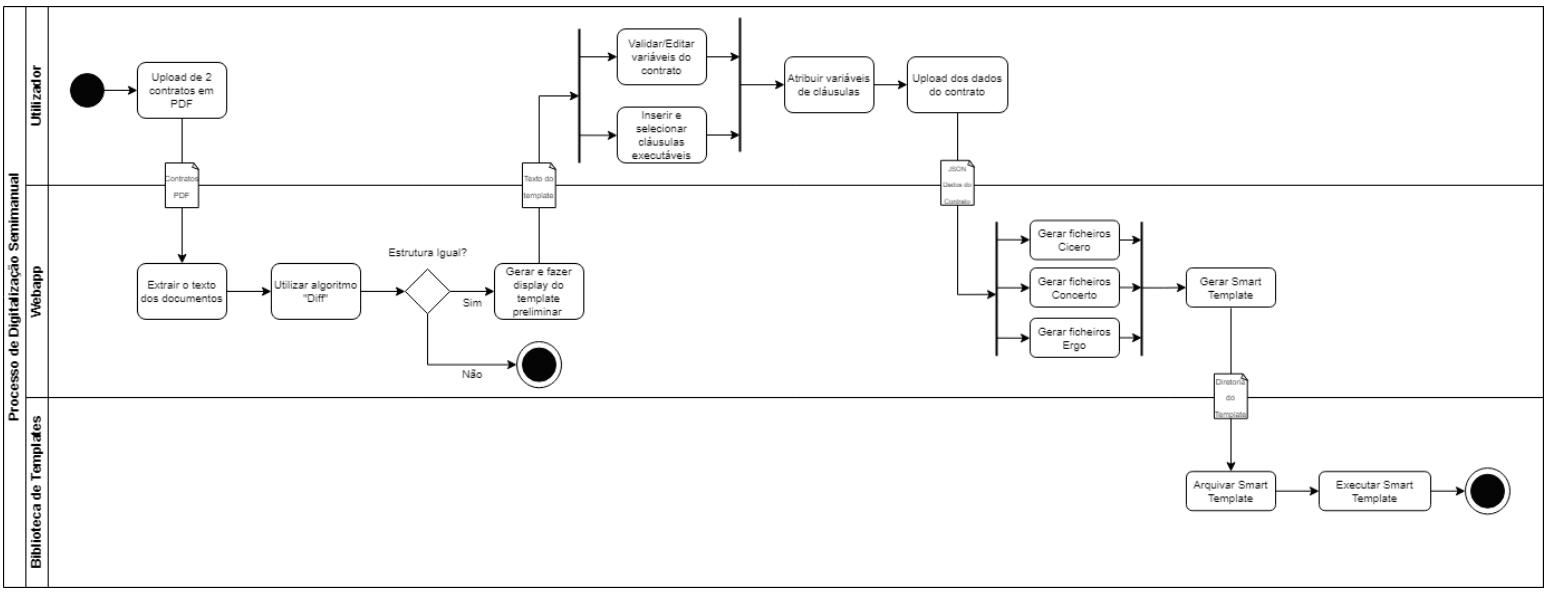


Figura 16 - Processo de Digitalização Semimanual.

Este processo de digitalização de contratos foi baseado no algoritmo *Diff*, tal como demonstra a *Figura 16*. O utilizador inicia o processo ao efetuar o *upload* de dois contratos em PDF com a mesma estrutura, mas com diferenças nas porções do texto onde se desejam as variáveis. De seguida, o texto de ambos os contratos são extraídos e processados pelo algoritmo *Diff*, que após de detetar as diferenças entre ambos os contratos, cria um *template* preliminar, contendo uma suposição das variáveis que existem e quais os seus tipos, e este é enviado ao utilizador para validação manual.

O utilizador analisa o *template* preliminar do contrato, alterando as variáveis que foram detetadas através da edição dos seus nomes, tipos e até associações às porções do texto do contrato. Após a validação do *template,* o utilizador insere também as cláusulas executáveis,

incluindo o seu nome e que tipo de cláusula, de uma lista em que cada cláusula terá funcionalidades diferentes. Após estes dois passos, o utilizador atribui determinadas variáveis às cláusulas e faz upload dos dados inseridos para a aplicação.

De seguida, a aplicação web, gera automáticamente código, nas linguagens *Cicero, Concerto e Ergo* definidas pelo *Accord Project*, e a estrutura das diretorias que compõem o template de um *Smart Legal Contract*.

No final deste processo, é gerado um *Smart Template* do contrato e arquivado numa biblioteca digital onde este é instânciado e executado pelo motor do *Accord Project*.

4.5 Processo de Instanciação de Contractos

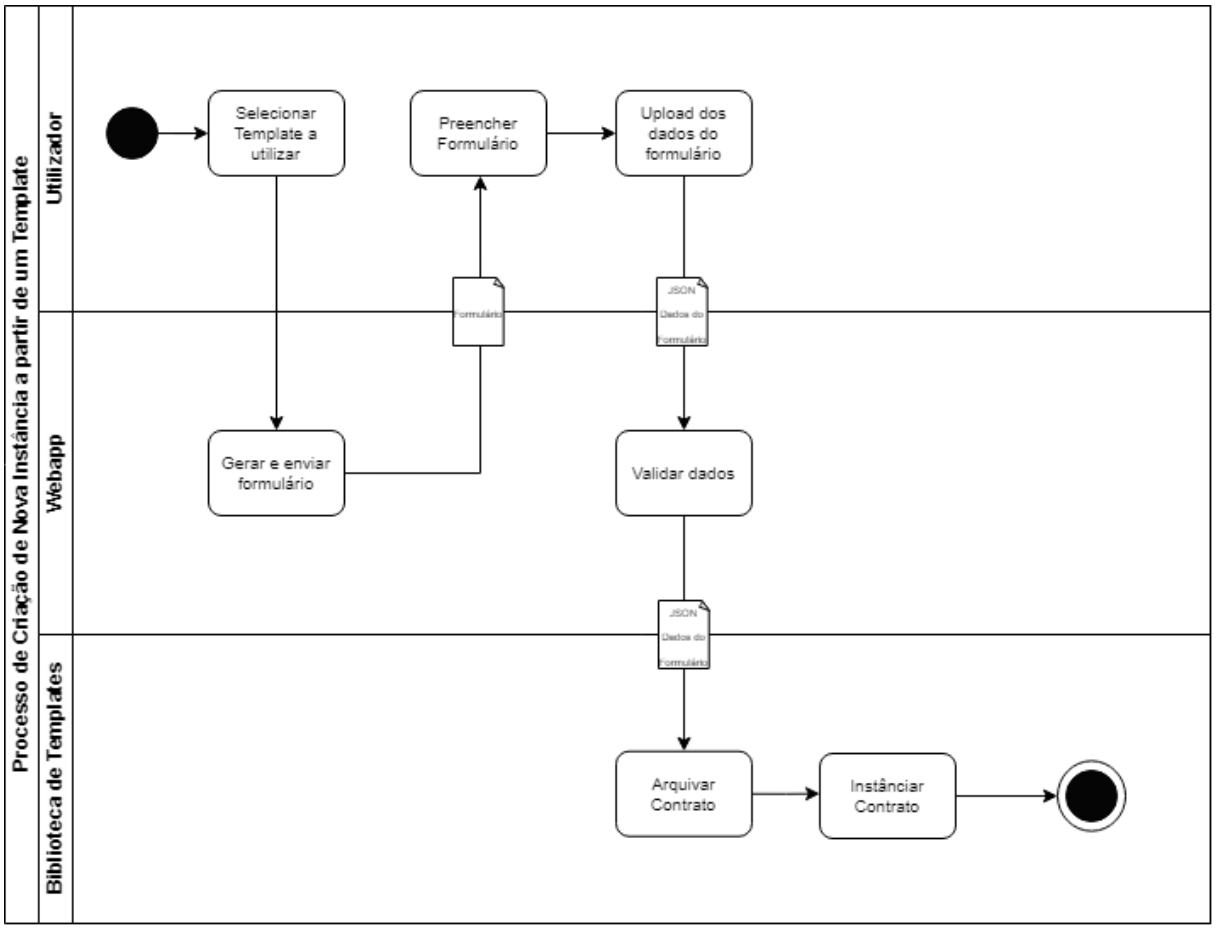


Figura 17 - Processo de Instanciação de Contratos.

A *figura 17*, detalha o processo em que o utilizador, após gerar o *Smart Template* de um contrato, pode criar instâncias do mesmo. Estas instâncias contêm os dados e valores das variáveis de cada contrato individual, que serão processados pelo *Smart Template*. Como a lógica e modelação destes contratos já está presente no *template*, é apenas necessário arquivar a informação num simples ficheiro *JSON.*

O utilizador inicia o processo de instanciação de contratos ao selecionar o *template* que deseja utilizar. Após a escolha é-lhe enviado um formulário composto pelo texto do contrato, com campos de *input* (o tipo do *input* dependente do tipo da variável) na posição onde se inserem as variáveis como definidas no *Smart Template*. De seguida, o formulário é enviado a um servidor que valida os dados e arquiva o contrato, criando também uma instância deste para ser processado pelo *template*.

5. PROVA DE CONCEITO

Foi realizada uma prova de conceito com base no processo de criação de um *Smart Template* manual, detalhado no capítulo 5.3. Para tal, foi desenvolvida uma simples *Web App,* composta por um *front-end* e um *back-end,* como demonstra a *figura 18.* O *front-end* inclui uma página web programada com *HTML, CSS* e *Javascript,* enquanto que o *back-end* se baseia em *Node.js,* recorrendo às bibliotecas do *Accord Project*.

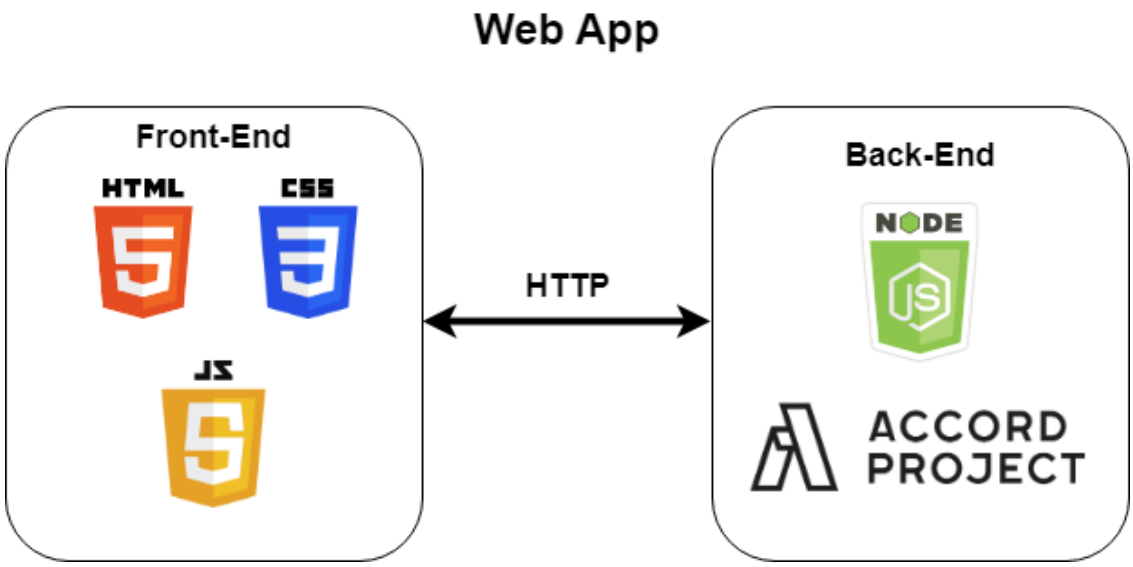


Figura 18 - Modelo de Arquitetura da prova de conceito.

5.1 Upload do Contrato

Como definido anteriormente, o utilizador inicia o processo com o *upload* de um contrato através da página *web* exemplificada na *figura 19.*

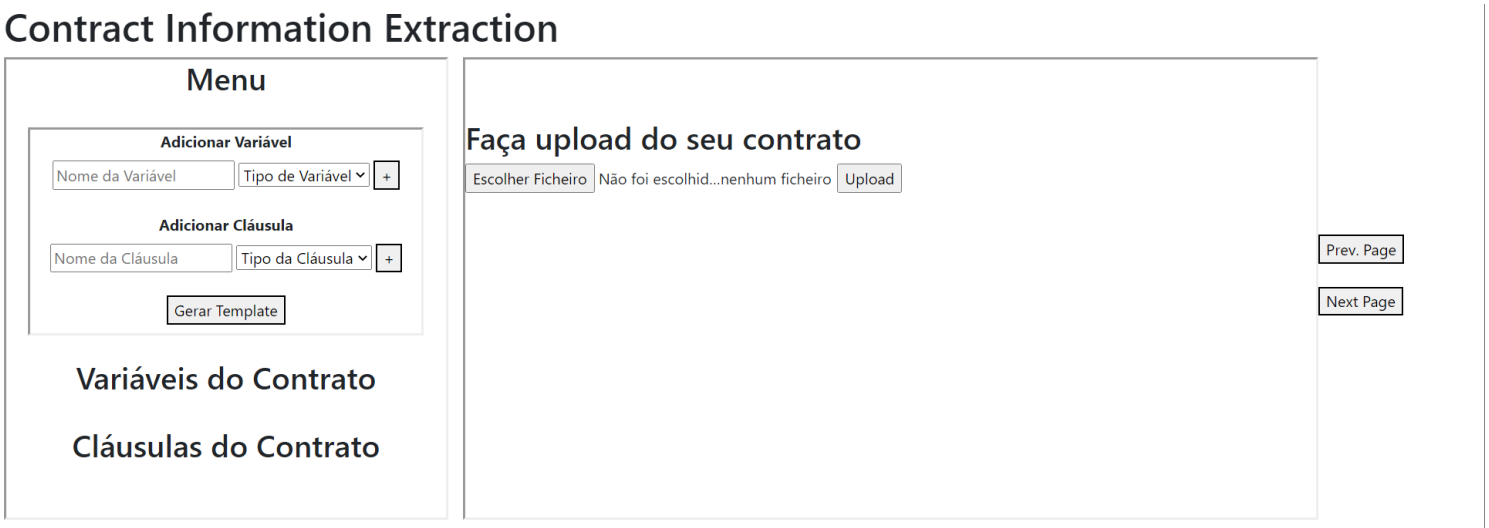


Figura 19 - Página web do processo de digitalização de contratos.

Para simplificar esta prova de conceito, foi criado um documento de testes simples com pouco texto, como se pode observar na *figura 20*.

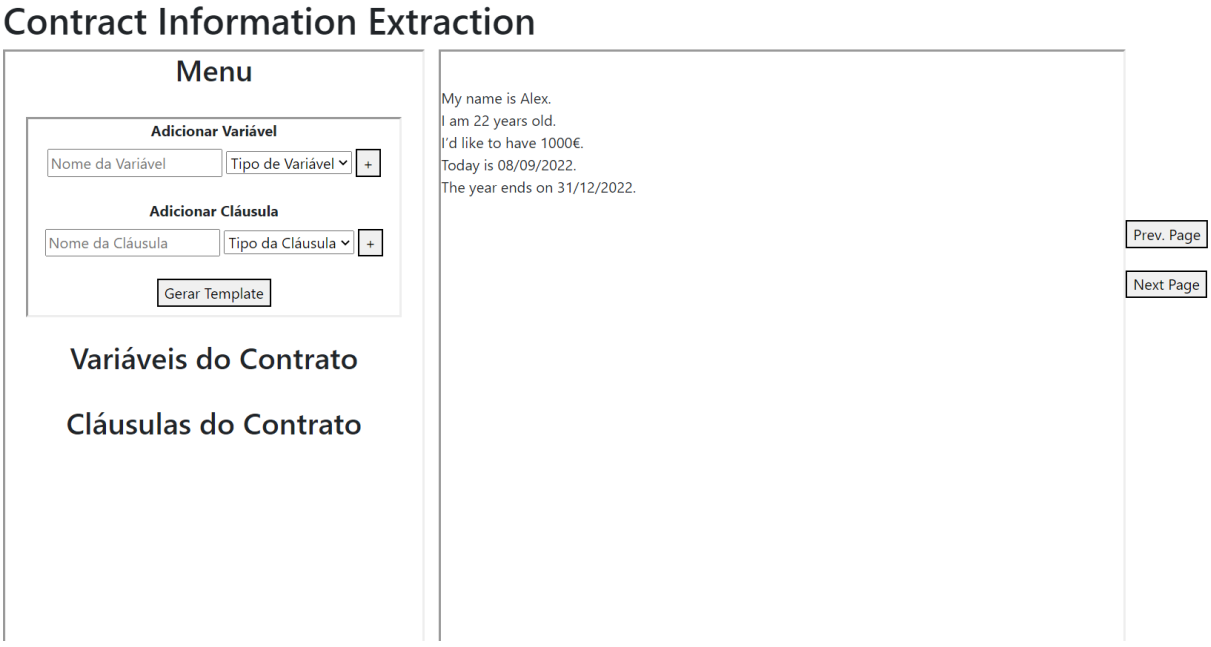


Figura 20 - T texto de um contrato carregado na página web.

Depois do texto do contrato ser obtido, o utilizador pode realizar o resto do processo utilizando o menu realçado na *figura 21*.

Figura 21 - Menu de adição de variáveis e cláusulas.

5.2 Criação das Variáveis

A *figura 22* demonstra a criação de uma variável, na qual o utilizador deve definir o nome e selecionar o seu tipo.

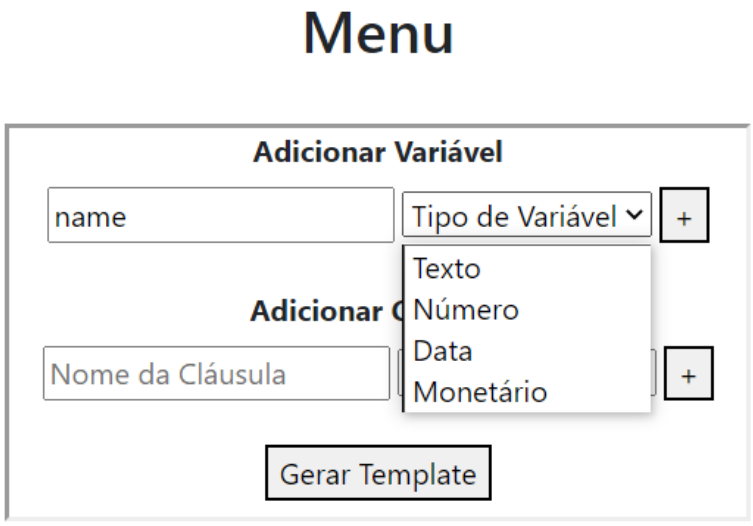


Figura 22 - Criação de uma variável.

De seguida, o utilizador deve selecionar uma parte do texto do contrato, exemplificado na *figura 23*, associando então não só o texto à variável, mas também a sua posição no contrato.

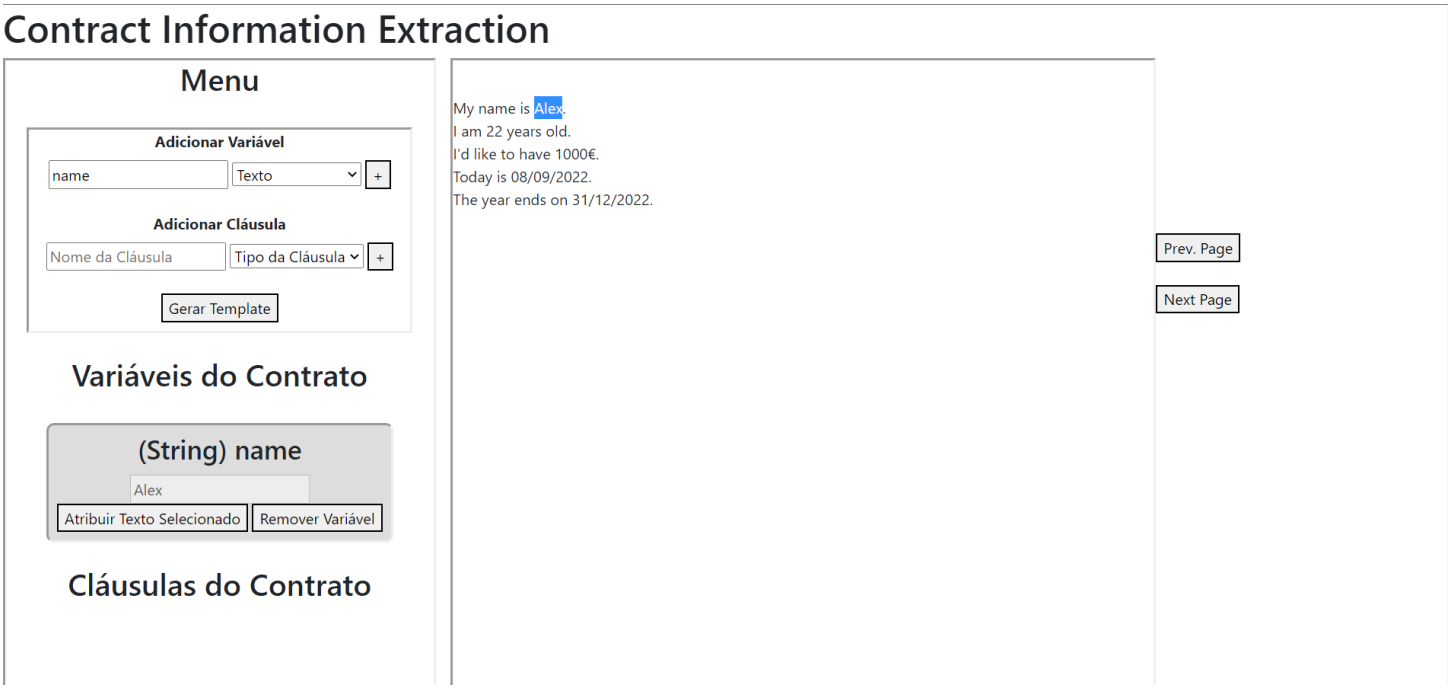


Figura 23 - Associação de texto a uma variável.

5.3 Criação das Cláusulas

Semelhante às variáveis, a criação das cláusulas visualizada na *figura 24,* passa pela definição do seu nome e tipo, na qual neste caso, o seu tipo define também a funcionalidade da cláusula.

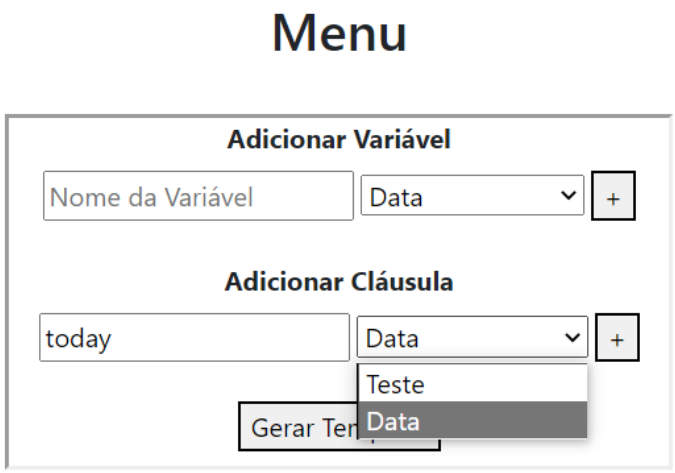


Figura 24 - Criação de uma cláusula.

No último exemplo foi escolhida uma cláusula do tipo “Data”, este tipo de cláusula verifica e compara a data atual a uma data limite definida no contrato. Como tal, esta data limite deve ser definida como uma variável que será por fim associada à cláusula, como demonstrado na *figura 25*.

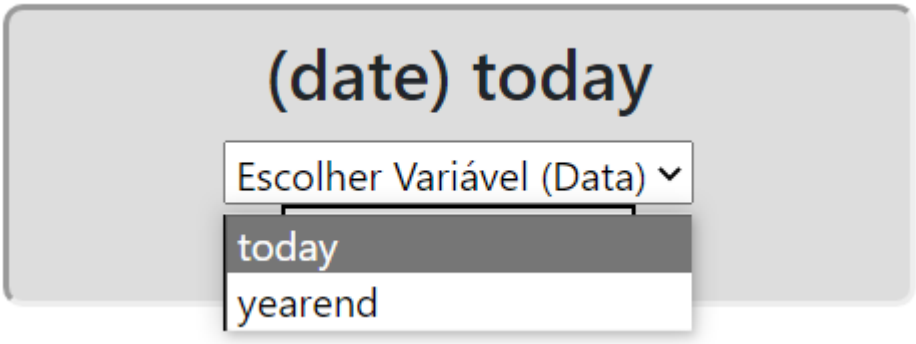


Figura 25 - Associação de uma variável a uma cláusula executável.

5.4 Geração do Smart Template

Assim que o utilizador finaliza ambos o processo de criação de variáveis e de cláusulas, deve pressionar o botão “Gerar Template”, como demonstrado na *figura 26*, que envia os dados do contrato recolhidos e definidos pelo utilizador para o servidor, onde serão processados para criar o *Smart Template*.

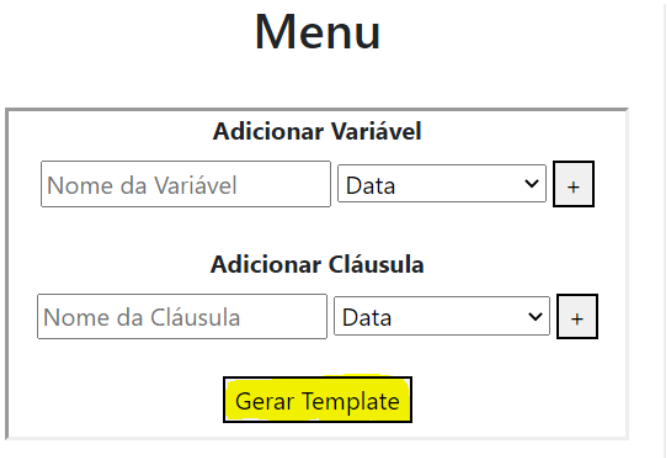


Figura 26 - Botão "Gerar Template".

Inicialmente, a parte do algoritmo dedicada à conversão do texto simples para o formato *Cicero*, passou pelo uso de expressões regulares ou *Regex,* devido à sua capacidade de deteção e substituição de porções de texto, sendo esta a funcionalidade necessária para a estruturação das variáveis. A conversão de expressões regulares em autómatos de estado finito faz parte da informática há décadas, usando transdutores de estado finito, criados e compilados pelo uso de expressões regulares, podemos resolver muitos problemas de engenharia da linguagem natural de uma forma mais eficiente (Karttunen et al., 1996). No entanto, no caso de um contrato, o uso das expressões torna-se mais complicado pois as partes de texto que queremos realçar podem ser idênticas a outras porções do texto. Este caso foi devidamente explorado e determinou-se que os pedaços idênticos dos textos de um contrato criam uma ambiguidade que torna o uso de *Regex*, pouco fiável. Alternativamente, desenhou-se uma solução em que o texto inteiro do contrato, ou pelo menos os parágrafos relevantes, eram utilizados com os algoritmos de expressões regulares: no entanto na prática a solução acabou por ser pouco eficiente e mal estruturada. Acabou por se criar outro método baseado numa melhor estruturação dos dados do protótipo, consistindo na adição de campos como a posição inicial e final do texto quando as variáveis eram definidas. De seguida foi apenas utilizado um simples algoritmo de *string slicing*, criando assim os ficheiros *Cicero* de modo fiável e previsível.

A geração dos ficheiros *Concerto* e *Ergo,* do *template* passou pela análise do tipo das variáveis e das cláusulas e das suas variáveis associadas. A cada tipo de cláusula está associada uma função especifica que gera corretamente o código relativo ao seu tipo, definindo também os seus *inputs* e *outputs* de acordo com a sua funcionalidade. Adicionalmente, o *Accord Project* requer também que cada *template* inclua ficheiros adicionais, tais como um *JSON* com os metadados do contrato, e uns ficheiros com dados de um contrato exemplo usados para testar a execução do *Smart Template.* A *figura 27* demonstra a estrutura final do *Smart Template*, estruturado como uma directoria.

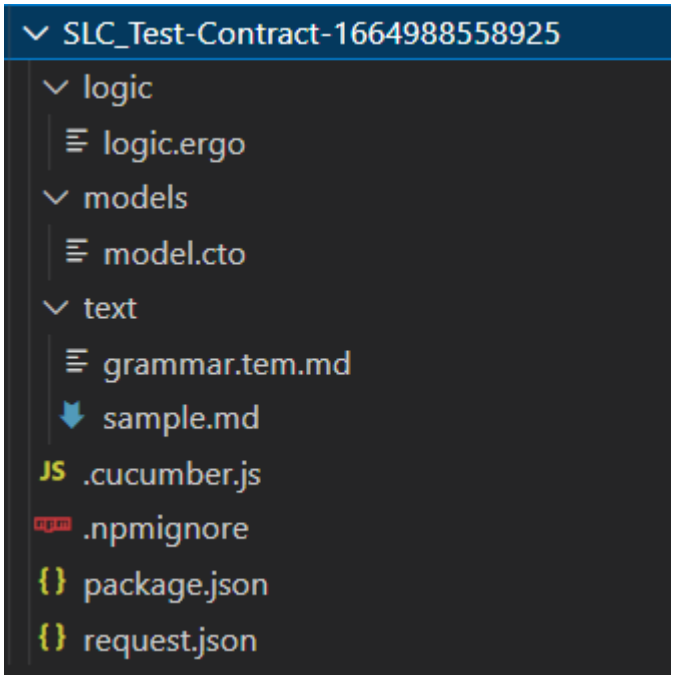


Figura 27 - Diretoria e estrutura representativa de um Smart Template gerado.

5.5 Execução do Smart Template

Através das bibliotecas do *Accord Project,* é teoricamente possível executar o contrato do *Smart Template.* No entando, ao utilizar a biblioteca na prática surgiram numerosos erros, que após investigados durante semanas e resolvidos, revelaram um erro fatal final. Este erro devia-se ao *download,* realizado pela biblioteca, de uma dependência no formato de um contrato base do *Accord Project* que é fundamental para a execução dos *templates*. Especulou-se que o erro se devia ao *template* gerado pelo processo realizado, mas acabou por se confirmar que de facto, o erro era proveniente da dependência, sendo impossível continuar por esse caminho. No entanto, felizmente, a *Accord Project* fornece no seu *website,* um "*Template Studio*" funcional, apresentado na *figura 28*, onde é possível inserir o código do *Smart Template* gerado pelo processo de digitalização.

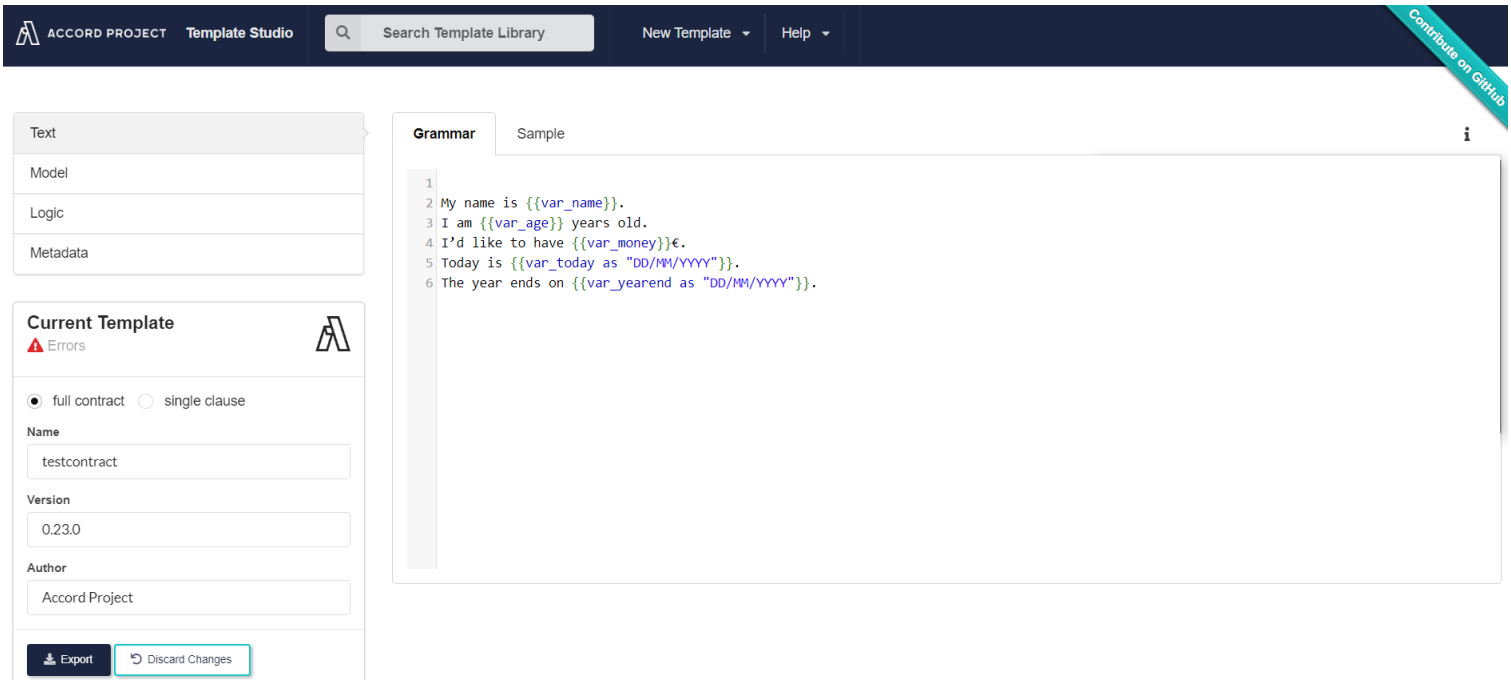


Figura 28 - Template Studio do Accord Project com um template carregado.

Através deste método, foi confirmado e validado o código gerado, sendo possível a realização de testes através das cláusulas executáveis geradas. Um destes testes foi a execução de uma cláusula do tipo "Data", exemplificado na *figura 29*, em que, após fornecer uma data como *input,* a cláusula validou e executou corretamente, fornecendo um *output* em que a data foi rejeitada por exceder o limite definido.

Figura 29 - Exemplo de execução de uma cláusula.

6. RESULTADOS E CONCLUSÃO

6.1 Cumprimento dos Objetivos

Os capítulos 2 e 3, que estudaram o *Accord Project* e os *Smart Legal Contracts* responderam ao primeiro objetivo definido, "Investigar tecnologias atuais para estruturar documentos legais em formatos digitais executáveis". A informação obtida desta investigação foi vital ao resto do projeto, especialmente ao desenvolvimento representado nos capítulos 5 e 6.

O capítulo 4 e a análise de custo-benefício do capítulo 5.1, serviram para o cumprimento do objetivo de "Investigar e analisar bibliotecas open source de Natural Language Processing, Document Understanding e de outros tipos de compreensão de texto". A exploração do mercado e de tecnologias open source foi essencial de um ponto de vista técnico para uma abordagem fundamentada do desenho dos processos de digitalização apresentados no capítulo 5, e para o desenvolvimento da prova de conceito exemplificada no capítulo 6.

O capítulo 5, representando a solução desenvolvida durante o curso do projeto, atinge os objetivos de "Manter o custo da solução, quer este seja monetário ou temporal, o mais baixo possível", devido à análise de custo-benefício no capítulo 5.1, e "Idealizar e planear um ou mais processos de digitalização de fácil utilização por parte do utilizador final", como demonstrado no resto do capítulo. Os processos de digitalização desenvolvidos surgem através da aplicação teórica e prática dos conceitos e tecnologias agregadas durante o resto da dissertação. Não se considera que foi atingido o potencial total do projeto apenas com estes processos, no entanto formam uma fundação sólida para uma expansão informada deste.

O capítulo 6, foi criado para completar o objetivo: "Construção de uma prova de conceito que demonstre um ou mais processos de digitalização previamente projetados". Nota-se que apesar dos erros e problemas encontrados, foi possível demonstar a validade do processo de digitalização exemplificado.

O objetivo de "Validação do processo e prova de conceito desenvolvidos com as organizações interessadas", será respondido no próximo capítulo 7.2.

6.2 Validação com a Empresa Acolhedora

Para a validação da solução com a empresa acolhedora, foi realizada uma reunião com um representativo que forneceu o seu *feedback*. Após uma demonstração completa da prova de conceito realizada no capítulo 6, foram colocadas questões as quais as suas respostas foram transcritas neste capítulo.

De que modo foram úteis as investigações realizadas no capítulo 3 e 4?

A empresa tem um grande interesse na informação obtida nestes capítulos. A análise do funcionamento do *Accord Project* e do seu estado atual traz grande valor para o processo de decisão de negócio. A investigação das tecnologias de *document processing* foram também valiosas para uma melhor compreensão do mercado atual e de como os desenvolvimentos mais recentes se relacionam e comparam entre si.

Os processos apresentados no capítulo 5 são viáveis para a empresa?

Sim, os processos são claros e simples. A análise custo-benefício foi apreciada devido à compreensão adquirida sobre o nível tecnológico necessário para construir os vários aspectos do projeto.

A solução produzida é uma boa resposta ao desafio proposto pela empresa?

A solução acabou por ser devidamente satisfatória para os interesses da empresa, apontando para o valor obtido sobre a compreensão e funcionamento desta tecnologia de digitalização e de *Smart Templates*.

Existe um futuro para este projeto?

O sistema foi considerado como uma boa base para um projeto empresarial completo, sendo que há um futuro determinado para a tecnologia estudada e desenvolvida.

6.3 Conclusão

A investigação do *Accord Project,* aconselhada pela empresa acolhedora, foi definida como uma tecnologia com o fim de estruturar documentos legais em formatos digitais executáveis, levando à aprendizagem de como realizar e estruturar *Smart Templates* e como estes eram compostos. A investigação da *Amazon Comprehend* e *Microsoft Azure AI* permitiu realizar um contraste com as ferramentas *open source* examinadas, em que se observou uma enorme diferença de capacidades entre as soluções comerciais e bibliotecas disponíveis abertamente. Com a análise custo-benefício, foi possivel recuperar o foco do projeto e realizou-se uma escolha de tecnologias, tendo em conta as necessidades e restrições de uma perspetiva empresarial e organizacional, impactando o processo de digitalização desenhado. Através da realização de vários diagramas e do seu aprofundamento, definiu-se dois processos de digitalização de contratos legais.

A prova de conceito acabou por ser completamente funcional no que toca à geração de *Smart Templates*, infelizmente a execução destes foi problemática. Mas, utilizando o "*Template Studio*" da *Accord Project,* foi possível validar, executar e testar o seu código, sendo este corretamente gerado e fiável, determinando a solução como sendo devidamente robusta.

Apesar de alguns problemas e contratempos, o projeto acabou por ser considerado um sucesso ao nivel empresarial. A investigação e experimentação com as tecnologias abordadas foi interesante e serviu para abranger o conhecimento e dominio sobre estas do aluno. Com este tipo de projetos existem sempre facetas a melhorar ou novas funcionalidades a implementar, pelo que foram tomadas as seguintes considerações futuras.

6.4 Considerações Futuras

Devido à falta de segurança dos contratos electrónicos básicos, propõe-se que todos estes tipos de contratos envolvam a assinatura digital como elemento primário como forma de autenticação e de melhorar a integralidade e a gestão centralizada (Yu et al., 2009).

Foi explorada também a RPA, ou Robotic Process Automation, que pode simular a "actividade" dos seres humanos através da utilização de um computador através da utilização do rato e das entradas do teclado. Geralmente, a RPA está dividida em tarefas de "assistente pessoal",

que simulam o comportamento interactivo da interface do programa humano e automatizam o funcionamento desta interface, e as tarefas "não assistidas", que utilizam APIs, SDKs e interfaces de dados para completar tais tarefas. A utilização de RPA traz vantagens, como a sua adaptação, sendo capaz de completar uma variedade de sistemas de informação e adaptações de dados. Também é não invasiva e segue um processo baseado em regras, não havendo assim o risco de fugas de informação e manipulação incorrecta de erros, sendo altamente eficiente quando comparado com a entrada humana. Com isto em mente, a RPA pode ser utilizada para executar tarefas de processamento de documentos tais como, processos de classificação automática, processos de identificação OCR de ficheiros, segmentação de texto, extracção de características de texto e classificação inteligente de texto (Ling et al., 2020). A RPA pode-se inserir no projeto como um passo adicional de automatização do processo de digitalização, permitindo um preenchimento automático dos *Smart Templates* ou até uma análise do conteúdo do contrato, extraindo automáticamente as cláusulas e os seus tipos.

O próximo grande passo para os processos de digitalização de contratos, baseia-se então, na deteção automática de cláusulas, tendo esta tarefa um alto nível de complexidade devido à variabilidade da linguagem natural, a existência de múltiplos idiomas e à falta de estruturação linguagem aplicada às cláusulas no sistema legal.

Adicionalmente, outra rota que a solução pode seguir será um alargamento da estrutura do contrato. Para abordar a não uniformidade de documentos técnicos são necessárias duas etapas. O primeiro consiste na detecção de linguagem não uniforme, com a ajuda de um método de classificação de *machine learning*. O segundo aplica uma correcção da linguagem não uniforme, que decide como corrigir o texto, substituindo as frases por uma escrita técnica apropriada. Determina-se que estes sistemas correspondem aos níveis de desempenho alcançados pelos anotadores especializados (Wang et al., 2021). A correção automática de texto para um formato especializado, poderá ser chave para a digitalização automatizada de contratos aplicados a *Smart Templates*, sendo que idilicamente os documentos poderiam ser apenas semelhantes em estrutura, em vez do requerimento atual de que sejam completamente idênticos, à parte das suas variáveis.

BIBLIOGRAFIA

Alani, H., Kim, S., Millard, D. E., Weal, M. J., Hall, W., Lewis, P. H., & Shadbolt, N. R. (2003). Automatic Ontology-Based Knowledge Extraction from Web Documents. *IEEE Intelligent Systems*, *18*(1), 14–21. https://doi.org/10.1109/MIS.2003.1179189

Bottou, L., Bengio, Y., & Le Cun, Y. (1997). Global training of document processing systems using graph transformer networks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 489–494. https://doi.org/10.1109/cvpr.1997.609370

Chalkidis, I., Androutsopoulos, I., & Michos, A. (2017). Extracting contract elements. *Proceedings of the International Conference on Artificial Intelligence and Law*, 19–28. https://doi.org/10.1145/3086512.3086515

Chalkidis, I., Fergadiotis, M., & Malakasiotis, P. (2019). *Neural Contract Element Extraction Revisited*. *NeurIPS*, 1–4. https://openreview.net/pdf?id=B1x6fa95UH

Dale, R. (2019). Law and word order: NLP in legal tech. *Natural Language Engineering*, *25*(1), 211–217. https://doi.org/10.1017/S1351324918000475

Del, M., & Jr, R. (2019). *Student Paper Comparison System using Kolmogorov Complexity and Diff Algorithm*. *36*(1), 9–27.

Hunt, J. W., & Mcilroy, M. D. (1976). *An Algorithm for Differential File Comparison*. 1–9.

Karttunen, L., Chanod, J. P., Grefenstette, G., & Schille, A. (1996). Regular expressions for language engineering. *Natural Language Engineering*, *2*(pt 4), 305–328. https://doi.org/10.1017/S1351324997001563

Koopmans, C., & Mouter, N. (2020). Cost-benefit analysis. *Advances in Transport Policy and Planning*, *6*(October), 1–42. https://doi.org/10.1016/bs.atpp.2020.07.005

Kwok, T., & Nguyen, T. (2006). An automatic method to extract data from an electronic contract composed of a number of documents in PDF format. *CEC/EEE 2006 Joint Conferences*, *2006*, 33–37. https://doi.org/10.1109/CEC-EEE.2006.13

Ling, X., Gao, M., & Wang, D. (2020). Intelligent document processing based on RPA and machine learning. *Proceedings - 2020 Chinese Automation Congress, CAC 2020*, 1349–1353. https://doi.org/10.1109/CAC51589.2020.9326579

Norta, A. (2018). Self-Aware Smart Contracts with Legal Relevance. *Proceedings of the International Joint Conference on Neural Networks*, *2018-July*, 1–8. https://doi.org/10.1109/IJCNN.2018.8489235

Pereira, J. C. L. (2019). *Smart Legal Contracts: A Génese da Revolução Digital no Direito dos Contratos*. 154. http://repositorium.sdum.uminho.pt/handle/1822/71811%0Ahttp://repositorium.sdum.uminho.pt/

Rao, S., & Havewala, A. M. (2019). Smart Legal Contract Migration using Machine Learning. *Proceedings - 2019 1st International Conference on Digital Data Processing, DDP 2019*, 65–69. https://doi.org/10.1109/DDP.2019.00022

Shah, P., Joshi, S., & Pandey, A. K. (2018). Legal clause extraction from contract using machine learning with heuristics improvement. *2018 4th International Conference on Computing Communication and Automation, ICCCA 2018*, 1–3. https://doi.org/10.1109/CCAA.2018.8777602

Tang, Y. Y., De Yan, C., & Suen, C. Y. (1994). Document Processing for Automatic Knowledge Acquisition. *IEEE Transactions on Knowledge and Data Engineering*, *6*(1), 3–21. https://doi.org/10.1109/69.273022

Upadhyay, K., Dantu, R., Zaccagni, Z., & Badruddoja, S. (2020). Is Your Legal Contract Ambiguous? Convert to a Smart Legal Contract. *Proceedings - 2020 IEEE International Conference on Blockchain, Blockchain 2020*, 273–280. https://doi.org/10.1109/Blockchain50366.2020.00041

Wang, W., Islam, A., Moh'd, A., Soto, A. J., & Milios, E. E. (2021). Nonuniform language in technical writing: Detection and correction. *Natural Language Engineering*, *27*(3), 293–314. https://doi.org/10.1017/S1351324920000133

Yu, X., Chai, Y., & Liu, Y. (2009). A secure model for electronic contract enactment, monitoring and management. *2nd International Symposium on Electronic Commerce and Security, ISECS 2009*, *1*, 296–300. https://doi.org/10.1109/ISECS.2009.184

Zhang, K., & Dell, M. (2019). Information Extraction from Text Regions with Complex Tabular Structure. *Nipsw*, *NeurIPS*.