UNIVERSIDADE FEDERAL DE UBERLÂNDIA-UFU
FACULDADE DE ENGENHARIA ELÉTRICA- FEELT
GRADUAÇÃO EM ENGENHARIA BIOMÉDICA

LUCAS GUARAGNA GUEDES

GERENCIAMENTO DE PROCESSOS DE NEGÓCIOS COM BPM E AUTOMATIZAÇÃO DE ATIVIDADES E TAREFAS COM PYTHON

UBERLÂNDIA-MG
AGOSTO DE 2022

UNIVERSIDADE FEDERAL DE UBERLÂNDIA-UFU
FACULDADE DE ENGENHARIA ELÉTRICA- FEELT
GRADUAÇÃO EM ENGENHARIA BIOMÉDICA

LUCAS GUARAGNA GUEDES

GERENCIAMENTO DE PROCESSOS DE NEGÓCIOS COM BPM E AUTOMATIZAÇÃO DE ATIVIDADES E TAREFAS COM PYTHON

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do título de Bacharel em Engenharia Biomédica pela Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia, sob orientação do Prof. Dr. Augusto Wohlgemuth Fleury.

UBERLÂNDIA-MG
AGOSTO DE 2022

UNIVERSIDADE FEDERAL DE UBERLÂNDIA-UFU
FACULDADE DE ENGENHARIA ELÉTRICA- FEELT
GRADUAÇÃO EM ENGENHARIA BIOMÉDICA

LUCAS GUARAGNA GUEDES

GERENCIAMENTO DE PROCESSOS DE NEGÓCIOS COM BPM E AUTOMATIZAÇÃO DE ATIVIDADES E TAREFAS COM PYTHON

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do título de Bacharel em Engenharia Biomédica pela Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia, sob orientação do Prof. Dr. Augusto Wohlgemuth Fleury.

BANCA EXAMINADORA:

Prof. Dr. Augusto Wohlgemuth Fleury Veloso da Silveira

Prof. Dr. Gustavo Brito de Lima

Prof. Dr. Sérgio Ricardo de Jesus Oliveira

DEDICATÓRIA

Dedico este trabalho aos meus pais, Rebeca Rezende Guaragna Guedes e Edward Ferreira Guedes Filho, e minha irmã, Giovana Guaragna Guedes, e agradeço por todo esforço, apoio, carinho, dedicação, amor, paciência, ajuda, companheirismo, que me deram ao longo dos anos de graduação e que me dão até hoje.

AGRADECIMENTOS

Meus agradecimentos a Universidade Federal de Uberlândia por fornecer um curso de graduação tão magnífico, que é a Engenharia Biomédica, bem como a todos os docentes envolvidos no processo de criação, manutenção e inovação do curso: Adriano Andrade, Adriano Pereira, Alcimar Barbosa, Ana Claudia Patrocínio, Fernando Pasquini, Márcia Simbara, Selma Milagre e Sérgio Ricardo de Jesus Oliveira.

Agradeço meu orientador, Augusto Fleury, pela orientação durante a formulação do TCC, bem como a estruturação do assunto como um todo, disposição e força de vontade.

No processo de graduação em Engenharia Biomédica, conheci pessoas magníficas, amizades as quais levarei para vida e, acredito que sem elas meu desempenho, tanto emocional quanto profissional, não teriam sido o mesmo. Foram pessoas que me ajudaram a crescer como ser humano e agregaram muito em minha vida. Cada uma delas tem um lugar especial em meu coração. São elas: André Trindade (*Jotinha*), Bianca Souza (*Bianquinha*), Camila Fernanda (*Camis*), Clóvis Júnior (*Crovim*), Daniel Costa (*Daniboy*), Eduardo (*Benson*), Gabriel Junqueira, Giovana Saraiva (Gio), Giovanna Reis (Jojo), Guilherme Rossin (*Guigui*), Gustavo Soubihe, Leonardo Matos (*Leo*), Lucas Fernandes Morais, Marcos Sírio (*Marquinho*), Maria Laura Camargos (Mary) e família, Paulo Amaro (*Paulitcho*), Rafael Canuto (*Cannes Man*), Raphael Leonardo, Poliana e por último e não menos importante, Victor Bértoli (*Vitinho*).

“Escrever nossa própria história pode ser difícil, mas não é tão duro quanto passar a vida fugindo dela. Aceitar nossas vulnerabilidades é arriscado, mas não é tão perigoso quanto desistir do amor, do pertencimento e da alegria, que por outro lado, são as experiências que nos deixam mais vulneráveis. Somente quando tivermos coragem suficiente para explorar a escuridão, descobriremos o poder infinito da nossa luz” – Brené Brown

RESUMO

Visando a metodologia qualitativa e o objetivo didático, este trabalho de conclusão de curso trata do assunto de gestão de processos de negócios, desde seu entendimento até a transformação de um processo como um todo, referenciado pelo guia BPM CBOK 3.0. Após a consolidação das práticas de BPM, são citadas algumas bibliotecas *python* para a automatização de atividades e tarefas comuns dentro de um negócio, bem como uma aplicação de exemplo. Com isso, este trabalho se conclui com uma proposta de aplicação da junção das práticas de BPM com a automatização *python* gerando uma hipótese de transformação de processo e mudança de paradigma: redirecionamento do foco para processos de maior valor agregado com a automatização de atividades e tarefas.

ABSTRACT

Aiming a qualitative methodology and a didactic objective, this term paper and undergraduate thesis deals with the discipline of business process management, from its understanding to its transformation, referenced by the BPM CBOK 3.0 Guide. After the consolidation of BPM practices, some *python* libraries are cited for the automation of common tasks and activities within a business, as well as an example application. Therefore, this final paper ends with a proposal of application: the union of BPM practices and *python* automation; this proposal generates a hypothesis of process transformation and paradigm change: focus redirection to higher value processes by automating tasks and activities.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

- Processo AS-IS – Processo como ele é, atualmente;
- Processo TO-BE – Processo como ele será, futuramente;
- BPM – *Business Process Management*, ou Gerenciamento de Processos de Negócio;
- SIPOC – *Suppliers*, *Input*, *Process*, *Output* e *Clients* (Fornecedores, Entrada, Processo, Saída e Clientes);
- *Script* – são as linhas de código de um programa, a programação em si.
- UMC – Uberlândia Medical Center: Estabelecimento Assistencial de Saúde privado presente na cidade de Uberlândia, MG.

SUMÁRIO

1 INTRODUÇÃO

Desde o surgimento das indústrias, entre 1760 e 1840, na Inglaterra, podemos observar a busca da substituição de processos manuais e/ou repetitivos por processos automáticos menos trabalhosos:

- na 1ª Revolução Industrial (1760 a 1840), houve uma substituição de mão de obra artesanal por máquinas a vapor;
- na 2ª Revolução Industrial (Século XIX a XX), a eletricidade e as linhas de montagem possibilitaram a produção em massa;
- na 3ª Revolução Industrial (aproximadamente 1960), com o advento de semicondutores e, com isso, computadores mais potentes, criação e alta utilização da internet. Máquinas programáveis foram criadas, além da otimização de linguagens de programação.

Percebe-se, então, que tais revoluções ocorreram quando existiu um aumento significativo da tecnologia e essa, por sua vez, impacta, diretamente, em uma alteração profunda nas estruturas sociais (SCHWAB, 2016). É possível alegar, então, que as estruturas sociais são moldadas a partir das mudanças da tecnologia que, basicamente, busca essa **substituição de um trabalho mais complicado, demorado, manual e repetitivo por um mais ágil e tecnológico**.

Nesse sentido, este trabalho busca uma proposta de substituição da execução de processos burocráticos e repetitivos (descobertos pela aplicação das práticas de BPM) pela automatização de atividades e tarefas (com ferramentas de automatização do *Python*). Diante disso, este trabalho se organiza a partir da explicação das disciplinas de BPM (tópico 2.1) e citação de algumas ferramentas de automatização (tópico 2.2), contextualizando o leitor para a proposta de aplicação: automatização de processos.

Nos tópicos 1.1 e 1.2, a seguir, será apresentada uma breve introdução das práticas de BPM e das ferramentas de automatização com Python.

1.1. Gerenciamento de Processos de Negócio (BPM)

Com a evolução das empresas e, consequentemente, o aumento da necessidade de gestão empresarial, surge, em 2003 nos Estados Unidos, a ABPMP - Associação de Profissionais Gerenciamento de Processos de Negócio (*Association of Business Process Management*) –, uma instituição que busca promover boas práticas de gerenciamento de processos de negócios, se tornando global em 2010 (ABPMP, institucional. 2022).

A ABPMP Brasil foca na formação de profissionais passíveis de transformação de organizações brasileiras, públicas ou privadas, oferecendo melhores produtos e serviços com

uma maior produtividade e eficiência, menores desperdícios e defeitos. Com isso, a utilização de boas práticas de gerenciamento de processos de negócio gera, ao cliente e à sociedade como um todo, um maior valor agregado (ABPMP, institucional. 2022)

No trabalho de conclusão de curso da Engenheira de Produção Isabella Dothling Cardoso, de 2022, a adoção das práticas de BPM na melhoria de processos administrativos, especificamente, no processo de criação e registro de empresas júniores na UFOP, se mostrou efetiva: houve a identificação de gargalos de processo e uma proposta de melhoria para o processo em questão, além de se criar uma documentação robusta através da análise e mapeamento (CARDOSO, Isabella. 2022). No entanto, é possível observar que a aplicação de práticas de BPM se mostra necessária e bem-vinda, mesmo atualmente, e podem contribuir para a evolução dos negócios brasileiros como um todo.

1.2. Python e a Automatização de Processos

A busca da facilidade de gerenciamento não foi a única reação mundial acerca dos avanços tecnológicos da indústria. Tendo isso em vista, outra área que foi amplamente aperfeiçoada foi a programação. Nesse sentido, ao se pensar em uma linguagem de programação simples e ágil, *Python* é uma linguagem que cresceu muito desde sua criação, na década de 90 por Guido Van Hossum, e hoje em dia é uma das linguagens mais utilizadas no mercado (SILVA, 2019).

Uma das características da linguagem que a torna simples e rápida é o seu foco em desenvolvimento ágil: é possível fazer muito mais com muito menos, quando comparada a outras linguagens de programação. Isso possibilita o uso de metodologias como o RAD (Rapid Application Development – Desenvolvimento rápido de aplicações) a qual foca na redução de desperdícios através da maior qualidade dos códigos, menor tempo e custo de produção (SILVA, 2019).

Uma das possíveis aplicações do *python* é na automatização de tarefas, como visto no trabalho de conclusão de curso de João de Almeida Neto, o qual utiliza *webscrapping* que é, basicamente, a extração de dados da internet, para a automatização da tarefa de geração de ressuprimento no Laboratório de Análises Clínicas da Policlínica Médica do CBMDF. Tal aplicação trouxe uma redução de 30% de tempo de ciclo de processo, quando comparado ao processo padrão, além de possibilitar a execução automática de sua atualização (NETO, 2021).

Portanto, levando em consideração as práticas de BPM e a automatização de processos com *python*, este trabalho busca uma abordagem disciplinar/didática de ambos os temas, bem como uma sugestão da utilização da automatização de tarefas e atividades com python como proposta de mudança de paradigma.

1.3. Justificativa

Durante o estágio do autor na área funcional de Gestão de Processos do Hospital UMC (Uberlândia Medical Center), em 2021, a utilização das práticas de BPM – Modelagem, Análise, Gerenciamento de Desempenho, Desenho e Transformação de Processos – levou-o a pesquisar e estudar ferramentas de automatização de atividades e tarefas. Diante disso, *python* se destacou por ser uma linguagem de baixo nível, o que possibilita um maior controle e personalização de aplicações - o que não é encontrado em outras ferramentas de automatização tais como Zapier, Pluga e Integromat (alto nível).

A função de mapeamento e análise dos processos da área funcional "Financeiro" do UMC levou o autor a identificar a existência de atividades e tarefas manuais e repetitivas dentre as diversas analisadas. Uma vez que esse tipo de atividade pode provocar falhas humanas e demonstra baixo valor agregado ao cliente de processo, foi sugerido, pelo autor, a utilização do Python para automatização desse processo em questão e a líder da Gestão de Processos, na época, aprovou tal projeto piloto.

Por questões de mudança de governança e de função da área funcional de Gestão de Processos, tal projeto não foi concluído, porém apresentou alta aceitabilidade pelos executores e líderes de processo além de se mostrar inovador, eficaz e eficiente. Nesse sentido, a experiência adquirida instigou o autor a estudar mais sobre as práticas de BPM e as ferramentas de automatização de processos, gerando, assim, o nascimento do tema deste trabalho: BPM + Python.

A proposta da junção das práticas de gerenciamento de processos de negócio com a automatização de atividades e tarefas com *python* pode revolucionar as formas de se pensar e executar dentro de um negócio: se um processo não agrega valor real ao cliente e é repetitivo, manual e burocrático, ele será automatizado. Com tais processos automatizados, rápidos, eficazes e eficientes, os executores podem passar a focar em processos que agregam maior valor ao cliente, aumentando sua satisfação com o negócio.

A aplicação de tal proposta pode requerer a criação de uma nova função dentro da área funcional responsável pelos processos dentro de um negócio: engenheiro de automatização de processos com python (ou algo parecido); uma vez que existe uma necessidade do bom entendimento das práticas de BPM e *python*. Tal função deve ser encarregada por modelar, analisar, gerenciar desempenho, desenhar e transformar processos e, dentro da área de transformação de processos, deve ser capaz de aplicar a automatização com *python*. Pode-se

também explorar novas tecnologias de melhoria de processos e aplicá-las de acordo com o que se mostra necessário dentro do negócio em questão.

É muito comum encontrar processos manuais e burocráticos nos estabelecimentos assistenciais de saúde. Com isso, a aplicação deste trabalho tem o potencial de impactar o mercado hospitalar brasileiro de certa forma que possibilite a otimização do trabalho dos profissionais de saúde, visando maior bem estar do paciente e menor burocratização de atividades e tarefas que não agregam valor real ao cliente final.

Objetivos

1.3.1. Objetivo Geral

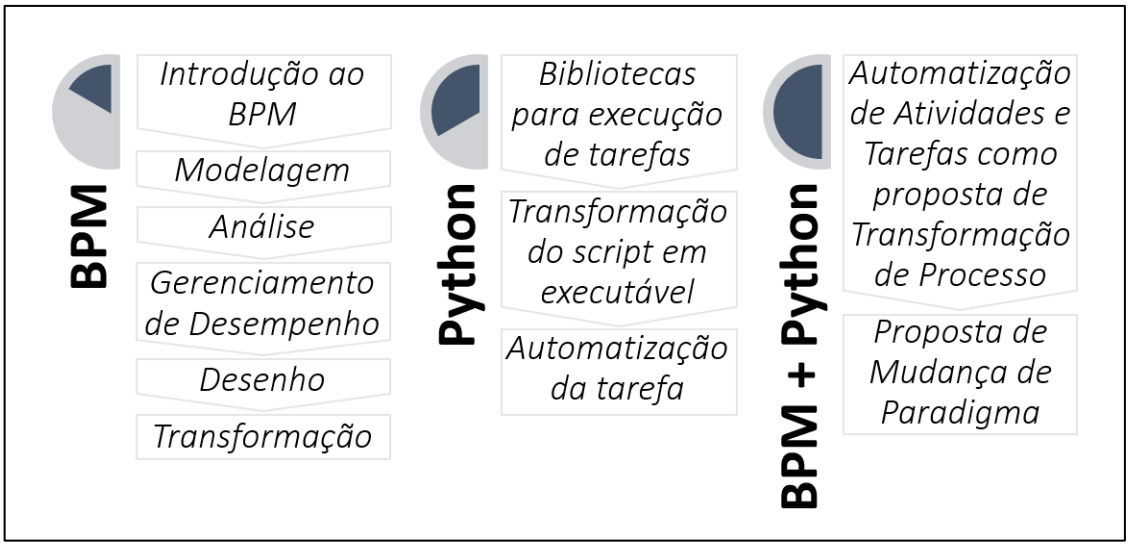
Analisar práticas de gestão de processos de negócio robustas para a confecção de processos AS-IS e propor uma aplicação da automatização de atividades e tarefas com a linguagem de programação *python* como possível abordagem de transformação de processo.

1.3.2. Objetivos Específicos

- Formular uma base para entendimento para as práticas de BPM através do estudo das disciplinas de BPM apresentadas no Guia BPM CBOK 3.0;
- Formular uma base para entendimento de automatização de algumas atividades e tarefas com Python;
- Apresentar uma aplicação da automatização de tarefas com *Python*

1.4. Estrutura do trabalho

Figura 1 - Estrutura do Trabalho de Conclusão de Curso de Lucas Guaragna



Fonte: autoria própria

A organização deste trabalho se mostra da seguinte maneira: no referencial teórico, as práticas de BPM, as quais se referem a todo entendimento de gerenciamento de processos, é apresentada de maneira didática (com todos os termos necessários para uma compreensão sólida acerca do assunto) e, em seguida, são apresentadas algumas ferramentas para automatização de atividades e tarefas com *python* de maneira simples e direta. Na discussão deste trabalho, encontra-se uma proposta de mudança de paradigma: a junção das práticas de BPM com a automatização de tarefas com Python. A Figura 1 demonstra tal divisão.

Dentro das práticas de BPM, é feita a divisão:

- **Introdução ao BPM**: apresentação de termos importantes para o bom entendimento das práticas de gerenciamento de processos de negócios, assim como valores e crenças necessárias para o sucesso da aplicação desta prática;
- **Modelagem de Processos**: aborda níveis de processos, qual notação será utilizada para documentação do processo bem como ferramentas utilizadas para tal;
- **Análise de Processos**: apresentação de algumas técnicas de análise, bem como critérios para a documentação do processo analisado com foco no sucesso da análise;
- **Gerenciamento de Desempenho de Processos**: mostra a importância de se mensurar um processo, além de como deve ser o foco da medição de desempenho de um processo;
- **Desenho de Processos**: cita algumas atividades-chave para a criação de um novo processo e importantes considerações para tal;
- **Transformação de Processos**: mostra as amplitudes de transformação de processos, bem como técnicas para o gerenciamento da mudança e a visão necessária para tal.

Agora, dentro da automatização de processos com *Python*, temos a seguinte separação:

- Ferramentas de execução de tarefas com *Python* (formulação de *scripts*):
  - Análise de dados com *pandas;*
  - Manipulação de pastas e arquivos de computador com *pathlib;*
  - Aquisição de dados em Bancos de Dados com *pyodbc;*
  - Envio de e-mails com *smtplib;*
  - *Webscrapping* com *selenium.*
- Transformação dos *scripts* em arquivo executável (“.py” para “.exe”):
  - Utilização da biblioteca *auto_py_to_exe.*
- Usando o agendador de tarefas do Windows para programar execução de programas em frequências específicas.

Após o referencial teórico, existe uma discussão sobre uma proposta de junção das práticas de BPM com a automatização de tarefas com *python* como hipótese de mudança de paradigma,

visando a transformação de processo, bem como uma **aplicação de automatização de processos**.

2 REFERENCIAL TEÓRICO

Antes de se transformar um processo, seja ele primário, de suporte ou de gerenciamento (explicados mais a frente), deve-se entender o processo "AS-IS", ou seja, como ele é de fato. Para tal, é necessário, primeiro, entender as práticas de Gerenciamento de Processos de Negócio (*Business Process Management* – BPM), o qual se mostra ser uma disciplina essencial para o entendimento de um processo.

Tendo isso em vista, o referencial teórico está dividido em duas partes: "BPM" e "Python e a Automatização de Atividades e Tarefas". A primeira parte busca identificar as principais práticas de BPM com o foco específico em automatização de processos. Uma vez consolidada e com o entendimento necessário de um processo, o Python – segunda parte do referencial teórico – pode surgir como ferramenta de automatização de processos: uma solução vista como transformação de processo para o BPM.

2.1. BPM

Business Process Management (BPM), ou Gerenciamento de Processos de Negócio em português, é definido, pelo guia *Business Process Management Common Book Of Knowledge 3.0* (BPM CBOK 3.0), como sendo uma disciplina gerencial que possibilita o estabelecimento de princípios e práticas os quais beneficiam as empresas que adotá-la. Por definição:

> Gerenciamento de Processos de Negócio (BPM – Business Process Management) é uma disciplina gerencial que integra estratégias e objetivos de uma organização com expectativas e necessidades de clientes, por meio do foco em processos ponta a ponta. BPM engloba estratégias, objetivos, cultura, estruturas organizacionais, papéis, políticas, métodos e tecnologias para analisar, desenhar, implementar, gerenciar desempenho, transformar e estabelecer governança de processos (BPM CBOK 3.0 – p. 40)

O BPM CBOK 3.0 foi concebido com o intuito de fornecer um panorama abrangente de conceitos, melhores práticas e lições compiladas pela ABPMP (associação internacional de profissionais de BPM) além de buscar uma padronização de vocabulário.

Dentro da disciplina de BPM existem diversas áreas do conhecimento: **Modelagem de Processos**, **Análise de Processos**, **Desenho de Processos**, **Gerenciamento e Desempenho de Processos** e **Transformação de Processos**. Tais áreas serão abordadas a seguir.

O presente documento **não** substitui a leitura do guia BPM CBOK 3.0, mas, **sim**, auxilia no seu entendimento e em sua aplicação neste trabalho.

2.1.1. Introdução ao BPM

Para enfrentar desafios de negócios globalmente diversificados, as organizações devem melhorar sua capacidade de resposta e adaptação a mudanças: tanto de mercado, quanto de demandas de clientes. Empresas que utilizam BPM têm a capacidade de trabalhar bem com a mudança – fator necessário para o bom funcionamento de empresas como um todo.

Com o intuito de facilitar a compreensão de termos utilizados, a seguir serão listadas definições de conceitos amplamente utilizados neste trabalho, além de algumas práticas importantes para o sucesso da aplicação de BPM:

2.1.1.1. Negócio

Definido como um conjunto de pessoas as quais executam uma sequência de atividades que entregam algum bem (produto ou serviço) para um determinado cliente, gerando retorno às partes interessadas. Pode ter caráter público ou privado, com ou sem fins lucrativos e de qualquer porte ou segmento (ABPMP, p.35, 2013).

2.1.1.2. Processo

Processo é uma sequência de atividades, as quais fornecem visão de sequência e fluxo, executadas por humanos ou máquinas para alcançar determinado resultado. É composto por atividades inter-relacionadas as quais solucionam uma questão específica (ABPMP, p.35, 2013).

2.1.1.3. Processo de Negócio

Se apresenta como uma sequência de atividades as quais entregam valor para os clientes, além de poder gerenciar outros processos, podendo ser de ponta a ponta, interfuncional e interorganizacional (ABPMP, p.35, 2013).

Os processos de negócio podem ser classificados em: processos primários, de suporte ou de gerenciamento, descritos a seguir.

2.1.1.3.1. Processo Primário

É um processo que agrega valor, diretamente, ao cliente. Podem ser chamados também de processos essenciais ou finalísticos (ABPMP, p.36, 2013).

2.1.1.3.2. Processo de Suporte

São processos que influenciam processos primários, porém não diretamente. Um processo de suporte pode dar suporte a processos primários ou a processos de suporte (processos de

suporte de segundo, terceiro, quarto nível, e assim por diante). Esse tipo de processo agrega valor a outros processos, e não ao cliente (ABPMP, p. 36, 2013).

2.1.1.3.3. Processo de Gerenciamento

São processos que têm o propósito de medição, monitoração, controle de atividades, administração do presente e futuro de negócio. Não agregam valor ao cliente, mas são essenciais para o cumprimento de objetivos e metas de uma organização (ABPMP, p. 37, 2013).

2.1.1.4. Instância de Processo

Instância de processo é cada execução de um processo. Em um processo de pesquisa de satisfação de cliente, cada pesquisa é uma instância de processo (ABPMP, p. 37, 2013).

2.1.1.5. Função de Negócio

Se refere a grupos de atividades relacionadas a objetivos particulares. São, normalmente, representados por setores dentro de uma organização. Cada função de negócio gerencia um recurso, como apresentado na tabela 1 (ABPMP, p. 38, 2013).

Tabela 1 - Funções de negócio e recursos gerenciados

Função de Negócio	**Recurso Gerenciado**
Financeiro	Dinheiro
Recursos Humanos	Profissionais
Gestão da Qualidade	Normas e padrões
Gestão de Processos	Processos
Patrimônio	Bens
Suprimentos	Materiais e insumos

Fonte: Adaptação de (ABPMP, p. 38, 2013)

2.1.1.6. Processo x Função

Como boa prática de BPM, é interessante dar mais importância ao processo de negócio do que a função de negócio em si, pois o processo de negócio engloba uma visão ponta a ponta a qual relaciona diferentes funções de negócio. Tal visão possibilita uma gestão horizontal da organização em si e a geração de valor se foca no cliente do processo (ABPMP, p. 39, 2013).

2.1.1.7. Valores, crenças, liderança e cultura

O compromisso com o valor do processo e do cliente é a base da prática de BPM. Se tais características são estabelecidas – oportunidades de crescimento de profissionais, discussões

abertas, relacionamento externo com clientes e fornecedores – o sucesso de uma organização se mostra mais factível (ABPMP, p. 39, 2013).

2.1.1.8. Compromisso Organizacional

Organizações tradicionais são, frequentemente, organizadas por funções de negócio e, para o sucesso da aplicação de práticas de BPM, a organização deve se organizar com foco em processos de negócio, de uma forma mais horizontal (como comentado no tópico 2.1.1.6). Para isso, a organização deve adotar novos papéis: donos de processo, analistas e arquitetos de processos. Os responsáveis pelo desenho de processos devem ter total abertura com gerentes além de terem apoio da liderança executiva (ABPMP, p. 40, 2013).

2.1.2. Modelagem de Processos

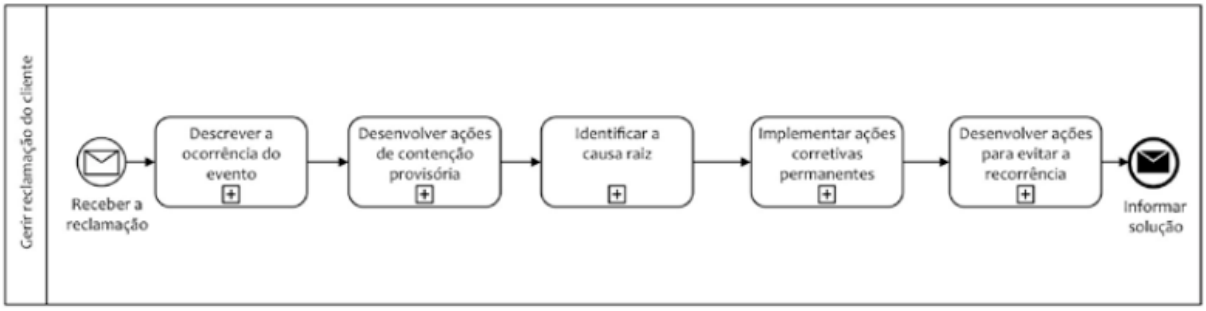
O conjunto de atividades envolvidas na criação da representação de processos de negócio, AS-IS ou TO-BE, se define por modelagem de processos de negócio (ABPMP, p. 72, 2013). Ela pode ser abordada em qualquer tipo de processo (primário, de suporte ou de gerenciamento) e em qualquer nível.

Um modelo de processo é representado por ícones os quais representam atividades, eventos, decisões, condições, entre outros e dependem da notação utilizada. O relacionamento entre esses ícones e dos ícones com o ambiente, o fluxo e como este se comporta são características representadas dentro de um modelo de processo (ABPMP, p. 72, 2013).

Existem diversos tipos de notações de modelagem de processos, cada uma com características e abordagens diferentes. No caso do presente trabalho, a notação que se mostrou mais adequada para a aplicação de automatização de processos é a BPMN: *Business Process Model and Notation*. Tal metodologia foi criada pela BPMI (*Business Process Management Initiative*) e tem uma gama de símbolos para a modelagem de diferentes aspectos de processos de negócio além de possibilitar a modelagem de diferentes níveis de processo (ABPMP, p. 79, 2013).

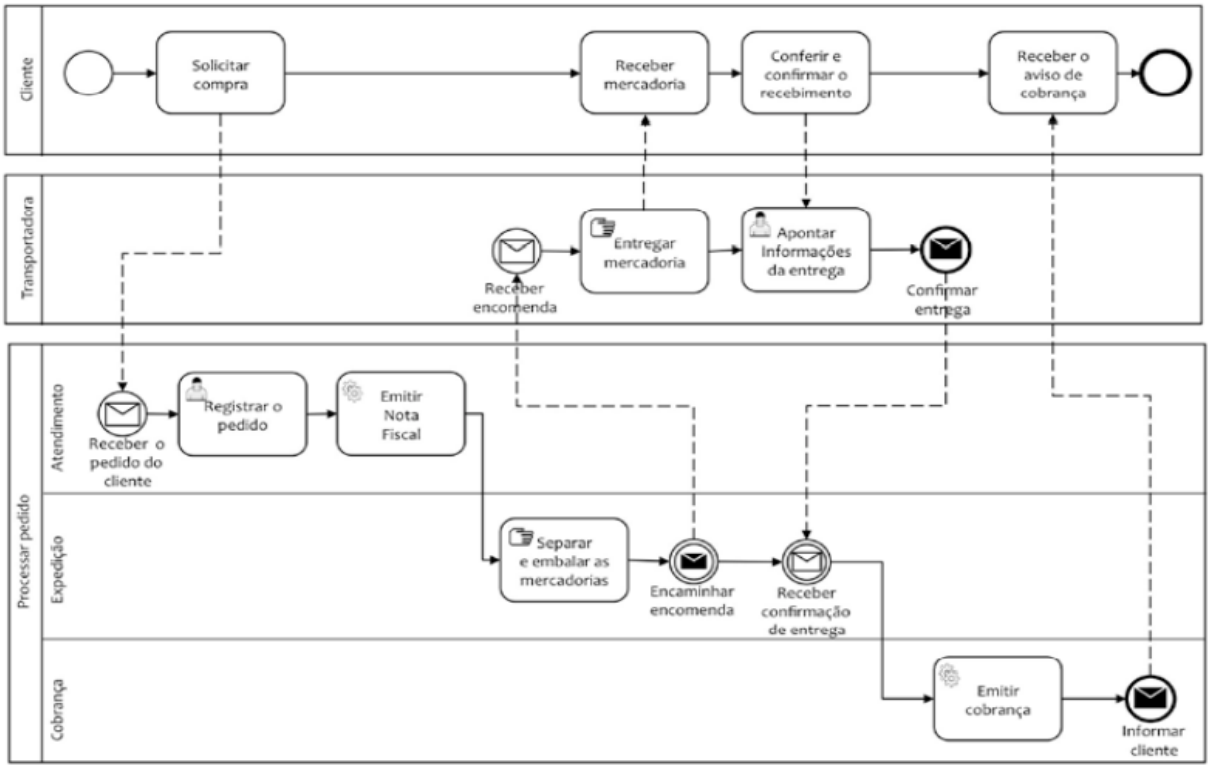
As Figuras 2 e 3 mostram exemplos da notação BPMN. Como comentado, BPMN nos possibilita a modelagem em qualquer nível (alto (Figura 2) ou baixo (Figura 3) nível).

Figura 2 - Fluxo em alto nível com BPMN



Fonte: BPM CBOK 3.0 (2013)

Figura 3 - Fluxo em baixo nível com BPMN

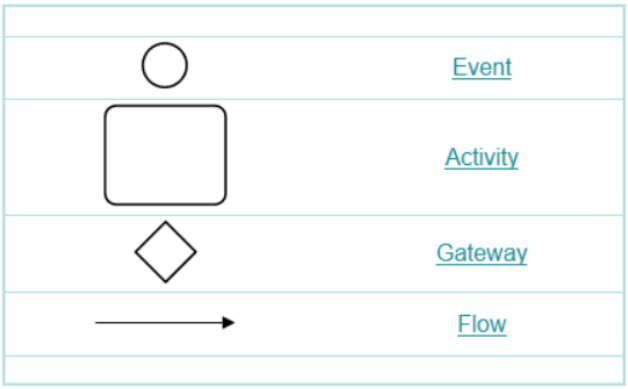


Fonte: BPM CBOK 3.0 (2013)

A utilização de somente uma notação de modelagem de processos pode gerar problemas: uma única notação pode não servir para todos os propósitos. Porém, como o foco deste estudo é em automatização de processos, a notação BPMN se mostra robusta.

Dentro desta notação, cada ícone recebe um significado específico. Acompanhando a figura 2: o primeiro círculo de borda simples, dentro da raia “Cliente”, define o evento de início do processo. Note que existem outros tipos de círculos dentro deste fluxo de processo, bem como quadrados e linhas de fluxo. A figura 3, abaixo, mostra o significado de um dos ícones mais importantes da notação BPMN.

Figura 4 - Ícones de Evento, Atividade, Gateway e Linhas

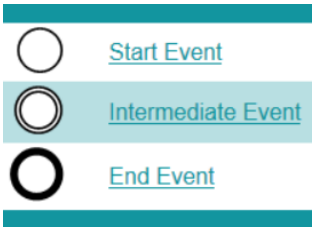


Fonte: BPMN Quick Guide (2022)

Cada um desses ícones pode apresentar variações: eventos podem ser de início, de intermédio ou de fim; atividades podem descrever tarefas e subprocessos além de poderem ter caráter manual, de envio ou recebimento de mensagens, entre outros; Gateways podem ser exclusivos, podem ser paralelos, entre outros; e Linhas de Fluxo podem indicar sequência, fluxo de envio de mensagens, associações, entre outros. Todas essas possibilidades de ícones se encontram no manual de utilização de BPMN ou no BPMN Quick Guide (BPMN Quick Guide, 2022).

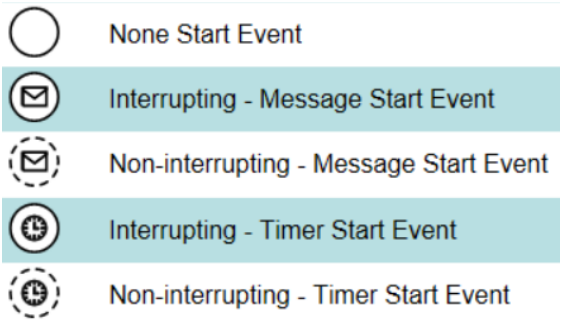
As Figuras 5 a 9 indicam alguns exemplos da diversidade de possibilidades de ícones da notação BPMN.

Figura 5 - Tipos de Evento



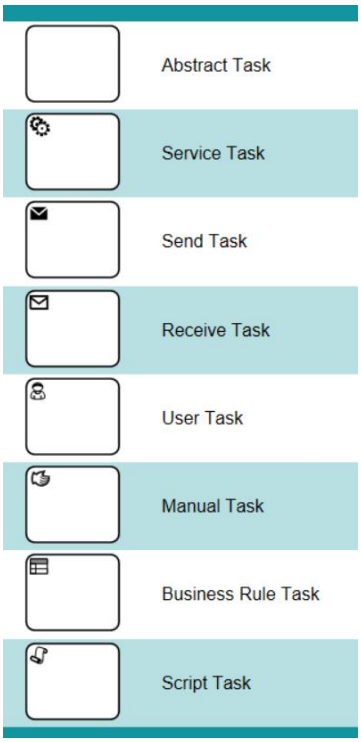
Fonte: BPMN Quick Guide (2022)

Figura 6 - Tipos de Evento de Início



Fonte: BPMN Quick Guide (2022)

Figura 7 - Tipos de Atividades

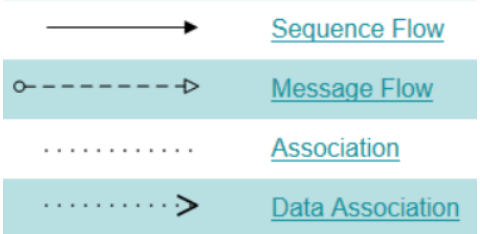


Fonte: BPMN Quick Guide (2022)

Figura 8 - Tipos de Gateway

Fonte: BPMN Quick Guide (2022)

Figura 9 - Tipos de Linhas de Fluxo

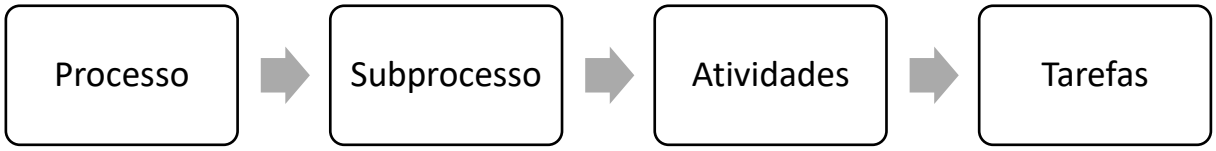


Fonte: BPMN Quick Guide (2022)

2.1.2.1. Níveis de Processo

Como visto nas figuras 1 e 2, processos podem apresentar níveis diferentes. Tais níveis costumam variar de nome, a depender da organização (ABPMP, p. 99, 2013), mas todos seguem a mesma lógica: de um nível mais alto, a um nível mais baixo, ou seja, do macro para o micro. Uma boa forma de dividir os níveis de processo é a descrita na Figura 10.

Figura 10 - Níveis de Processo



Fonte: autoria própria

2.1.2.2. Abordagens de Modelagem

A abordagem de modelagem varia de acordo com o propósito e escopo. Quando o foco é o fluxo de trabalho (Atividades e Tarefas) e o funcionamento de áreas funcionais a abordagem *bottom-up* (menor nível até maior nível de processo) (ABPMP, p. 99, 2013). Como a automatização será focada em atividades e tarefas, a abordagem *bottom-up* se apresenta mais adequada: entender como funcionam as atividades e tarefas, ou seja, entender o menor nível de processo.

2.1.2.3. Ferramentas de Apoio a Modelagem

2.1.2.3.1. Diagrama SIPOC

Como ferramenta de apoio de modelagem, o diagrama SIPOC pode facilitar o entendimento do processo como um todo. SIPOC é uma sigla: S de *Suppliers* (Fornecedores), I de *Input* (Entrada), P de *Process* (Processo), O de *Output* (Saída) e C de *Clients* (Clientes). Este diagrama é, comumente, feito em uma tabela, onde cada letra é uma coluna, como mostrado na tabela abaixo:

Tabela 2: Diagrama SIPOC.

Suppliers	**Input**	**Process**	**Output**	**Clients**

Fonte: Adaptado de BPM CBOK 3.0

Ao se entender quais são os fornecedores de processo, qual a entrada do processo, como o processo funciona, qual a saída do processo, e quais seus clientes, o entendimento do fluxo

de processo como um todo pode se mostrar mais visível. Temos um exemplo de processo de agendamento de consultas na Tabela 3.

Tabela 3: Diagrama SIPOC do Processo de Agendamento de Consultas

Suppliers	**Input**	**Process**	**Output**	**Clients**
Recepção	Dados do paciente	Receber ligação do paciente	Consulta agendada	Paciente
	Disponibilidade de Agenda	Realizar cadastro do paciente ou atualizar dados	Dados do paciente atualizados	Médico
		Verificar horários disponíveis para consulta		Enfermeiros
		Agendar consulta		Técnicos de Enfermagem
		Documentar Agendamento de consulta: ficha de atendimento		Recepcionistas
		Informar profissional para o qual a consulta foi agendada		Administrativo
		Confirmar presença do paciente com 1 dia de antecedência		Gestão da Qualidade

Fonte: Autoria Própria

2.1.2.3.2. Bizagi Modeler

Para a realização do desenho do fluxo de processos, uma boa ferramenta de modelagem que se mostra de fácil utilização e utiliza a notação BPMN como padrão é o Bizagi Modeler (GÓMEZ, G et. al., 2022). O Bizagi é um software especializado para modelagem de processos e, em seu site oficial, existem cursos gratuitos para o auxílio de sua utilização. Para mais informações: https://www.bizagi.com/pt/plataforma/modeler.

2.1.2.4. Possíveis problemas com modelagem de processos

O Guia BPM CBOK 3.0 traz alguns problemas que devem ser levados em consideração. A seguir, serão apontados tais problemas e em sequência sua relação com a automatização de processos.

Tabela 4: Problemas com Modelagem de Processos e suas soluções para este trabalho.

Problemas	**Soluções (deste trabalho - Automatização)**
Não considerar a modelagem como um problema de comunicação	A modelagem deve ser padronizada de certa forma que permita fácil compreensão.
Modelar sem um objetivo claro	Objetivo deste trabalho: automatização. O foco deve ser em processos passíveis de automatização.
Limitar-se a uma única notação	No caso de automatização, a notação BPMN se mostra robusta.
Não gerir expectativas em relação ao objetivo da modelagem	Ao se analisar o processo (tópico discutido mais a frente), os executores e líderes do processo devem estar cientes das possibilidades de transformação. Isso deve ser feito durante a análise e modelagem de processos.
Não identificar e organizar as partes antes de modelá-las	É importante entender o todo antes de entender uma atividade. O diagrama SIPOC pode ajudar nisso.

Fonte: Adaptação de BPM CBOK 3.0 p. 101

2.1.3. Análise de Processos

Depois da fase de modelagem de processos, a qual define qual abordagem, quais ferramentas de apoio, qual notação será utilizada, inicia-se, então, a fase de análise do processo em si. Aqui, serão citadas algumas técnicas de análise, bem como o que deve ser focado para que a documentação do processo seja a mais próxima da realidade possível.

Antes de se definir ou atualizar um novo processo, deve-se, primeiro, obter o máximo de conhecimento do **estado atual** do processo (**AS-IS**) e como ele cumpre suas metas e objetivos.

Para tal, cada informação do processo é importante: nada pode ser deixado de fora e tudo deve ser questionado (ABPMP, p. 127, 2013).

Cabe atentar ao fato de que algumas melhorias podem ser encontradas durante a fase atual e elas podem ter um caráter de rápida solução - as chamadas "*quick-wins*" - ou podem requisitar um maior tempo para sua solução (ABPMP, p. 127, 2013). Porém, durante essa fase, o importante é **analisar**, **entender** e **documentar** o processo **AS-IS**, ou seja, como ele é. Tais possibilidades de melhoria devem ser documentadas e analisadas, somente, durante a etapa de transformação de processos ou desenho de estado futuro de processo (ABPMP, p. 135, 2013).

Ao realizar uma boa análise do processo AS-IS, qualquer transformação futura poderá ser comparada com o estado antigo do processo: será possível saber se o novo desenho é melhor ou não do que o antigo. Algumas técnicas de medição de desempenho serão abordadas no tópico 2.1.5 deste atual trabalho.

2.1.3.1. Técnicas de Análise de Processos

A seguir, serão apresentadas algumas técnicas analíticas para a extração de informações de um processo (ABPMP, p. 129, 2013). São, no total, 13 técnicas:

- **Análise de custos**: cada atividade tem um custo. Ao conhecer o custo de cada atividade, o custo do processo pode ser determinado. Tal custo pode ser comparado com o custo de um novo processo o qual deve priorizar o aumento de produtividade e diminuição de custos.
- **Análise de tempo de ciclo**: O tempo é a duração entre a entrada e saída do processo e pode ser estipulado, também, como a soma do tempo de cada atividade.
- **Análise de padrão**: essa análise busca a duplicidade de processos, ou seja, se um processo é executado mais de uma vez (retrabalho). Eliminar duplicidades acarreta uma economia de escala.
- **Análise de causa-raiz**: procura entender a causa de um resultado específico. É um processo complicado pois pode apresentar mais de uma causa e ele inclui a coleta de dados, investigação e diagramação de causa e efeito.
- **Análise de sensibilidade (*what-if*)**: analisa como o processo lida com a variação de alguns fatores. Por exemplo, como o processo funciona caso a quantidade da entrada seja aumentada ou diminuída. Aqui, é possível determinar a variabilidade do processo, ou seja, como a saída responde a mudanças no processo. Esta análise busca o desempenho ideal do processo.
- **Análise de riscos**: procura entender como o processo lidaria com cenários de risco. Para isso, deve-se entender quais são os riscos do processo e qual a probabilidade de eles ocorrerem. O risco de maior probabilidade deve ter uma maior atenção. Além disso, os

planos de contingência devem ser revisados e, caso não existam, devem ser criados a partir desta análise.

- **Análise de layout do local de trabalho**: analisa o fluxo físico do processo, ou seja, os materiais que são movidos, fisicamente, ao longo do processo. Aqui, podem ser encontrados gargalos de movimentos e deslocamentos desnecessários.
- **Análise de alocação de recursos**: essa análise questiona se um recurso disponibilizado se mostra suficiente para a execução do trabalho, além da quantidade desses recursos, se há subutilização ou sobrecarga.
- **Análise de motivação e recompensa**: verifica a existência e a qualidade das motivações e recompensas dos executores de processo. Deve considerar quais motivações e recompensas podem ser inseridas em um novo processo.
- **Análise da qualidade**: essa análise pode ser feita da perspectiva do cliente de processo ou do fornecedor de processo. O foco é em como o processo é executado e na capacidade de execução sem erros e falhas.
- **Análise do valor**: identifica quais atividades agregam valor ou não. Pode ser dividido em três tipos de valor: Adiciona valor ao cliente (influencia a satisfação do cliente), adiciona valor ao negócio (segue políticas e regulamentações) ou não adiciona valor.
- **Análise de conformidade legal**: aqui, deve ser feito o levantamento de requisitos legais e o entendimento da legislação aplicável ao processo de negócio em questão.
- **Análise de redes sociais**: é uma análise do relacionamento entre os atores de processo. Pode ser modelada em diagramas de rede.

2.1.3.2. Documentação da Análise de Processos

A documentação da análise deve ser feita de forma clara e objetiva: deve considerar o entendimento de todos que entrarem em contato com o documento, além de deixar claro o tipo de processo (AS-IS, no caso). O documento deve, também, apresentar os potenciais de melhoria observadas durante a análise (ABPMP, p. 134, 2013).

O guia BPM CBOK 3.0 indica alguns itens importantes que devem estar presentes na documentação da análise de processos (muitos desses itens são resultados da utilização das técnicas analíticas abordadas anteriormente):

- Visão geral do ambiente de negócio atual
- Propósito do processo, motivo de sua existência
- Processos indicando seus subprocessos e suas interações

- Fluxo de trabalho da área funcional de análise, bem como suas atividades e tarefas executadas
- Requisitos de medição de desempenho e como é feita a medição
- Métricas
- Lacunas no desempenho do processo
- Razões e causas para as lacunas de desempenho do processo
- Redundâncias do processo que poderiam ser eliminadas e economias esperadas
- Regras documentadas e não documentadas que controlam o trabalho
- Problemas e seus impactos nos processos
- Aplicações da tecnologia da informação e onde são utilizadas no negócio
- Política de requisitos de auditoria interna
- Oportunidades de melhoria e seus possíveis benefícios
- Riscos e seus impactos no processo

2.1.3.3. Considerações para o sucesso da análise de processos

Para que uma análise de processo funcione da melhor maneira possível, algumas considerações devem ser feitas.

Em primeiro lugar, a **liderança executiva da organização deve ser o principal patrocinador da análise**, além de dar suporte e encorajamento. Porém, para que a liderança entenda a importância da análise, pode ser necessária a demonstração de algumas *quick-wins*, ou seja, melhorias imediatas (ABPMP, p. 135, 2013).

Deve ser levado em consideração, também, a **maturidade em processos de negócio**, ou seja, se já houve iniciativa de análise, desenho e transformação de processos, se há ou não aplicações de práticas de BPM. A depender da maturidade, a dificuldade da análise pode variar de forma inversamente proporcional (ABPMP, p. 135, 2013).

Além disso, devemos considerar: **evitar a paralisia por análise -** a análise de processos é um meio, e não um fim e, por isso, não deve tomar tanto tempo; **foco no cliente**; a análise deve ser **baseada em fatos** (ABPMP, p. 135, 2013).

Por último, mas não menos importante, pode existir uma **resistência potencial**, ou seja, a área funcional analisada pode considerar a análise como uma ameaça, tanto os executores de processo quanto os líderes e gestores. Por isso, a liderança deve prover o suporte necessário além de manter o processo de análise o mais transparente possível (ABPMP, p. 135, 2013).

2.1.4. Desenho de Processos

Desenho de processos é, em si, o ordenamento das atividades em um **fluxo** o qual define, formalmente, os objetivos, a saída, a organização das atividades e as regras necessárias para chegar no resultado requerido. O foco, aqui, é na transformação das operações de negócio de forma a conectar os objetivos organizacionais com a necessidade do cliente (ABPMP, p. 143, 2013).

Na disciplina de BPM, o desenho pode ser aplicado em diversos níveis de processo. No entanto, cabe diferenciar **fluxo de trabalho** de **fluxo de processo**: o primeiro se encontra, exclusivamente, dentro de uma área funcional; já o segundo considera a visão ponta a ponta e interfuncional, ou seja, o processo como um todo, independentemente da quantidade de áreas funcionais envolvidas (ABPMP, p. 143, 2013).

Portanto, antes de se iniciar o desenho, o nível de processo deve ser definido e, a depender do nível, a abordagem, quantidade de esforço, governança, resultados e benefícios se mostrarão diferentes. Mas, antes de se desenhar um processo TO-BE, o AS-IS, resultante da análise e modelagem de processo, deve ser bastante entendido pelas partes interessadas (ABPMP, p. 145, 2013).

Ao considerar o trabalho atual, o qual foca na automatização de atividades e tarefas, o desenho de processos focaria no desenho TO-BE do fluxo de trabalho, dentro da área funcional específica. Porém, o entendimento do fluxo de processo ainda seria necessário, uma vez que a mudança de atividades e tarefas de uma área funcional pode alterar o processo como um todo.

2.1.4.1. Desenho de Processo TO-BE

Uma vez entendido o processo AS-IS, determinado o nível de processo e escolhida a(s) ferramenta(s) de modelagem, deve haver um entendimento sobre o nível de mudança e qual será seu impacto em todos os níveis de processo. Qualquer mudança pode ter impactos ocultos e por essa razão que a equipe de desenho de processos deve assegurar que nenhum dano será causado pela mudança (ABPMP, p. 152, 2013).

Se a forma como um processo é executado muda (de forma a otimizá-la), mas a saída, a qual se mostra como entrada para o processo consecutivo, é a mesma, não existirão impactos negativos ao próximo processo (ABPMP, p. 154, 2013).

A seguir, serão listadas algumas **atividades chave** para o desenho efetivo de processos, segundo o guia BPM CBOK 3.0 (ABPMP, p. 154, 2013), e levando em consideração os objetivos deste trabalho:

- Desenho do novo processo nos diversos níveis de detalhe apropriados;

- Levar em consideração todos os documentos gerados a partir da análise de processos;
- Definição de atividades internas ao novo processo e identificação do fluxo de trabalho e dependências;
- Definição de métricas desejadas ao novo processo;
- Simulação de modelo, teste e aceitação;
- Comparação de análises existentes;
- Criação e execução de um plano de implementação.

Quando existe a possibilidade de mudança, é importante envolver o máximo possível de executores, do processo em questão, para o aproveitamento da experiência e conhecimento deles. Isso assegura o afastamento da aversão a mudança, além de evitar possíveis erros na execução do processo futuro (ABPMP, p. 158, 2013).

Um desenho de processos deve se mostrar simples: de fácil e rápida interpretação, além de possibilitar uma fácil execução. Complexidades desnecessárias aumentam o custo de operação e diminuem o tempo de resposta da organização, o que transforma a mudança em algo de difícil execução (ABPMP, p. 169, 2013).

Um novo processo requer gerenciamento de mudança, e isso será discutido mais a frente em “Transformação de Processos”. Além disso, para o sucesso da transformação, deve haver um gerenciamento de desempenho, e esse será tratado a seguir.

2.1.5. Gerenciamento de Desempenho de Processos

Ao se gerenciar um negócio, medidas, métricas, e indicadores são necessários para a monitoração e ela, por sua vez, é responsável por verificar o alcance ou não de metas e objetivos traçados pela organização. O Gerenciamento de Desempenho de Processos envolve, de forma simultânea, a compreensão de **o que** medir e de **como** medir e, somente com o entendimento de ambas as partes que o gerenciamento se torna efetivo (ABPMP, p. 183, 2013).

Além disso, o gerenciamento pode ser aplicado tanto ao **fluxo de trabalho** quanto ao **fluxo de processo**, porém, ele se define de forma diferente a depender do nível. Em nível de fluxo de trabalho, o foco se mostra entre as atividades: o movimento físico entre elas e os locais onde se encontram os problemas. Já em nível de fluxo de processo, o foco é entre as áreas funcionais: o que é entregue de uma área para outra. Em ambos os níveis de fluxo, as medições devem ser consistentes: tempo, custo, capacidade e qualidade (ABPMP, p. 187, 2013).

A melhor forma de se iniciar um gerenciamento de desempenho de processo é observando a **eficácia** de um processo. Aqui, a eficiência não vale de nada se a eficácia é ruim. Portanto,

tal gerenciamento começa com uma visão sobre os processos que terão, de fato, seus desempenhos monitorados: eles estão de acordo? (ABPMP, p. 188, 2013).

Tudo deve ser questionado, e não deve ser suposto que tudo esteja certo. É importante, portanto, ter alguns questionamentos em mente (ABPMP, p. 188, 2013):

- Questionar o motivo do processo estar no negócio. Ele é exclusivo?
- Questionar em qual mercado o negócio se encontra e qual o desafio desse mercado
- Questionar quem é o cliente do negócio e o que ele procura
- Questionar se o produto ou serviço entregue ao cliente está de acordo com o que ele procura
- Questionar o que as organizações semelhantes fazem melhor
- Questionar se o processo atual apoia os benefícios estratégicos da organização
- Questionar quais problemas devem ser resolvidos primeiro e o que deve ser feito para resolvê-los

Com as respostas desses questionamentos em mãos, é possível ter uma noção dos **processos mais importantes** dos fluxos de processo ou de trabalho em questão, bem como suas atividades e tarefas. Além disso, cabe considerar que nem sempre o que é mais barato é melhor e, sem esse entendimento, não é possível entender a qualidade e os requisitos de processo e a medição de desempenho estaria limitada a fatores elementares de tempo e movimento (ABPMP, p. 188, 2013).

2.1.5.1. Definindo o Desempenho de Processos

Podem existir diversas definições para desempenho de processos, porém, segundo as práticas de BPM, se define por:

> O rendimento de um processo em termos de extrapolação de tempo, custo, capacidade e qualidade (ABPMP, p.191, 2013).

O primeiro passo para a medição de desempenho de um processo é determinar **o que** será medido e o **motivo** da medição, além da identificação de qual valor será usado para comparação. Alguns fatores devem ser determinados: objetivo da medição, item a medir, parâmetro de comparação, onde medir, o que medir, como será medido e quem será o responsável pela medição (geralmente, gestores das áreas funcionais em questão) (ABPMP, p. 192 e 193, 2013).

Segundo as práticas de BPM, nenhum processo deve ser transformado antes de ser medido seu desempenho. Sem essa medição, não existirá fator de comparação para o novo processo. Por isso, deve-se ter medições, monitoramento e controle dos processos para possibilitar o alcance de seus objetivos (ABPMP, p. 194, 2013).

Temos 4 tipos de medições de desempenho (ABPMP, p. 197, 2013):

- **Tempo**: relacionado com a duração do processo. Ex.: tempo de entrega desde a data de solicitação, tempo médio entre falhas, tempo de espera;
- **Custo**: valor, normalmente, monetário. Ex.: custo de produção, custo de logística, custo de mão de obra;
- **Capacidade**: volume ou montante de saídas viáveis de um processo. Ex.: número de atendimentos efetivos por unidade de tempo;
- **Qualidade**: é, normalmente, expressa pela porcentagem real em relação ao ótimo, apesar de poder assumir diversas formas. Ex.: taxa de erros da saída de um processo, confiabilidade do produto ou serviço, experiência do usuário.

Uma medição, ou medida, passa a ser um **indicador** quando é utilizado para facilitar a interpretação de um processo quando comparada a uma referência ou meta. Deve-se ter um cuidado muito grande ao utilizar indicadores, pois caso sejam utilizados de maneira errônea, podem levar a tomadas de decisão equivocadas (ABPMP, p. 200, 2013).

Dentro dos tipos de indicadores, temos 2: os **indicadores direcionadores**, ou *drivers*, e os **indicadores de resultados**, ou *outcome*. O primeiro é responsável por monitorar a causa, antes do efeito, e, com ele, é possível alterar o curso de um processo para o alcance de um resultado. Já o segundo monitora o efeito, em si, e não permitem alterar o resultado (ABPMP, p. 200, 2013).

Dentro de um processo, um indicador que demonstra o desempenho do fluxo de trabalho ou de processo é um **PPI**: *Process Performance Indicador*, ou **Indicador de Desempenho de Processo**. Portanto, para o monitoramento do desempenho de um processo de forma efetiva, também é necessário o uso de PPIs os quais devem ser baseados nos 4 tipos de medições de desempenho, citados anteriormente (ABPMP, p. 201, 2013).

Assim que definido **o que**, deve ser definido **como** será medido: uma medição pode ser, simplesmente, uma tabela manual baseada por uma fórmula. O mais importante dessa medição não é a fórmula em si, e sim a aprovação da utilização dela pelos membros executores do processo a que se deseja medir. Tal tabela pode ser mostrada em painéis de monitoramento que vão servir para a observação do desempenho, o que possibilita o aprofundamento em detalhes que antes não eram vistos (ABPMP, p. 207, 2013).

2.1.6. Transformação de Processos

Pessoas parecem estar à espera de um estado de estabilidade para, então, agirem. Isso é algo que não faz sentido pois a estabilidade nunca é alcançada dentro de organizações. Elas devem, então, aprimorar suas capacidades de mudança e transformação (ABPMP, p. 233, 2013).

Existem duas táticas diferentes quando o assunto é Transformação de Processos: **táticas incrementais** e **táticas radicais**. As táticas incrementais se definem pelas pequenas modificações que não rompem, bruscamente, a forma de adaptação a mudanças de ambiente tradicionais do negócio em questão. Já as táticas radicais buscam romper práticas organizacionais tradicionais em busca de mudanças (ABPMP, p. 233, 2013).

Abaixo são listados alguns motivos para a realização da transformação de processos (ABPMP, p. 233, 2013):

- Construir processos com foco no cliente
- Aumentar a produtividade
- Reduzir defeitos
- Reduzir desperdícios
- Garantir a sustentabilidade de operações
- Reduzir o tempo de ciclo de processos
- Melhorar a qualidade de processos
- Aumentar a capacidade
- Aproveitar e desenvolver oportunidades
- Inovar
- Mudar paradigma
- Reduzir risco e custo

Muitas organizações não possuem maturidade em BPM o suficiente para o início na transformação de processos como primeiro passo. Portanto, é necessário realizar, antes, a modelagem dos processos em alto nível e a identificação das áreas funcionais envolvidas na mudança. Porém, caso já exista o entendimento do modelo de processo, cabe, somente, sua revisão e atualização (ABPMP, p. 234, 2013).

Durante a análise de processo, algumas possibilidades de melhorias são apontadas, como visto anteriormente. Aqui, na transformação de processos, cabe o estudo em cima dessas possibilidades (ABPMP, p. 234, 2013).

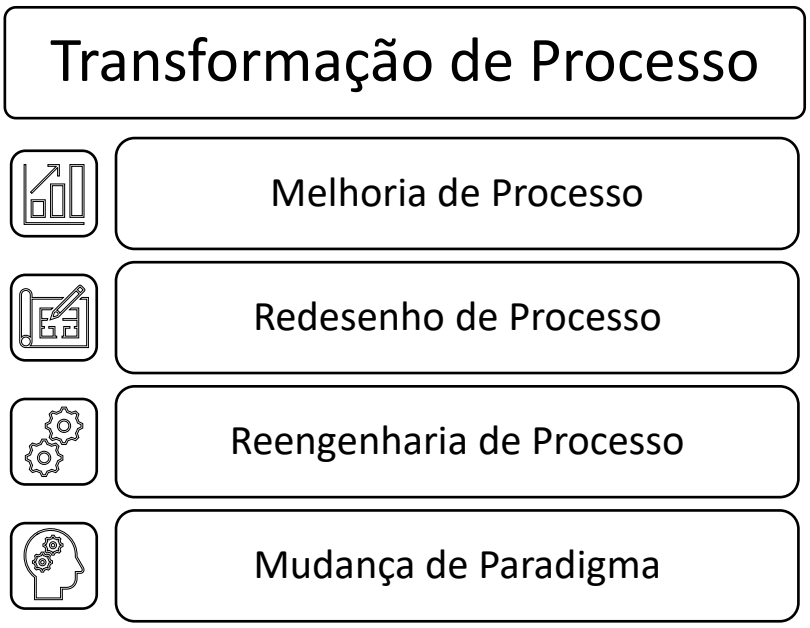
A transformação de processos, tem, portanto, o objetivo de encontrar a melhor maneira de se realizar um processo específico, seja através da aquisição de um equipamento novo, novas abordagens, ou de uma nova aplicação (ABPMP, p. 234, 2013).

2.1.6.1. Amplitudes de Transformação de Processo

A transformação de um processo pode ser aplicada em qualquer nível de processo, seja ele de ponta a ponta ou dentro de uma área funcional específica (ABPMP, p. 234, 2013), além

disso, ela pode apresentar amplitudes de transformação diferentes, tais como: melhoria de processo, redesenho de processo, reengenharia de processo ou mudança de paradigma (Figura 11).

Figura 11 - Amplitudes de Transformação de Processos



Fonte: autoria própria

A **melhoria de processos** de negócio, ou BPI (*Business Process Improvement*) implica em uma mudança ou alteração específica em um processo, atividade ou tarefa baseada em um projeto de mudança específico com o foco em um melhor alinhamento ou desempenho de processo. Existem algumas abordagens, amplamente utilizadas, para a melhoria contínua, tais como o Lean, Seis Sigma e TQM (ABPMP, p. 236, 2013). Porém, a melhoria de processos não significa que uma empresa esteja comprometida com as práticas de BPM. [BPM CBOK 3.0, p. 234]

O **redesenho de processos** é o repensar, ponta a ponta, de um processo o qual está sendo realizado atualmente. É diferente da melhoria de processos pois envolve uma visão holística ao invés de identificar e implementar mudanças incrementais (ABPMP, p. 240, 2013).

A **reengenharia de processos** é o redesenho radical e um repensar fundamental de processos em busca de melhorias dramáticas em um negócio. Ela inclui as áreas funcionais que participam do processo, ou seja, é interfuncional e pode ser aplicada em qualquer nível de negócio (ABPMP, p. 240, 2013).

Já a **mudança de paradigma** é uma dedicação à inovação, é arriscar uma adoção de uma abordagem que permita a criação, difusão e incorporação do conhecimento a novos processos e utilizá-la como vantagem competitiva. Ideias não são criadas, são descobertas, portanto,

quando um ambiente possibilita tal abordagem, inovações vêm à tona e alavancam os processos de um negócio (ABPMP, p. 242, 2013).

Melhoria é, por definição, tornar algo (que já se tem) melhor. Ou seja, se o foco é tornar um processo mais rápido, com maior eficiência, é melhoria, somente. Mas é importante ter em mente que não adianta buscar melhorar aquilo que nem deveria existir. Nesse quesito, o guia BPM CBOK 3.0 trata a melhoria como algo importante, mas não suficiente para o aumento significativo do desempenho de um processo, colocando a mudança de paradigma, redesenho e a reengenharia como mais impactantes (ABPMP, p. 244, 2013).

Muitas empresas, ao focarem em uma mudança tecnológica com a aplicação de um novo software, por exemplo, deixam de lado os processos de negócio e, ao fazerem isso, não conseguem utilizar o maior potencial da modernização tecnológica em questão. Trabalhos manuais desnecessários, retrabalho e excesso de esforço em processos que não agregam real valor ao cliente continuarão existindo (ABPMP, p. 248, 2013).

2.1.6.2. Visão necessária para a Transformação de Processos

A transformação de um processo impactará, diretamente, o trabalho de diversas pessoas, mas os mais impactados serão os chamados **executores de processo**. Os executores, principalmente, devem estar cientes e de acordo com a mudança. Além disso, o desempenho do novo processo deve ser estipulado, bem como a estrutura organizacional necessária para a governança, novos conhecimentos e habilidades de equipe necessários e o uso de ferramentas apropriadas de produtividade (ABPMP, p. 249, 2013).

A gerência deve estar de acordo e deve ajudar a disseminar a visão de transformação de processos como forma de apoio e incentivo. Além disso, problemas políticos podem surgir, portanto o patrocinador da transformação de processos deve ter autoridade o suficiente para combater tais conflitos (ABPMP, p. 251, 2013).

2.1.6.3. Gerenciando a mudança

Para que o processo de transformação seja efetivo, deve haver um gerenciamento para suportar a mudança e deve ser levado em consideração desde o início de qualquer iniciativa de transformação (ABPMP, p. 254, 2013). Por definição:

> Gerenciamento de mudança é um processo iterativo que utiliza um conjunto de técnicas para auxiliar uma organização e seus colaboradores na transição de um estado atual para um estado futuro sustentável. Promove o alinhamento na organização em momentos de mudança, provê condições para obtenção de capacidades e conhecimentos necessários, foca objetivos certos, prepara a organização para a mudança e motiva os colaboradores a alcançar resultados sustentáveis. (ABPMP, p. 255, 2013)

Caso a equipe responsável pelo novo processo não esteja de acordo com a mudança, ela passará a ter desconfiança, ressentimento e, muitas vezes, resistência ativa contra a transformação. Portanto, para evitar tal acontecimento, **a equipe deve estar envolvida no processo de mudança**. Os movimentos de oposição à mudança podem ocorrer devido (ABPMP, p. 267, 2013):

- O processo não estar alinhado com os sistemas de avaliação de desempenho e recompensa;
- O processo não ser adequado ao nível atual de conhecimento e habilidade de equipe;
- O processo não estar alinhado com prioridades de mudança;
- Interesses políticos contrários.

Além dessas possibilidades de oposição, a resistência a mudança também pode ser associada a (ABPMP, p. 264, 2013).

- Possível perda de controle e poder sobre o processo;
- Sobrecarga com as responsabilidades já existentes;
- Falta de entendimento da necessidade de mudança;
- Incerteza sobre possuir as capacidades necessárias para o novo processo;
- Dúvida, medo, falta de confiança;
- Conforto com o processo atual;
- Medo da nova forma demandar mais trabalho e esforço do que a forma atual.

Portanto, para que essas oposições acima não ocorram, deve existir um **plano de comunicação** visando a transparência dos estágios de transformação. Pode existir a necessidade de envolver a Gestão de Pessoas da organização para facilitar tal transparência (ABPMP, p. 268, 2013).

Investir tempo no gerenciamento da transformação de processos, priorizando a transparência e alinhamento, **aumenta a probabilidade de sucesso da mudança**, além de acelerar a adoção e reduzir perdas de produtividade (ABPMP, p. 271, 2013).

2.1.6.4. Transformando um processo

Uma transformação de processo só pode ser realizada após o entendimento por completo do processo AS-IS, como falado anteriormente na sessão de Análise de Processo. Cada atividade e tarefa deve ser revisada, indicadores de desempenho devem ser considerados e uma visão sistêmica é indispensável (ABPMP, p. 272, 2013).

Antes de se desenhar o novo processo, deve ser considerado, também, algumas possíveis restrições: recursos financeiros, tecnologia da informação e restrições legais. O pensamento

fora da caixa da transformação de processos deve estar alinhado com as possíveis restrições citadas (ABPMP, p. 274, 2013).

Resumindo, o novo processo deve considerar (ABPMP, p. 278, 2013).

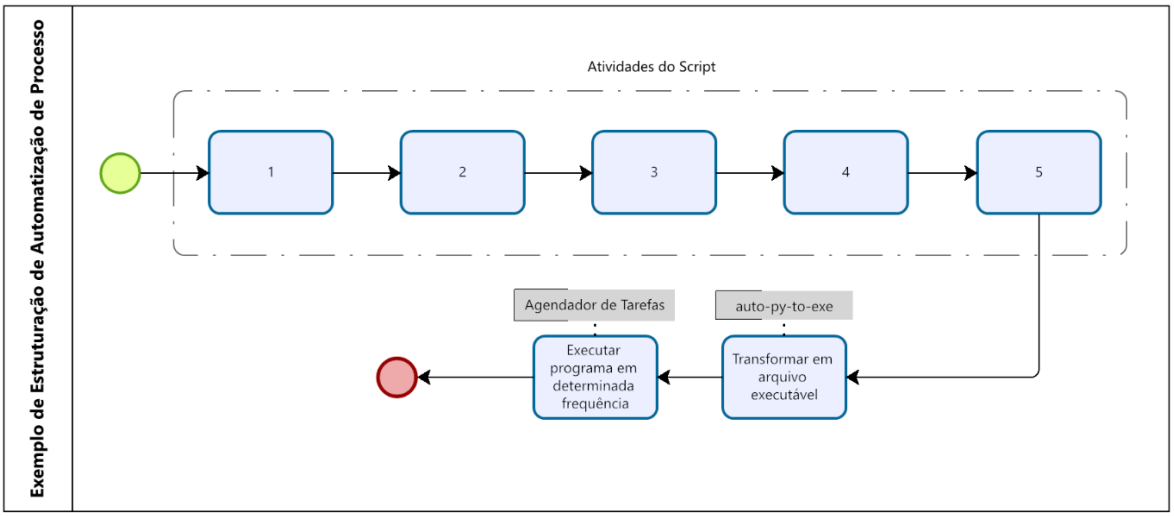
- Revisão do processo AS-IS;
- Foco no cliente;
- Entendimento da estrutura organizacional;
- Descrição das novas habilidades;
- Alinhamento com as partes interessadas;
- Gerenciamento de desempenho do novo processo;

2.2. Python e a Automatização de Atividades e Tarefas

Neste tópico, serão apresentadas algumas maneiras de se executar atividades e tarefas específicas utilizando a linguagem de programação Python. Após se realizar um script em Python com a atividade em questão, é possível gerar um arquivo executável da atividade (auto-py-to-exe – Tópico 2.2.6) e programá-la para ser executada com frequências diferentes (Agendador de Tarefas – Tópico 2.2.7). Em outras palavras, é possível **automatizar atividades e tarefas com Python**.

Cada um dos tópicos apresentados pode ser visto como atividades e tarefas isoladas e **podem ser colocadas em uma sequência desejada para executar um processo específico**. Podemos verificar isso na Figura X, onde as atividades 1, 2, 3, 4 e 5 podem se apresentar de maneiras e quantidades diferentes.

Figura 12 - Exemplo de Estruturação de Automatização de Processo



Fonte: Autoria própria

As atividades 1, 2, 3, 4 e 5 podem ser as seguintes[1]:

- Manipulação e análise de dados com pandas;
- Manipulação de pastas e arquivos do computador com pathlib e shutil;
- Envio de e-mails com yagmail;
- Manipulação de Bancos de Dados com pyodbc;
- Manipulação de sites da internet e aquisição de dados;

[1] Aqui são citadas somente algumas das possibilidades de execução de tarefas com *python.* No entanto, existem outras possíveis aplicações

- A abordagem de cada um desses tópicos se mostrará simples, com o intuito de um entendimento básico sobre algumas possibilidades dentro de cada umas das bibliotecas mencionadas. Cabe ao leitor, portanto, explorar as referências bibliográficas para um melhor aprofundamento e um maior poder de aplicação dessas ferramentas.

2.2.1. Manipulação e Análise de Dados com "*pandas"*

A utilização de Pandas, uma biblioteca do Python, nos permite manipular planilhas por meio de linhas de código de maneira rápida e eficaz (PANDAS DOCUMENTATION, 2022). Além disso, funções e métodos dentro dessa biblioteca nos permitem importar arquivos csv, xlsx (também suporta xls, xlsx, xlsm, xlsb, odf, ods and odt), os quais são mais, amplamente, utilizados em planilhas.
Veja o exemplo 1, abaixo:

```
1. import pandas as pd
2. df = pd.read_csv('Nome_do_arquivo.csv', sep = ';')
```

Antes de utilizarmos uma biblioteca, devemos, primeiro, "chamá-la" em nosso código. Para tal, como podemos ver na linha 1, utilizamos a palavra *"import"* seguido a biblioteca de interesse. Note que a palavra "as" foi utilizada para vincular o nome *"pandas"* a *"pd"* como estratégia de simplificação para sua chamada futura (como podemos ver em ***pd***.*read_csv* ao invés de ***pandas***.*read_csv*).

Na linha 2, estamos atribuindo a leitura do arquivo "Nome_do_arquivo.csv", ou seja, o DataFrame (tabela em Python, basicamente), à variável "df" e, nesse caso, o arquivo csv é do tipo separado por ponto-e-vírgula (*sep = ';'*).

Note que podemos utilizar outros tipos de importação, um para cada tipo de arquivo: *pandas.read_excel, pandas.read_xml, pandas.read_json*, entre outras possibilidades (PANDAS DOCUMENTATION, 2022). Cabe lembrar, também, o arquivo deve estar salvo na mesma pasta do programa que está sendo rodado ou, deve-se passar o local do arquivo entre parêntesis em seu lugar (exemplo: 'C:/lucas/tcc/Nome_do_arquivo.csv').

Uma vez que importamos uma tabela de um arquivo excel para dentro de um programa *Python*, podemos manipular a tabela, agora denominada de DataFrame (comumente chamada de "df"), das seguintes formas:

- ***df['Coluna']*:** é uma lista da coluna "Coluna" do DataFrame "df", em formato de um DataFrame de uma coluna só. Podemos selecionar mais de uma coluna: df[['Coluna1',

‘Coluna2’, ‘Coluna3’]]: Isso nos daria um DataFrame de 3 colunas: Coluna1, Coluna2 e Coluna3;

- ***df[‘Coluna’][0]*:** seleciona o primeiro item da coluna ‘Coluna’
- ***df[:X]*:** seleciona até a linha de índice X (número inteiro) do DataFrame “df”.

Utilizando dessas manipulações, podemos aplicar algumas funções para modificar a estrutura de um DataFrame, aplicar fórmulas, filtrar dados requeridos, entre muitas outras possibilidades. A seguir, temos algumas possibilidades:

2.2.1.1. Eliminar linhas ou colunas, respectivamente:

- *DataFrame.drop([[“Linha1”, “Linha2”, “Linha3”]], axis = 0)*
- *DataFrame.drop([[“Coluna1”, “Coluna2”, “Coluna3”]], axis = 1)*

2.2.1.2. Mesclar DataFrames

- *DataFrame1.merge(DataFrame2, on = ‘ColunaComum’)*. 1 e 2 tendo uma coluna em comum: “ColunaComum”.

2.2.1.3. Renomear Colunas:

- *DataFrame.rename(columns = {“NomeAntigo”: “NomeNovo”})*. Também é possível mudar o nome de múltiplas colunas passando “{“NomeAntigo1”: “NomeNovo1”, “NomeAntigo2”: “NomeNovo2”}”.

2.2.1.4. Obter a frequência de repetição de itens em uma coluna:

- *DataFrame[“Coluna”].value_counts()*.

2.2.1.5. Plotar gráficos:

- *DataFrame.plot()*. Onde é possível:
  - manipular o tamanho da figura com *DataFrame.plot(figsize = (x, y))*, onde x e y são números inteiros;
  - manipular o nome dos eixos do gráfico: *xlabel = ’nome_x’, ylabel = ‘nome_y’*;
  - estipular limites dos eixos x e y*: xticks(x1, x2, intervalo)* ou *yticks(y1, y2, intervalo)*, onde x1 e y1 são os limites inferiores; x2 e y2 são os limites superiores e “intervalo” é o intervalo entre cada dado;
  - Escolher o tipo de gráfico: *DataFrame.plot(kind = “pie”)*. Nesse caso, apresenta-se um gráfico de pizza. A lista de possibilidades se encontra na documentação pandas.

2.2.1.6. Somar dados de uma coluna, obter o máximo e mínimo:

- *DataFrame["Coluna"].sum()*;
- *DataFrame["Coluna"].max()*;
- *DataFrame["Coluna"].min()*.

2.2.1.7. Agrupar dados:

- *DataFrame.groupby("Coluna")*.

2.2.1.8. Exportar DataFrame para csv ou xlsx:

- *DataFrame.to_csv("Nome_do_Arquivo.csv", sep = ";")*;
- *DataFrame.to_excel("Nome_do_Arquivo.xlsx").*

2.2.2. Manipulação de Arquivos e Pastas do Computador com "*pathlib"*

Podemos acessar pastas e arquivos do computador utilizando a biblioteca "*pathlib*" (PATHLIB, 2022). Abaixo, será listado as possibilidades de se utilizar a biblioteca:

2.2.2.1. Importando a biblioteca

```
1. from pathlib import Path
```

2.2.2.2. Descobrindo o local do arquivo

```
1. caminho = Path.cwd()
```

2.2.2.3. Navegando até uma pasta específica

```
1. caminho = Path('pasta1/pasta2/pasta3/pasta4/')
```

2.2.2.4. Listando todos os arquivos da pasta atual

```
1. arquivos = Path.iterdir()
2. for arquivo in arquivos:
3. |    print(arquivo)
```

2.2.2.5. Criando uma nova pasta

```
1. Path('pasta/nova_pasta').mkdir()
```

2.2.2.6. Verificando a existência de um arquivo

```
1. arquivo = Path('pasta/arquivo.extensao_do_arquivo')
2. arquivo.exists()
```

2.2.2.7. Copiando um arquivo

```
1. import shutil
2. shutil.copy2('arquivo_que_quer_copiar.extensao',
   'nome_da_copia_criada.extensao')
```

2.2.2.8. Movendo um arquivo

```
1. import shutil
2. shutil.move(Path('caminho/arquivo.extensao'),
   Path('caminho_novo/arquivo.extensao'))
```

2.2.3. Envio de E-mail com "*smtplib"*

É possível enviar e-mails com o *python* utilizando a biblioteca "*smtplib*" (SMTPLIB, 2022). Seguem os passos:

2.2.3.1. Importando a biblioteca

```
1. import smtplib
```

2.2.3.2. Atribuição de Assunto, remetente, destinatário e corpo de e-mail

```
1. Msg = EmailMessage()
2. Msg['Subject'] = "Assunto do e-mail"
3. Msg['From'] = "email_remetente@email.com"
4. Msg['To'] = "email_destinatário@email.com"
5. Msg.add_header('Content-Type', 'text/html')
6. Msg.set_payload('corpo_do_e-mail')
```

2.2.3.3. Anexando arquivos no e-mail

```
1. Caminho_arquivo = "C:/usuário/pasta1/pasta2/arquivo.extensao"
2. Mime_type, encoding = mimetypes.guess_type(Caminho_arquivo)
3. With open(Caminho_arquivo, 'rb') as fp:
4.         Dados = fp.read()
5. Msg.add_attachment(Dados, maintype=Mime_type.split("/")[0],
   subtype=Mime_type.split("/")[1], filename= "nome_arquivo.extensao")
```

2.2.4. Manipulação de Banco de Dados com "*pyodbc"*

A biblioteca "pyodbc" permite a manipulação de banco de dados (PYODBC WIKI, 2022). Aqui, é feita a importação de uma tabela, com pandas, para facilitar o entendimento.

2.2.4.1. Importando a biblioteca

```
1. import pyodbc
```

2.2.4.2. Estabelecendo dados de conexão com o Banco de Dados

```
1. dados_de_conexao = ("Driver = {nome_do_driver};" "Server=nome_do_servidor;"
   "Database=nome_da_base_de_dados;" "UID=Login;" "PWD=Senha;")
2. conexao = connect(dados_de_conexao)
```

2.2.4.3. Utilizando o cursor para a interação com o banco de dados

```
1. cursor = conexao.cursor()
```

2.2.4.4. Lendo o banco de dados com Pandas

```
1. import pandas as pd
2. df = pd.read_sql('SELECT * FROM BaseDeDados.Tabela', conexao)
```

A partir de então, volta ao tópico de análise de dados com pandas (PANDAS DOCUMENTATION, 2022)

2.2.5. Manipulação de Sites da Internet com *"selenium"* (*Webscraping*)

O foco, aqui, é automatizar alguma atividade ou tarefa feita em algum site de um navegador utilizando a biblioteca *selenium* (MUTHUKADAN, 2022). No caso, o navegador utilizado será o Google Chrome. Mas, para isso, deve ser baixado o driver do google chrome, o ChromeDriver, disponível para download a depender da versão do Google Chrome utilizada (CHROMEDRIVER, 2022). O ChromeDriver deve estar presente na mesma pasta que contém o script em python para que o programa funcione.

Abaixo, serão mostradas as possibilidades da utilização do selenium:

2.2.5.1. Importando a biblioteca

```
1. from selenium import webdriver
```

2.2.5.2. Abrindo o Google no Google Chrome

```
1. driver = webdriver.Chrome()
2. driver.get("http://www.google.com")
```

Ao entrarmos em um site qualquer na internet, é possível localizar elementos utilizando o método *find_element* do Selenium (MUTHUKADAN, Locating Elements, 2022). Estes são elementos em html, uma linguagem de programação utilizada para construir o *frontend* de um site.

A biblioteca possibilita localizar os elementos:

- *ID*;
- *Name*;
- Xpath;
- *Link text*;
- *Partial link text*;
- *Tag Name*;
- *Class Name*;
- *Css Selector*.

Após localizar um elemento em um site (utilizando a função "*.find_element(By.)"*), podemos realizar diversas interações com ele. As possibilidades são (considerando que o elemento foi armazenado na variável "elemento"):

2.2.5.3. Clicar em um elemento

```
1. elemento.click()
```

2.2.5.4. Digitar algo no campo do elemento

```
1. from selenium.webdriver.common.keys import Keys
2. elemento.send_keys("texto")
```

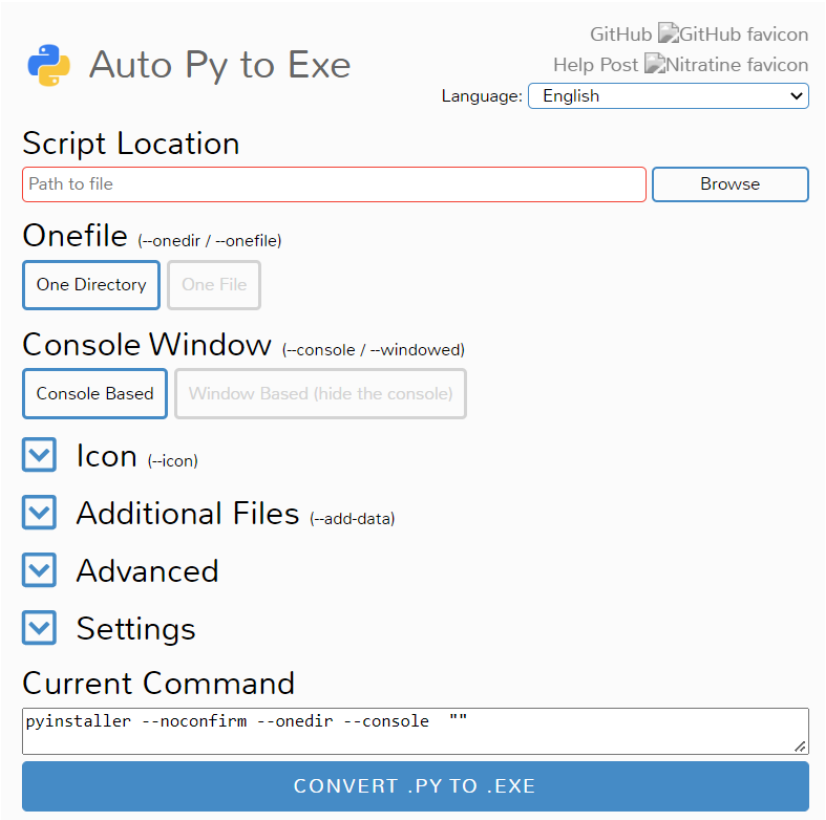
2.2.5.5. Dar "Enter"

```
1 from selenium.webdriver.common.keys import Keys
2 elemento.send_keys(Keys.RETURN)
```

2.2.6. Transformando o script python em arquivo executável

Existem diversas formas de se transformar um arquivo “.py” em “.exe”, porém a forma de se fazer isso, aqui, será utilizada a biblioteca “auto-py-to-exe” a qual se mostra ser uma maneira bem intuitiva e direta (AUTO-PY-TO-EXE, 2022).

Figura 13 - Janela do app "Auto Py to Exe"



Fonte: print de tela da janela do app “Auto-Py-To-Exe”

Após sua instalação, é possível executar o programa e a janela que abre é a representada na Figura 12.

No campo “Script Location”, deve ser inserido o arquivo “.py” requerido para transformação e, ao seguir o passo a passo em https://pypi.org/project/auto-py-to-exe/, o programa executável será criado com sucesso.

2.2.7. Utilizando o Agendador de Tarefas do Windows para executar o programa com frequências específicas

O programa “Agendador de Tarefas” do Windows (MICROSOFT, 2022), pode ser utilizado para inicializar um programa específico em uma data e hora específica, com frequências desejadas. Ou seja, caso o programa “.exe” criado seja algo que deve ser feito todo dia, toda semana ou todo mês, o Agendador de Tarefas pode ser programado para executá-lo na frequência especificada.

Ao iniciar o programa, basta clicar em “Criar Tarefa Básica”, definir um disparador (diariamente, semanalmente, mensalmente, a depender da frequência desejada) e especificar hora e o número de repetições desejadas. Após isso, basta escolher a ação, no caso “Iniciar um Programa”, inserir o programa “.exe” desejado e concluir o processo.

Após isso, a tarefa estará listada na biblioteca do Agendador de Tarefas e pode ser excluída ou editada a qualquer momento.

3 METODOLOGIA

Uma pesquisa científica pode ser classificada quanto a quatro pontos de vista diferentes: sua natureza, seus objetivos, seus procedimentos e sua abordagem (PRODANOV et al, 2013) Com isso, a atual pesquisa contém uma **natureza aplicada** com **objetivos exploratórios, procedimentos baseados em pesquisas bibliográficas e documentais** e considera uma **abordagem qualitativa**

A natureza (aplicada) foi determinada desta maneira, uma vez que o trabalho demonstra possíveis aplicações práticas e busca soluções específicas: automatização de processos, estudados e estruturados com as práticas de BPM, com a linguagem de programação *python.*

Ao se considerar os objetivos do trabalho (exploratórios), o propósito exploratório se mostra aparente, uma vez que busca o aprimoramento de ideias, maior familiaridade com o problema e a possível formulação de hipóteses (GIL, p. 41, 2002).

Os procedimentos foram baseados na pesquisa bibliográfica do livro de referência informativa BPM CBOK 3.0, ao se falar da área de Gestão de Processos de Negócio, e em pesquisas documentais e bibliográficas nas comunidades *python* e suas bibliotecas, ao se falar de Automatização de Atividades e Tarefas, além dos conhecimentos adquiridos pelo leitor no curso de Python da Hashtag Treinamentos (CURSO DE PYTHON ONLINE, 2022).

Com relação à sua abordagem (qualitativa), o foco deste trabalho se dá na melhoria da qualidade de um processo através de sua automatização, além de não utilizar métodos estatísticos para formulação de uma hipótese. Agora, a aplicação das ideias aqui formuladas pode gerar a necessidade de uma abordagem quantitativa, uma vez que para se entender a melhoria de um processo, deve-se mensurá-lo, como comentado anteriormente.

3.1. Materiais

- Guia BPM CBOK 3.0;
- Bibliotecas Python:
  - Pandas;
  - Pyodbc;
  - Selenium;
  - Pathlib;
  - Smtplib
- Programa “Auto-Py-To-Exe”;
- Programa “Agendador de Tarefas” do Windows;
- Google Scholar;

- Visual Studio Code.

3.2. Métodos

Como citado na justificativa, a escolha do estudo do guia BPM CBOK 3.0 foi devido a utilização das práticas de BPM no estágio realizado pelo autor na área funcional de Gestão de Processos do Hospital UMC. Uma vez que a utilização dessas práticas se mostrou eficaz, ela foi utilizada para desenvolver a ideia de analisar, modelar e transformar processos visando a automatização.

No guia BPM CBOK 3.0, foi feita a leitura dos capítulos: Guia para o BPM CBOK 3.0, Gerenciamento de Processos de Negócio, Modelagem de Processos, Análise de Processos, Desenhos de Processos, Gerenciamento de Desempenho de Processos e Transformação de Processos (totalizando 290 páginas). Durante a leitura, foram feitas anotações em um documento ".txt" e após sua conclusão, tal documento foi impresso e usado de base para a formulação da primeira parte do referencial teórico (BPM).

A segunda parte do referencial teórico foi baseada no conhecimento adquirido pelo autor no curso "Python Impressionador" (CURSO DE PYTHON ONLINE, 2022), da empresa "Hashtag Treinamentos", além da documentação de cada uma das bibliotecas: Pandas, Pyodbc, Pathlib, Selenium e Smtplib.

Com o conhecimento de BPM e Python em mãos, levando em consideração a importância de se entender um processo antes de transformá-lo (BPM), foi criado, então, o exemplo de aplicação deste trabalho. Para isso, as funcionalidades do script foram determinadas:

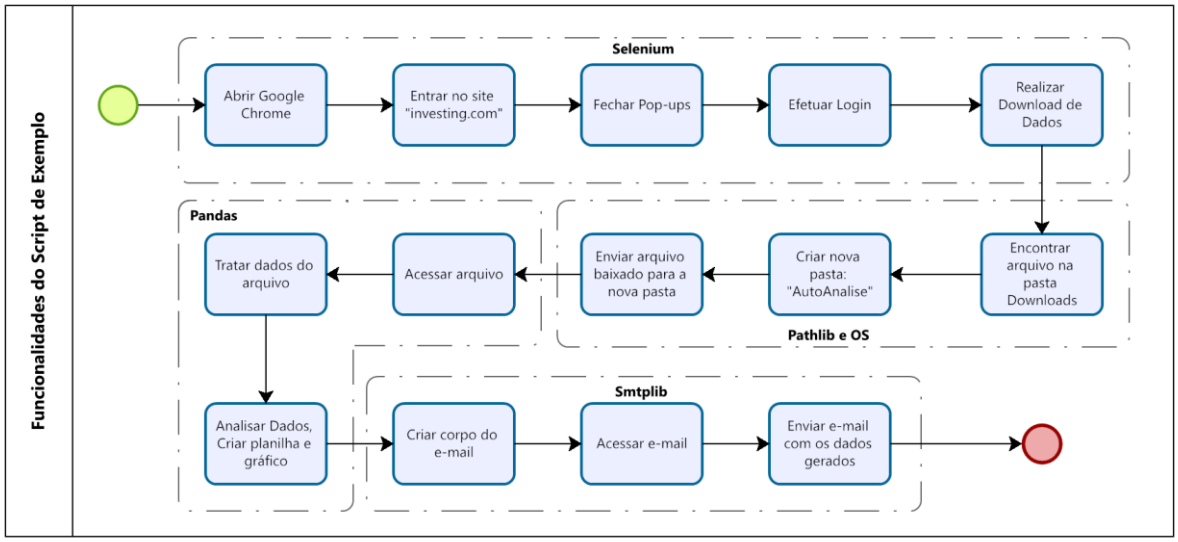
- Entrar no site "https://br.investing.com/currencies/eur-brl-historical-data", o qual demonstra dados históricos do preço do Euro, em Reais;
- Efetuar login[2] no site;
- Realizar o download da planilha de dados (EUR no último mês);
- Tratar dados, formular planilha e gráfico;
- Enviar e-mail contendo a planilha formulada.

Como forma de documentação do novo processo, foi criado o fluxograma com a notação BPMN descrevendo todas as funcionalidades do script (Figura 14). As atividades e tarefas estão organizadas em sequência e estão circuladas por linhas tracejadas as quais representam quais bibliotecas *python* serão responsáveis pelo grupo de atividades ali descritas. Por exemplo: as

[2] Essa atividade pode ser encontrada em diversos sites diferentes, e o procedimento para tal será muito parecido em todos esses sites. Isso abre portas para outros tipos de aplicação.

atividades “Abrir Google Chrome → Entrar no site “investing.com” → Fechar Pop-ups → Efetuar Login → Realizar Download de Dados” são realizadas pela biblioteca “Selenium” do *python*.

Figura 14 - Fluxograma das Funcionalidades do Script de Exemplo



Fonte: Autoria própria

A partir desse fluxograma (Figura 14) o script “.py” foi sendo criado. Para facilitar entendimento e organização, o código, encontrado no APÊNDICE A deste documento, foi separado em 4 partes:

- Selenium: responsável por entrar no site e realizar o download de dados;
- Pathlib: responsável por manipular o arquivo baixado dentro do computador;
- Pandas: responsável por tratar e analisar os dados do arquivo baixado;
- Smtplib: responsável por enviar o e-mail com as análises feitas.

Após a formulação do script “.py”, seguiu-se o processo de transformação deste arquivo “.py” em um arquivo “.exe”, como discutido na Figura 12. Para tal, utiliza-se o programa “auto-py-to-exe”, discutido em 2.2.6. Uma vez com o programa executável em mãos, foi utilizado o Agendador de Tarefas do Windows para executar o programa em uma frequência desejada: de mês em mês.

A seguir, será explicado como o código foi criado, parte a parte.

3.2.1. Entrar no site, realizar login e obter dados com Selenium

O primeiro passo do processo é acessar o site, mas, para isso, deve-se primeiro abrir o navegador Google Chrome. Será atribuída a variável “driver” o Google Chrome em si e toda vez que ela for chamada, alguma execução será feita dentro do navegador.

Na Figura 15, temos a abertura do navegador e “time.sleep(5)” é o comando atribuído para a espera de 5 segundos antes de o código continuar. Em “service”, dentro de “Chrome()”, é atribuída “ChromeDriverManager().install()” para que o código funcione a partir da versão do driver Google Chrome instalada pelo usuário, instalando-se, então, a versão adequada para sua execução.

Figura 15 - Script python responsável por abrir o Google Chrome

```
# Abre o Chrome e certifica o download do chromedriver de versão adequada
driver = webdriver.Chrome(service = Service(ChromeDriverManager().install()))
time.sleep(5)
```

Fonte: Autoria Própria

Como comentado no referencial teórico, em 2.2.5.2., “driver.get()” é responsável por abrir uma página do navegador. Para isso, então, é executado: driver.get(“Site da Investing”) na Figura 16.

Figura 16 - Script python responsável por abrir o site da Investing.com

```
# Abre o site da Investing
driver.get("https://br.investing.com/currencies/eur-brl-historical-data")
```

Fonte: Autoria Própria

Após a abertura do site, é utilizada a função “find_element()” para localizar o elemento dentro do site que precisamos. Como comentado em 2.2.5.3, podemos localizar um elemento pelo seu ID, Nome da Classe, Nome da Tag, entre outros. Na Figura 17, é utilizada “find_elements(By,ID, “ID_do_elemento”)”.

Para encontrar um atributo de um elemento em um site, podemos, dentro do site, pressionar o botão “F12” do teclado para inspecionar seus elementos (Figura 18). Analisando a Figura 18: dentro do **quadrado azul**, temos a área onde se apresentam os **elementos**, em html, do site em questão; O **losango amarelo** é um **botão** permite, ao ser clicado, **localizar um elemento** com o ponteiro do mouse; no caso desta figura, o elemento inspecionado é a figura “Investing.com” dentro do quadrado amarelo; Como resultado, o **círculo amarelo** nos apresenta os **elementos atribuídos à figura** e é possível perceber a **classe** deste elemento (“**logo_logo-svg__1XpS**”), **sublinhada em azul** dentro do **círculo amarelo**.

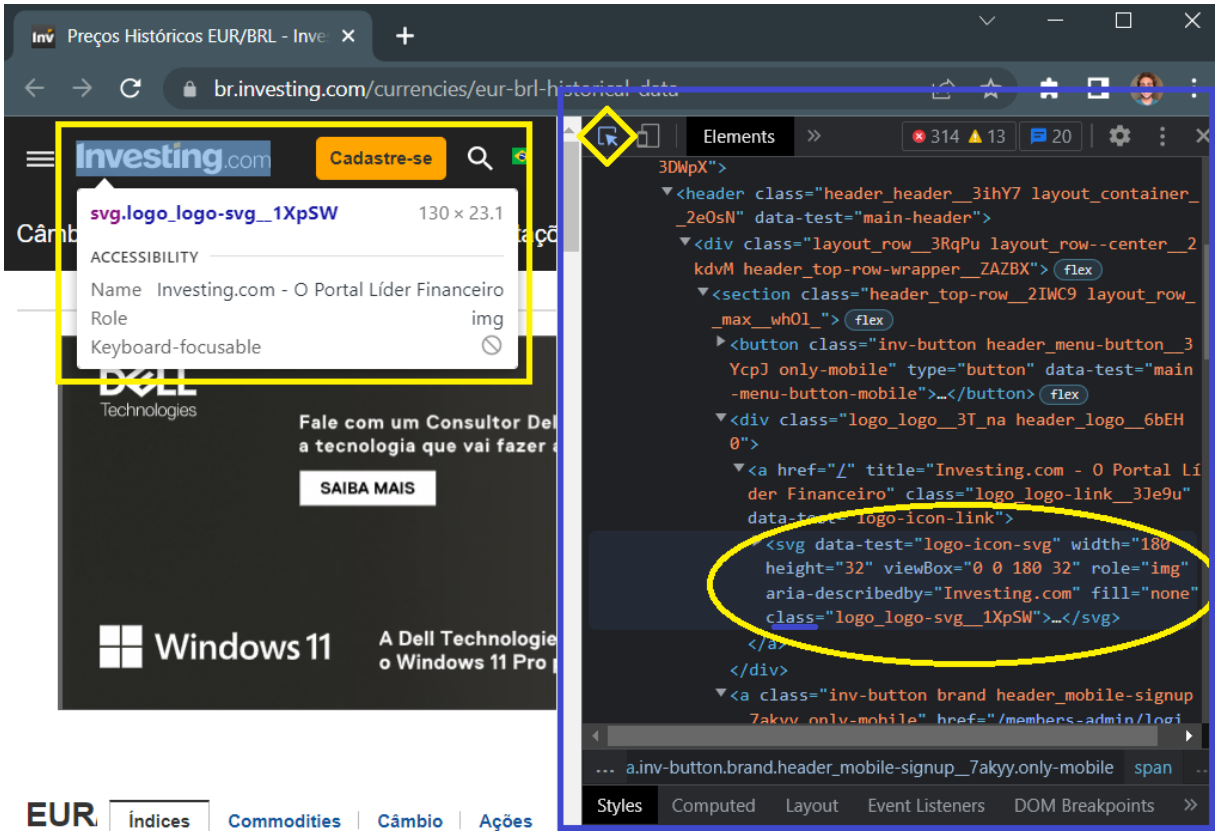
Portanto, esse é o processo utilizado para localizar um elemento dentro de uma página, seja um ID, uma classe, uma tag, entre outros, e ele se encontra ao longo de todo o código. Assim, na última linha do código python da Figura 17, é utilizado o ID (onetrust-accept-btn-handler) para localizar o botão “Aceito” do pop-up que se manifesta ao abrir o site e, em seguida, é colocado “.click()” para que o botão seja clicado.

Figura 17 - Script python responsável por clicar no botão "Aceito" do Pop-up do site

```
# Clica no botão "aceito" do pop-up
while len(driver.find_elements(By.ID, "onetrust-accept-btn-handler")) == 0:
    time.sleep(1)
time.sleep(1)
driver.find_element(By.ID, "onetrust-accept-btn-handler").click()
```

Fonte: Autoria Própria

Figura 18 – Exemplo do processo de inspeção de elementos dentro de um site



Fonte: adaptação de https://br.investing.com/currencies/eur-brl-historical-data

Ainda na Figura 17, a função “find_elements()”, diferente da função “find_element()” no singular, retorna uma lista de elementos com o nome que desejamos buscar. Nesse caso, ela

é utilizada como estratégia para aumentar a velocidade do programa (para isso que é utilizada dentro do laço “while”): enquanto o tamanho (“len()”) da lista “driver.find_elements(By.ID, “onetrust-accept-btn-handler”)” for igual a zero (== 0), ou seja, enquanto não existir nenhum ID com o nome “onetrust-accept-btn-handler” no site, aguarde 1 segundo. Aqui, basicamente, o laço espera que o site carregue tal elemento[3].

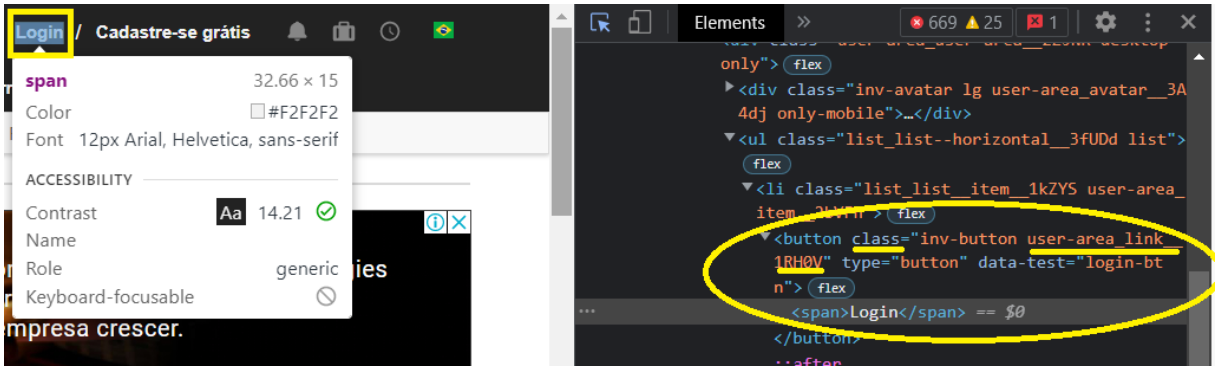
Dessa mesma forma, o script clica no botão de Login do site (Figuras 19 e 20), agora localizando o elemento pelo nome de sua classe[4].

Figura 19 - Código python responsável por clicar no botão "Login"

```
# Clica no botão "Login"
while len(driver.find_elements(By.CLASS_NAME, "user-area_link__1RH0V")) == 0:
    time.sleep(1)
time.sleep(1)
driver.find_element(By.CLASS_NAME, "user-area_link__1RH0V").click()
```

Fonte: Autoria Própria

Figura 20 - Botão de Login do Site ao inspecionar elemento (F12)

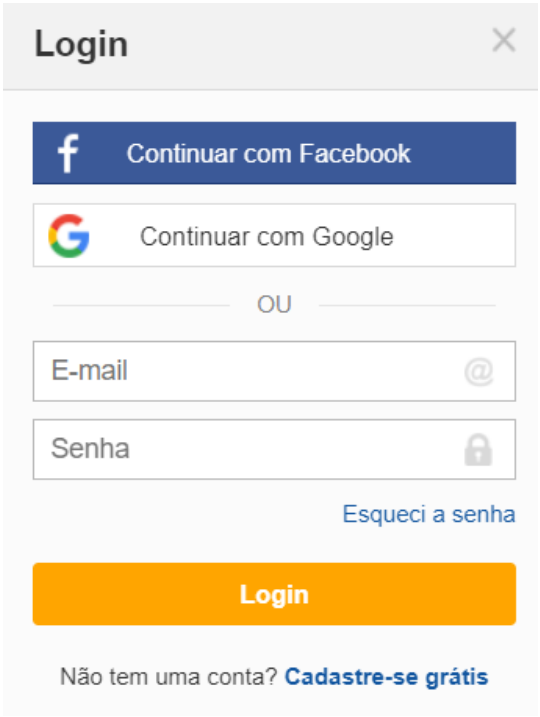


Fonte: adaptação de https://br.investing.com/currencies/eur-brl-historical-data

Após o botão de login ser clicado, é exibida a tela de login (Figura 21), e, da mesma forma é feito o preenchimento dos campos “E-mail” e “Senha”, só que em vez de clicar nos elementos em questão, é utilizado “.send_keys(“texto_que_queremos_enviar”)” para preencher os campos em questão (Figura 22).

[3] Essa estratégia é utilizada ao longo de todo o código e seu entendimento é de suma importância
[4] Aqui, o elemento tem duas classes: “inv-button” e “user-area-link__1RH0V”. No caso do código da Figura 19, é utilizado o nome “user-area-link__1RH0V” para localizar o elemento. Poderia ser utilizado “inv-button”, mas esse nome pode não ser único dentro da página como um todo, podendo apresentar erros na execução do programa caso fosse utilizado.

Figura 21 - Tela de Login do Site



Fonte: print de tela de https://br.investing.com/currencies/eur-brl-historical-data

Figura 22 - Código python responsável por preencher e-mail e senha nos campos da Figura 21

```
# Escreva o '[redacted]" no campo desejado
driver.find_element(By.ID, 'loginFormUser_email').send_keys([redacted])
# Como o campo de senha está no mesmo iframe do campo de email, escreva a senha:
driver.find_element(By.ID, "loginForm_password").send_keys([redacted])
# Clique no botão "Login":
driver.find_element(By.CLASS_NAME, "newButton").click()
```

Fonte: Autoria Própria

O código mostrado na Figura 22 se encontra dentro de um “try except”, dentro de um “for” (veja no APÊNDICE A). O “try except” é utilizado como tratamento de um erro onde os iframes[5] não eram encontrados no site em questão e funcionou desta forma: tente (try) código_x (Figura 23), caso dê errado (except), espere 4 segundos e execute código_x (Figura 23).

[5] “Iframe é a abreviação de inline frame, que nada mais é do que um elemento poderoso para quem trabalha como web designer. Ele é um código que é inserido dentro das páginas do seu site que abre outra página em determinado box.” (NETSUPPORT, 2022)

Figura 23 - Parte do código python (código_x) responsável por preencher o campo de login

```
# iframe é uma lista dos iframes presentes na página
iframe = driver.find_elements(By.TAG_NAME, "iframe")
# Percorrendo a lista de iframes
for i in range(len(iframe)):
    # Mudando para o iframe "i"
    driver.switch_to.frame(iframe[i])
    # Procura o ID 'loginFormUser_email' no iFrame "i" e armazena na lista "teste"
    teste = driver.find_elements(By.ID, 'loginFormUser_email')
    # Se a lista estiver vazia:
    if not teste:
        driver.switch_to.default_content() # Retorne para a página inicial
        pass # passa para o próximo item do for (i + 1)
    # Agora, se a lista tiver o ID que procuramos:
    else:
        while len(driver.find_elements(By.ID, 'loginFormUser_email')) == 0:
            time.sleep(1)
        time.sleep(1)
        # Escreva o "[redacted]" no campo desejado
        driver.find_element(By.ID, 'loginFormUser_email').send_keys('[redacted]')
        # Como o campo de senha está no mesmo iframe do campo de email, escreva a senha:
        driver.find_element(By.ID, "loginForm_password").send_keys('[redacted]')
        # Clique no botão "Login":
        driver.find_element(By.CLASS_NAME, "newButton").click()
        break
```

Fonte: Autoria Própria

Ao realizar o login, o último passo é executado: clicar no botão “baixar dados” e fechar o navegador (mesma estratégia utilizada anteriormente), representada na Figura 24.

Figura 24 - Código python responsável por clicar no botão "baixar dados", esperar o download e fechar o Google Chrome

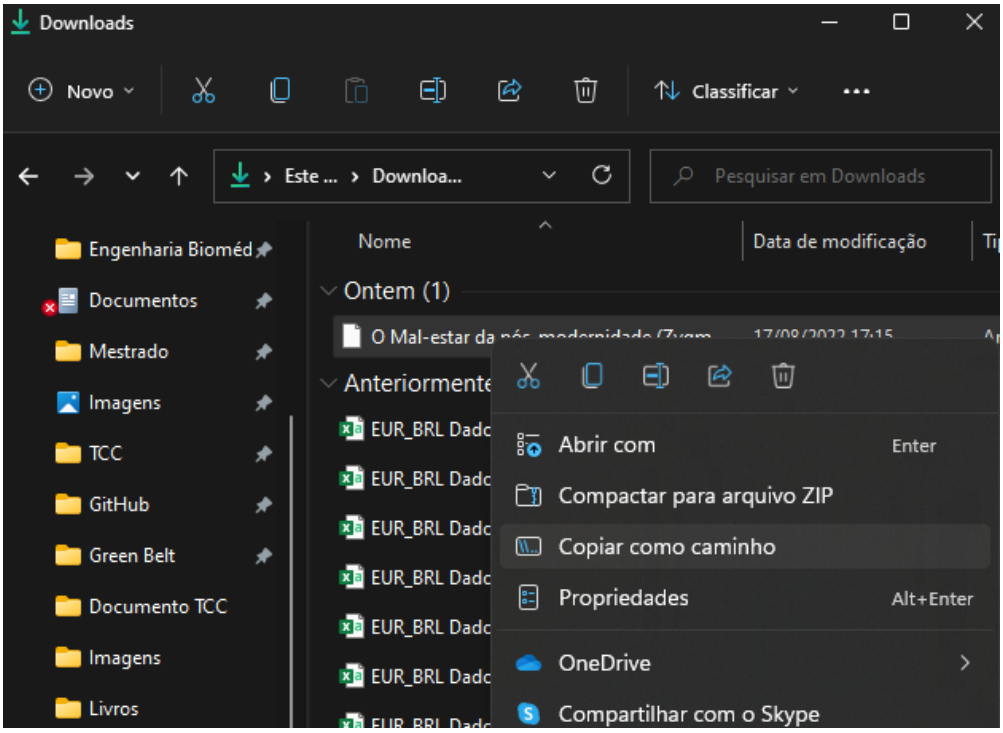
```
# Clica no botão "Baixar Dados"
# Caso ele não encontre o elemento, ele aguarda 1 segundo. Depois de encontrado,
# o elemento é clickado:
while len(driver.find_elements(By.CLASS_NAME, 'download-data_text__1gHMw')) == 0:
    time.sleep(1)
time.sleep(1)
driver.find_element(By.CLASS_NAME, 'download-data_text__1gHMw').click()

# Espera 4 segundos para realizar o download
time.sleep(4)

# Fecha navegador
driver.close()
```

Fonte: Autoria Própria

3.2.2. Manipulando o arquivo baixado com Pathlib

Agora, o segundo passo é pegar o arquivo baixado, criar uma pasta padrão, chamada de "AutoAnalise" e movimentar este arquivo para a pasta em questão. Para isso, será utilizado o caminho dos arquivos dentro do computador. Podemos descobrir o caminho de um arquivo ao clicar com o botão direito do mouse sobre ele e copiar seu caminho (Figura 25)

Figura 25 - Descobrindo o caminho de um arquivo



Fonte: Autoria Própria

Voltando à aplicação, o código da Figura 26 é uma função criada para descobrir o caminho do último arquivo de uma pasta: basta darmos à função a o caminho da pasta e a extensão do arquivo que queremos descobrir. Como resultado, a função retorna o caminho do último arquivo adicionado à pasta que queremos.

Como o último arquivo foi baixado e adicionado à pasta "Downloads", basta chamar a função "*descobrir_caminho_ultimo_arquivo("caminho_pasta_downloads, "extensão_do_arquivo_desejado"")*" para descobrir o caminho desejado. Com ele em mãos, basta criar a nova pasta "AutoAnalise" com *".mkdir()"* e mover o arquivo baixado para ela com "*sh.move("caminho_atual", "caminho_pasta_AutoAnalise")*" (Figura 27).

Figura 26 - Código python da função que descobre o caminho do último arquivo adicionado a uma pasta específica

```python
def descobrir_caminho_ultimo_arquivo(caminho, ext):
    '''
    Descobre o caminho do ultimo arquivo da pasta desejada
    - caminho: str. (caminho da pasta do arquivo)
    - ext: str. (extensão do arquivo. Ex: ext=".csv")
    '''
    # lista_arquivos recebe a lista de arquivos da pasta Downloads:
    lista_arquivos = os.listdir(caminho)
    lista_datas = []
    for arquivo in lista_arquivos:
        # descobrindo a data do arquivo:
        if ext in arquivo:
            data = os.path.getmtime(f"{caminho}/{arquivo}")
            lista_datas.append((data, arquivo))
    # Ordenando a lista de datas
    lista_datas.sort(reverse=True)
    # o ultimo arquivo será o primeiro item da lista de datas
    ultimo_arquivo = lista_datas[0]
    # ultimo arquivo retorna [data, arquivo] e, como
    # queremos o nome do ultimo arquivo, queremos ultimo_arquivo[1]
    nome_ultimo_arquivo = ultimo_arquivo[1]
    # Portanto, o caminho do ultimo arquivo pode ser descrito:
    caminho_ultimo_arquivo = caminho / Path(nome_ultimo_arquivo)
    return caminho_ultimo_arquivo
```

Fonte: Autoria Própria

Figura 27 - Código python que descobre a localização do arquivo baixado, cria uma nova pasta (AutoAnalise) e move o arquivo para essa pasta

```python
caminho_arquivo = descobrir_caminho_ultimo_arquivo("C:/Users/lucas/Downloads", ext=".csv")
caminho_atual = Path.cwd()

# Criando uma nova pasta no caminho atual chamada 'AutoAnalise':
if 'AutoAnalise' not in os.listdir(caminho_atual):
    caminho_pasta = caminho_atual / Path('AutoAnalise')
    Path(caminho_pasta).mkdir()
else:
    caminho_pasta = caminho_atual / Path('AutoAnalise')

# Movendo o arquivo para o caminho atual:
sh.move(str(caminho_arquivo), str(caminho_pasta))
```

Fonte: Autoria Própria

Agora, com os dados na pasta padrão, pode-se iniciar a análise e tratamento de dados, explicados no tópico seguinte.

3.2.3. Tratamento e Análise de Dados com Pandas

Quando baixamos uma planilha da internet ou de um banco de dados, muitas vezes, ela vem com formatos que impossibilitam a análise de dados. No presente caso, a planilha baixada no site da “Investing.com” vem com os números e datas no formato “*object*”, se comportando como textos (*string*) (Figuras 28 e 29). No mundo da programação, é impossível realizar cálculos com números armazenados em formato de texto e, para isso, antes da análise destes dados, deve-se realizar um tratamento adequado.

Figura 28 - Código python responsável por importar a planilha baixada e informar os tipos de dados desta planilha

```
# Importando a Planilha e armazenando-a em 'dados_df'
dados_df = pd.read_csv(descobrir_caminho_ultimo_arquivo(caminho_pasta, ext=".csv"))
display(dados_df.info())
```

Fonte: Autoria Própria

Figura 29 - Saída do código representado na Figura 28. “Dtype” representa os tipos de dados de cada coluna da planilha em questão.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24 entries, 0 to 23
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Data      24 non-null     object
 1   Último    24 non-null     object
 2   Abertura  24 non-null     object
 3   Máxima    24 non-null     object
 4   Mínima    24 non-null     object
 5   Vol.      24 non-null     object
 6   Var%      24 non-null     object
dtypes: object(7)
memory usage: 1.4+ KB
```

Fonte: Autoria Própria

A biblioteca Pandas possibilita tal tratamento e análise de forma simples e direta, como veremos adiante.

3.2.3.1. Tratando a coluna “Data”

A coluna “Data”, presente na planilha se mostrou no formato de texto: “DD de MMM. de AAAA”, onde “DD” são números que representam o dia, “MMM.” São letras que representam o mês e “AAAA” são números que representam o ano (Figura 30).

Figura 30 - Código python que mostra as 5 primeiras linhas da coluna "Data"

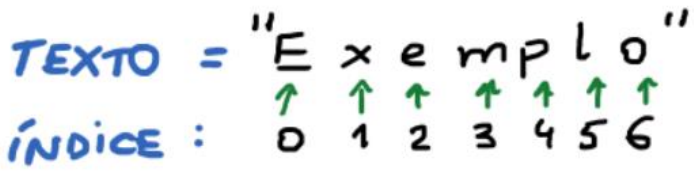
```
dados_df['Data'].head()
✓ 0.8s

0    18 de ago. de 2022
1    17 de ago. de 2022
2    16 de ago. de 2022
3    15 de ago. de 2022
4    12 de ago. de 2022
Name: Data, dtype: object
```

Fonte: Autoria Própria

Em *python*, o formato adequado para datas é o “*datetime*”, expressado no formato “DD/MM/AAAA”, onde “DD” são números inteiros que representam o dia, “MM” são números inteiros que representam o mês e “AAAA” são números inteiros que representam o ano.

O processo de transformação do formato “object” em “datetime” foi o processo mais demorado do tratamento pois envolveu a manipulação dos índices de strings. Na Figura 31, temos um exemplo de string (TEXTO = “Exemplo”) e seus respectivos índices.

Figura 31 - Exemplo de indexação de *strings*



Fonte: Autoria Própria

É possível manipular essas *strings* utilizando a indexação, como visto na Figura 32. Podemos cortar a *string* em partes: “*texto[:2]*” retorna as duas primeiras letras (“*Ex*”);

"*texto[1:4]*" retorna as letras de índices 1,2 e 3 ("*xem*"); "*texto[-2:]*" retorna as duas últimas letras ("*lo*").

Figura 32 - Exemplo de Manipulação de Índices de uma *String*

```
texto = "Exemplo"
print(texto[:2], texto[1:4], texto[-2:], sep='\n')
```

✓ 0.9s

```
Ex
xem
lo
```

Fonte: Autoria Própria

A partir da estratégia da Figura 32, foi construído o código da Figura 33.

Figura 33 - Código python responsável por manipular os índices das strings armazenadas na coluna "data"

```
# Manipulando a string presente na coluna Data para obter Dia, Mês e Ano
lista_dia = []
lista_mes = []
lista_ano = []
for data in dados_df['Data']:
    lista_dia.append(data[:2])
    lista_mes.append(data[6:9])
    lista_ano.append(data[-4:])
```

Fonte: Autoria Própria

Porém, como nesse caso as *strings* estão armazenadas dentro de uma tabela (DataFrame), é necessário utilizar um laço "*for*" para percorrer cada linha da coluna "*Data*". A partir disso, foi criado[6] uma lista para cada dia, outra para cada mês e outra para cada ano (*lista_dia*, *lista_mês* e *lista_ano*) (Figura 34)

[6] O método ".append()" é responsável por preencher uma determinada lista. Lista.append("algo"), adiciona "algo" à lista "Lista".

Figura 34 - Exibição das listas de dias, meses e anos criadas a partir do código da figura 33

```
print(lista_dia, lista_mes, lista_ano, sep='\n')
```
✓ 0.5s Python

```
['18', '17', '16', '15', '12', '11', '10', '09', '08', '05',
'04', '03', '02', '01', '29', '28', '27', '26', '25', '22',
'21', '20', '19', '18']
['ago', 'ago', 'ago', 'ago', 'ago', 'ago', 'ago', 'ago',
'ago', 'ago', 'ago', 'ago', 'ago', 'ago', 'jul', 'jul', 'jul',
'jul', 'jul', 'jul', 'jul', 'jul', 'jul', 'jul']
['2022', '2022', '2022', '2022', '2022', '2022', '2022',
'2022', '2022', '2022', '2022', '2022', '2022', '2022',
'2022', '2022', '2022', '2022', '2022', '2022', '2022',
'2022', '2022', '2022']
```

Fonte: Autoria Própria

Diante disso, foram criadas novas 3 colunas na tabela “dados_df” (Figura 35): Colunas “Dia”, “Mês” e “Ano” a partir das listas criadas (“lista_dia”, “lista_mes” e “lista_ano”).

Figura 35 - Criação das colunas "Dia", "Mês"e "Ano"

```
# Criando as colunas Dia, Mês e Ano:
dados_df['Dia'] = lista_dia
dados_df['Mês'] = lista_mes
dados_df['Ano'] = lista_ano
```

Fonte: Autoria Própria

Agora, os meses, representados por letras, devem passar a ser representados por números. Para isso, criou-se um dicionário python (“dic_mes”) e inserido dentro da função “replace()”, responsável por trocar strings. Nesse caso, ela substitui “jul” por “7” e “ago” por “8” e isso já transforma a coluna em uma coluna de números inteiros (Figura 36).

Figura 36 - Código responsável por substituir os meses por números inteiros

```
# Substituindo mês str por mês int:
dic_mes = {'jan': 1, 'fev': 2, 'mar': 3, 'abr': 4, 'mai': 5, 'jun': 6, 'jul': 7, \
           'ago': 8, 'set': 9, 'out': 10, 'nov': 11, 'dez': 12}
dados_df['Mês'] = dados_df['Mês'].replace(dic_mes)
```

Fonte: Autoria Própria

O próximo passo foi transformar as colunas “Ano” e “Dia” em numerais, excluir a coluna “Data” antiga, transformar as colunas “Dia”, “Data” e “Ano” em *strings* para então unilas em uma nova coluna “Data” no formato “DD/MM/AAAA”. (Figura 37)

Figura 37 – Código *python* responsável por criar a nova coluna "Data" no formato "DD/MM/AAAA"

```
# Transformando a coluna ano e dia em numeral
dados_df['Ano'] = pd.to_numeric(dados_df['Ano'])
dados_df['Dia'] = pd.to_numeric(dados_df['Dia'])

# Excluindo a coluna Data
dados_df = dados_df.drop('Data', axis=1)

# Transformando as colunas Dia, Mes e Ano em strings
dados_df['Ano'] = dados_df['Ano'].map(str)
dados_df['Dia'] = dados_df['Dia'].map(str)
dados_df['Mês'] = dados_df['Mês'].map(str)

# Criando uma nova coluna Data e unindo Dia/Mês/Ano
dados_df['Data'] = dados_df['Dia'] + '/' +  dados_df['Mês'] + '/' + dados_df['Ano']
```

Fonte: Autoria Própria

Dessa maneira, pode-se passar a coluna “Data” para o formato “datetime” através do método “pd.to_datetime(Coluna_x, format = “DD/MM/AAAA”)”. Após isso, foi feita a exclusão das colunas “Dia”, “Mês” e “Ano” e a reordenação das colunas da tabela (Figura 38).

Figura 38 - Código *python* responsável por transformar a coluna "Data" em “datetime” e reordenar as colunas da tabela

```
# Transformando a coluna Data para o tipo 'DateTime'
dados_df['Data'] = pd.to_datetime(dados_df['Data'], format='%d/%m/%Y')

# Excluindo as colunas Dia, Mês e Ano:
dados_df = dados_df.drop(['Dia', 'Mês', 'Ano'], axis=1)

# Reordenando as colunas
dados_df = dados_df[['Data', 'Último', 'Abertura', 'Máxima', 'Mínima', 'Vol.', 'Var%']]
```

Fonte: Autoria Própria

3.2.3.2. Tratando o resto das colunas

A estratégia do tratamento das colunas restantes foi mais rápida, uma vez que não precisou fazer o mesmo tratamento de indexação feito na coluna “Data” (Figura 39).

Figura 39 - Restante das colunas

```
dados_df[['Último', 'Abertura', 'Máxima', 'Mínima', 'Vol.', 'Var%']].head()
```

✓ 0.6s

	Último	Abertura	Máxima	Mínima	Vol.	Var%
0	5,2534	5,2585	5,2681	5,2176	118,79K	-0,10%
1	5,2585	5,2325	5,2938	5,2227	259,08K	0,49%
2	5,2327	5,1797	5,2460	5,1608	212,03K	1,07%
3	5,1776	5,2039	5,2432	5,1646	195,15K	-0,51%
4	5,2044	5,3227	5,3287	5,1964	235,13K	-2,20%

Fonte: Autoria Própria

As colunas “Último”, “Abertura”, “Máxima” e “Mínimo” apresentaram o mesmo formato: números que utilizam a vírgula como separador decimal. Porém, antes de transformá-los em números inteiros, é necessário trocar a vírgula pelo ponto (“.replace(',', '.', regex=True)”). Após isso, basta transformar em números (".astype(float)”), como visto na Figura 40.

Figura 40 - Código *python* responsável por converter as colunas "Último", "Abertura", ""Máxima e "Mínima" em *float*

```
# Convertendo as colunas 'Último', 'Abertura', 'Máxima' e 'Mínima' para float
aux_df = dados_df[['Último', 'Abertura', 'Máxima', 'Mínima']]
aux_df = aux_df.replace(',', '.', regex=True ).astype(float)
```

Fonte: Autoria Própria

A estratégia utilizada na coluna “Vol.” foi similar, porém, antes de retirar a vírgula e transformar em *float*, precisou-se retirar a letra “K”. Após a transformação em *float*, foi necessário multiplicar o valor por 1000, uma vez que “K” é o mesmo que realizar essa multiplicação (Figura 41).

Figura 41 - Código *python* responsável pela transformação da coluna "Vol." em *float*

```
# Tratando a coluna Vol (K = x1000 e passando para float)
dados_df['Vol.'] = dados_df['Vol.'].replace('K', '',\
    regex=True).replace(',', '.', regex=True).astype(float) * 1000
```

Fonte: Autoria Própria

Agora, para a última coluna restante, “Var%”, foi utilizada a mesma estratégia para retirar o símbolo “%” do número. Porém, foi escolhido representar os itens da coluna como números somente positivos. Para isso, criou-se um laço “for” para percorrer a coluna e preencher uma nova lista, chamada “var_list”. Caso, o número da coluna tivesse o símbolo “-”, o novo número seria 100 menos o número antigo, caso contrário, seria 100 mais o número antigo (Figura 42).

Figura 42 - Código *python* responsável por formatar a coluna "Var%"

```
# Recriando a coluna Var% e transformando em float
dados_df['Var%'] = dados_df['Var%'].replace('%', '', regex=True)
var_list = []
for num in dados_df['Var%']:
    if '-' in num:
        num = num.replace('-','').replace(',','.')
        num = float(num)
        num = 100 - num
        var_list.append(num)
    else:
        num = num.replace(',','.')
        num = float(num)
        num = 100 + num
        var_list.append(num)
```

Fonte: Autoria Própria

O último processo de tratamento de dados foi a exclusão da antiga coluna “Var%”, a criação da nova coluna “Var%” a partir da lista “var_list”, a reordenação da tabela (dias mais antigos para dias mais recentes) e, por último, salvamento da planilha excel como “Dados_Analisados.xlsx” na pasta “AutoAnalise” (Figura 43).

Figura 43 – Código *python* responsável pela criação da nova coluna "Var%", reordenação da tabela e salvamento da planilha em Excel

```
dados_df = dados_df.drop('Var%', axis=1) # Excluindo a antiga coluna Var%
dados_df['Var%'] = var_list

# Reordenando a tabela: dia mais antigo para dia mais recente
dados_df = dados_df.sort_values('Data')
dados_df = dados_df.reset_index(drop=True) # Resetando o índice

# Passando a planilha para o Excel:
dados_df.to_excel(caminho_pasta / Path('Dados_Analisados.xlsx'), index = False)
```

Fonte: Autoria Própria

3.2.3.3. Análise de dados com Pandas

Com a planilha tratada de forma devida, foi possível realizar a análise dos dados nela presentes. Foi obtido, então: máximo e mínimo do euro; gráfico da variação do euro e as médias móveis de 5 e 7 dias (dados comumente utilizados no mercado financeiro).

A Figura 44 demonstra a obtenção dos dados de máximo (atribuída à variável "*max*") e mínimo (atribuída à variável "*min*") bem como o esboço do gráfico e seu salvamento na pasta "AutoAnalise" (*plt.savefig("caminho_pasta_AutoAnalise")*).

Figura 44 - Código *python* responsável pela análise dos dados da planilha tratada

```
# Obtendo valores máx e min:
max = dados_df['Último'].max()
min = dados_df['Último'].min()

# Esboçando o gráfico da variação do Euro bem como as médias móveis 5 e 7
dados_df['Último'].plot(figsize=(15, 5), label = 'EUR', xlabel = 'Data', ylabel = 'EUR em R$')
dados_df['Último'].rolling(5).mean().plot(label='MM5')
dados_df['Último'].rolling(7).mean().plot(label='MM7')
plt.xticks(range(len(dados_df['Data'])), dados_df['Data'])
plt.xticks(rotation = 90)
plt.title('Euro no Último Mês')
plt.legend()
plt.savefig(caminho_pasta / Path('grafico_EUR.jpg'), bbox_inches = 'tight')
```

Fonte: Autoria Própria

Além dessas análises, foi criada uma função para determinar se existe uma alta ou uma mínima do euro durante o mês analisado: *alta_ou_queda()*. A partir desta função, calcula-se se o último valor é maior ou menor do que o primeiro, retornando, a depender do resultado, "Alta" ou "Queda" (Figura 45).

Figura 45 - Código *python* responsável pela definição da função "alta_ou_queda()"

```
def alta_ou_queda():
    '''Calcula, com base na planilha dados_df se existe uma alta do Euro'''
    x = len(dados_df['Último'])
    if dados_df['Último'][0] > dados_df['Último'][x-1]:
        return 'Queda'
    else:
        return 'Alta'
```

Fonte: Autoria Própria

3.2.4. Envio dos dados por e-mail com Smtplib

Agora, com todos os dados analisados e salvos na pasta “AutoAnalise”, basta enviá-los por e-mail. Para isso, utilizou-se a biblioteca Smtplib. Foi criado, então, a função “enviar_email()” e, dentro dela, foi feito o código para o envio do e-mail: basta chama-la para o envio efetivo.

O primeiro passo, foi a criação do corpo do e-mail (Figura 46). A variável “corpo_email” recebe uma *string*, entre aspas triplas (“”” “”””), as quais permitem formular um *script* maior. O *script* foi escrito em *html* para facilitar a formatação do e-mail. Diante disso, foi utilizado:

- (texto escrito aqui) para construir parágrafos;
-
 para pular linhas;
- (texto escrito aqui) para formatar letras em negrito;
- (texto escrito aqui) para formatar letras em itálico;
- (texto aqui) é utilizada para alinhar o parágrafo à direita;
- (texto aqui) é utilizado para determinar o tamanho da letra.

Além disso, o “*.format()*” ao final da *string* permite inserir dados nos colchetes vazios “{}” dentro da *string* e, nesse caso, foi inserido o valor retornado da função “*alta_ou_queda()*”, as variáveis numerais *float* “*max*” e “*min*”.

Figura 46 - Código *python* responsável por criar o corpo do e-mail

```
# Criação do Corpo do E-mail
corpo_email = """
<p>Olá, Lucas! Tudo bom?</p>
<br>
<p>Segue a análise do Euro no último mês, como solicitado</p>
<p>O euro esteve, aparentemente, em <b>{}</b> nesse último mês: \
    <b>máximo de R${:.2f}</b> e <b>mínimo de R${:.2f}</b></p>
<p>Em anexo, segue o gráfico do euro neste mês contendo a média \
    móvel de 5 e 7 dias, bem como a planilha utilizada para a análise</p>
<p>Qualquer coisa estou à disposição!</p>
<br><br>
<p>Att.</p>
<p>Lucas Guaragna</p>
<br>
<p align='right'><font size='1'><i>este e-mail foi enviado automaticamente \
    utilizando <b>Python</b></i></font></p>
""".format(alta_ou_queda(), max, min)
```

Fonte: Autoria Própria

Em seguida (Figura 47), deve-se criar a mensagem, com "*msg = EmailMessage()*", inserir o assunto da mensagem em "*msg['Subject']*", o e-mail do remetente em "*msg['From']*", o e-mail do destinatário em "*msg['To']*". A senha é gerada no site do gmail em "Senhas de App" [7] e é atrelada e única ao remetente criado.

Figura 47 - Código *python* responsável pela criação da mensagem, definição do remetente, destinatário e senha, bem como o corpo do e-mail

```
# Criação de uma mensagem
msg = EmailMessage()
# Assunto da mensagem
msg['Subject'] = "Análise EUR/R$"
# Remetente
msg['From'] = [redacted]
# Destinatário
msg['To'] = [redacted]
# Senha - Gerada pelo google em "Senhas de App"
password = [redacted]
msg.add_header('Content-Type', 'text/html')
# Corpo de e-mail
msg.set_payload(corpo_email )
```

Fonte: Autoria Própria

[7] Mais informações em https://support.google.com/accounts/answer/185833?hl=pt-BR

Por último, foi anexado os arquivos “Dados_Analisados.xlsx” e “grafico_EUR.jpg” bem como as credenciais necessárias para envio (“smtp.gmail.com: 587” e adiante) (Figura 48)

Figura 48 - Código *python* responsável por anexar os arquivos e formatar as credenciais necessárias para o envio do e-mail

```
#Anexando os arquivos:
mime_type, encoding = mimetypes.guess_type(caminho_planilha)
with open(caminho_planilha, 'rb') as fp:
    dados = fp.read()
msg.add_attachment(dados, maintype=mime_type.split("/")[0], \
    subtype=mime_type.split("/")[1], filename='Dados_Analisados.xlsx')

mime_type, encoding = mimetypes.guess_type(caminho_grafico)
with open(caminho_grafico, 'rb') as fp:
    dados = fp.read()
msg.add_attachment(dados, maintype=mime_type.split("/")[0], \
    subtype=mime_type.split("/")[1], filename='grafico_EUR.jpg')

#
s = smtplib.SMTP('smtp.gmail.com: 587')
s.starttls()
# Login Credentials for sending the mail
s.login(msg['From'], password)
s.sendmail(msg['From'], [msg['To']], msg.as_string().encode('utf-8'))
print('Email enviado')
```

Fonte: Autoria Própria

Para finalizar o programa, foi criado um pop-up que indica uma mensagem final para o usuário: “Programa concluído com sucesso! Os arquivos enviados por e-mail se encontram na pasta AutoAnalise, localizada na mesma pasta do arquivo executável deste programa” (Figura 49).

Figura 49 - Código *python* responsável por criar um Pop-Up que finaliza o programa criado

```
# Abrindo a mensagem final do programa:

janela = Tk()
messagebox.showinfo('Status do Programa', 'Programa concluído com sucesso! \
    Os arquivos enviados por e-mail se encontram na pasta AutoaAnalise, \
        localizada na mesma pasta do arquivo executável deste programa')
janela.destroy()
```

Fonte: Autoria Própria

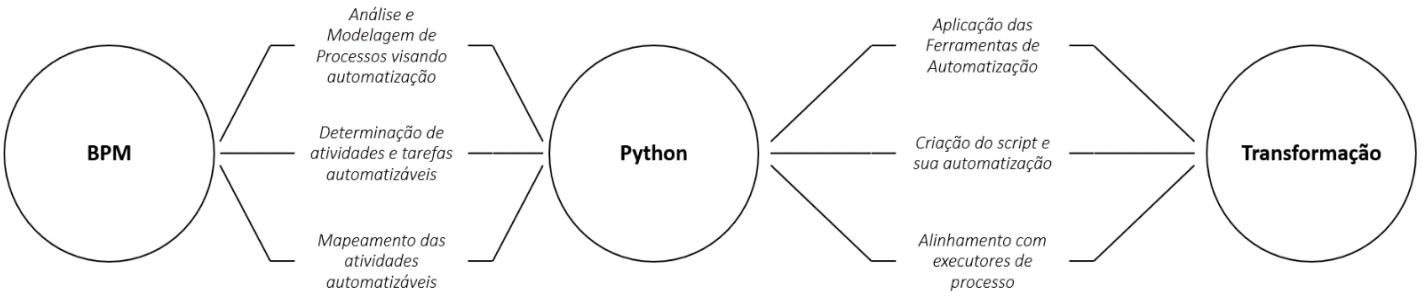
Com isso, concluiu-se o script *python* e foi iniciado o processo de transformação do script ".py" em ".exe", tratado no tópico 2.2.6, e a utilização do Agendador de Tarefas para executar o programa em uma determinada frequência, tratado no tópico 2.2.7, concluindo-se, assim, a metodologia deste trabalho.

4 RESULTADOS E DISCUSSÃO

4.1. Resultados

O estudo das práticas de BPM e das ferramentas de automatização com *python* possibilitou a formulação de uma hipótese: **a utilização das práticas de BPM combinadas com a automatização de atividades e tarefas com *python* carrega consigo a possibilidade de otimização, ou ainda, uma mudança de paradigma**. A Figura 50 descreve as iniciativas de aplicação dessa proposta.

Figura 50 - BPM + Automatização de Atividades e Tarefas com Python

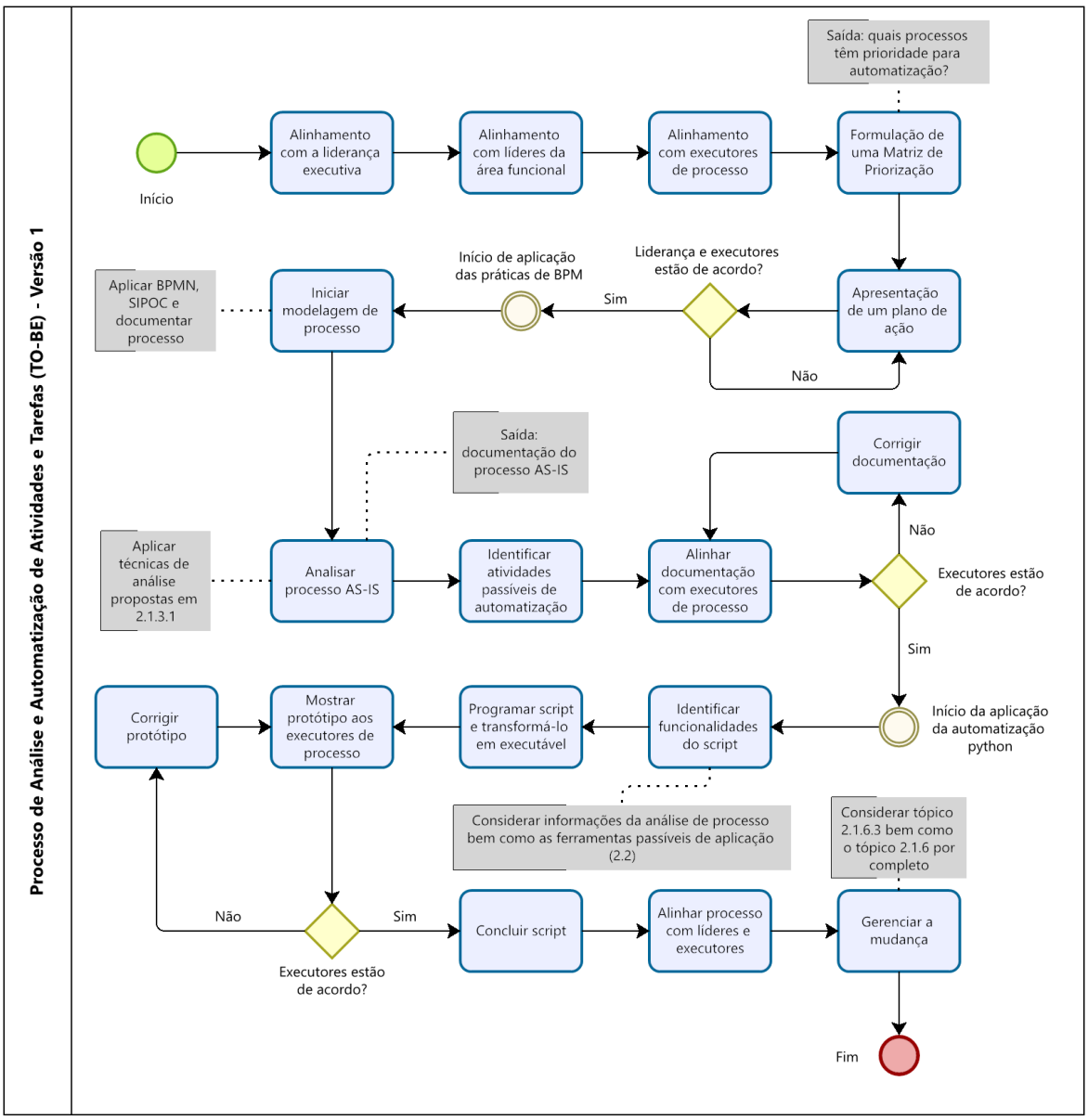


Fonte: autoria própria

As práticas de BPM contribuem para a análise e modelagem de processos, bem como a determinação de atividades e tarefas passíveis de automatização e seu mapeamento. A geração da documentação do processo contribui para o processo de aplicação das ferramentas de automatização, criação do script e automatização. Com o alinhamento do novo processo (TO-BE) com os executores e o gerenciamento da mudança, conclui-se a transformação.

Diante dessa hipótese, formulou-se então o fluxograma da Figura 51, o qual busca a união das práticas de BPM e automatização *python* seguindo os passos descritos: primeiro a aplicação de BPM, analisando e documentando processos passíveis de automatização e, em segundo, a utilização do *python* como ferramenta de automatização – Melhor indicado na figura em questão.

Figura 51 - Fluxograma do Processo de Análise e Automatização de Atividades e Tarefas



Fonte: Autoria própria

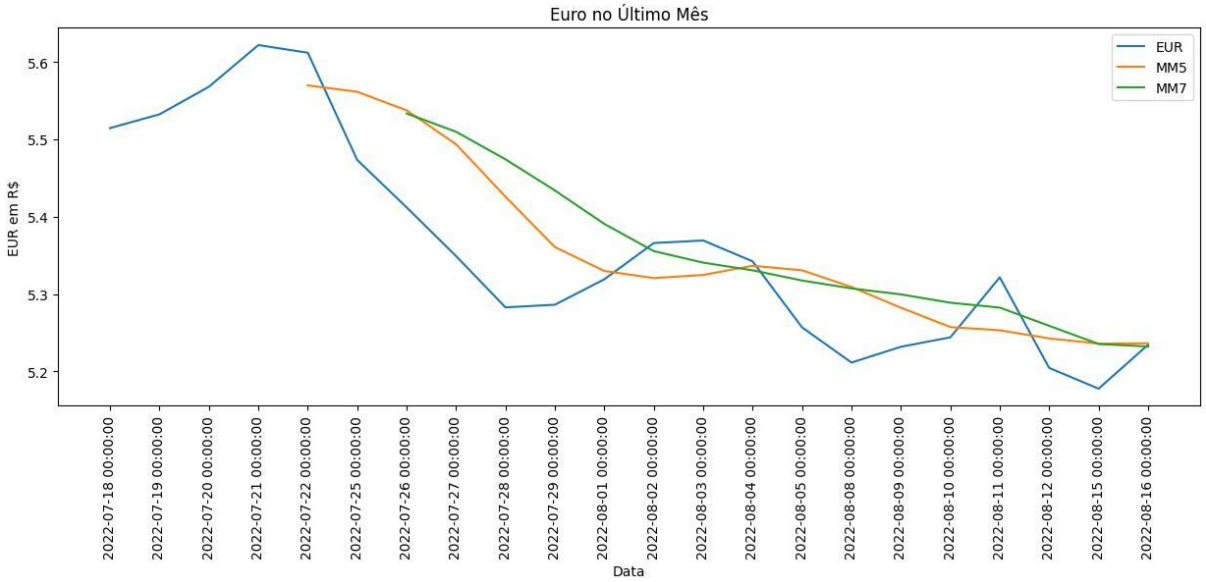
Como não foi possível aplicar as práticas de BPM em algum negócio específico, foi feito então, a automatização de um processo fictício, como foi tratado nos Métodos deste trabalho. A aplicação da automatização *python* gerou, como saída de processo, o e-mail indicado na Figura 52.

Figura 52 - E-mail enviado automaticamente pelo *script python*

Fonte: Autoria própria

Os anexos incluídos no e-mail da Figura 52 são apresentados nas Figuras 53 (Gráfico do Euro) e 54 (Planilha Excel).

Figura 53 - Gráfico gerado pelo *script python* apresentado na Figura 44, baseado no tópico 3.2.3 e anexado no E-mail apresentado na Figura 52



Fonte: Autoria própria

Figura 54 - Planilha gerada por *script python* a partir do tratamento de dados (tópico 3.2.3) e anexada no e-mail apresentado na Figura 52

Dados_Analisados.xlsx

	A	B	C	D	E	F	G
1	Data	Último	Abertura	Máxima	Mínima	Vol.	Var%
2	2022-07-18	5,5147	5,4555	5,5183	5,433	219550	101,07
3	2022-07-19	5,5325	5,5153	5,5847	5,4917	236460	100,32
4	2022-07-20	5,5682	5,5368	5,5721	5,5082	259020	100,65
5	2022-07-21	5,6221	5,5699	5,6284	5,5598	265910	100,97
6	2022-07-22	5,6122	5,6232	5,6266	5,5419	240850	99,82
7	2022-07-25	5,4735	5,6153	5,6409	5,4742	209140	97,53
8	2022-07-26	5,4121	5,4771	5,4924	5,4015	251510	98,88
9	2022-07-27	5,3491	5,4143	5,444	5,3398	252360	98,84
10	2022-07-28	5,2828	5,3478	5,3677	5,2537	271700	98,76
11	2022-07-29	5,2862	5,2863	5,3277	5,2382	247000	100,06
12	2022-08-01	5,3191	5,2898	5,338	5,2464	249810	100,62
13	2022-08-02	5,366	5,3211	5,3756	5,2969	291150	100,88
14	2022-08-03	5,3693	5,3677	5,3886	5,3375	219360	100,06
15	2022-08-04	5,3423	5,3709	5,3898	5,3184	244890	99,5
16	2022-08-05	5,2569	5,3428	5,364	5,2483	242790	98,4
17	2022-08-08	5,2114	5,2572	5,2762	5,2024	198390	99,13
18	2022-08-09	5,2318	5,2141	5,2688	5,2076	202700	100,39
19	2022-08-10	5,244	5,2346	5,2815	5,2052	236830	100,23
20	2022-08-11	5,3216	5,246	5,3362	5,2337	251540	101,48
21	2022-08-12	5,2044	5,3227	5,3287	5,1964	235130	97,8
22	2022-08-15	5,1776	5,2039	5,2432	5,1646	195150	99,49
23	2022-08-16	5,2344	5,1797	5,246	5,1608	212030	101,1

Fonte: Autoria própria

O tempo de ciclo do processo foi de aproximadamente **1 minuto**[8], desde o acesso do site até o envio do e-mail. O processo mais demorado, dentro da execução do *script* foi a parte que utilizou a biblioteca ***Selenium***, uma vez que depende da velocidade de processamento do computador para abrir o Chrome, variando entre **30 e 50 segundos** para sua execução. A criação da pasta e a movimentação do arquivo, com ***pathlib***, variou entre **0,1 e 0,9 segundo**. O tratamento e análise de dados, com ***pandas***, variou entre **0,5 e 1 segundo**. Finalizando o processo, o envio do e-mail, com ***smtplib***, variou entre **3 e 5 segundos**.

4.2. Discussão

Uma vez que a utilização das práticas de BPM traz à tona os gargalos e uma documentação robusta dos processos de negócios de maneira fidedigna, como visto no trabalho de Isabella (CARDOSO, 22), e a utilização do *python* como ferramenta de transformação de processo traz benefícios visíveis, como visto no trabalho de João (NETO, 2021), a hipótese da combinação das práticas de BPM com a automatização *python*, formulada neste documento, se sustenta por sua atualidade e sua possibilidade de demonstrar alta competitividade no mercado de trabalho.

O estudo do BPM, no presente trabalho, trouxe a formulação da ideia necessária para criar a Figura 51: hipótese de junção BPM/Python. Porém, apesar de não haver a aplicação das práticas de BPM, a base teórica, a literatura e a experiência do autor na Gestão de Processos sustentam a possibilidade de melhoria/mudança de paradigma a partir da aplicação de tal hipótese. Com isso, pode-se levar em consideração a alta importância de se entender um processo (modelagem, análise, desenho e gerenciamento de processos) antes de efetuar qualquer tipo de transformação.

Diante disso, ao se analisar os resultados da automatização, percebe-se a **oportunidade de ganho de tempo de processo**: se o processo de download, tratamento, análise, e envio de dados fosse realizado manualmente, com certeza duraria muito mais do que 1 minuto; além de poder haver erros associados ao manuseio de dados em excesso (erro humano).

Deve-se considerar, também, que o grau de análise, tratamento e geração de dados deste trabalho se apresenta como **moderado,** uma vez que existem análises muito mais rebuscadas e complexas do que a feita. O foco, aqui, é mostrar a **possibilidade de automatização** e **melhoria**

[8] O tempo foi estipulado a partir do tempo de execução indicado na IDE Visual Studio Code.

associada, o que, por sua vez, não deixa de lado a oportunidade de criação de análises mais complexas que satisfaçam as aplicações de um negócio[9].

Além disso, a utilização de *python* carrega consigo uma proposta de mudança de paradigma dentro de um negócio: seu uso sustenta o fato de ser uma abordagem que **permite criação**, **difusão** e **incorporação de conhecimentos**, bem como a possibilidade de ser usada como **vantagem competitiva**[10].

A automatização de processos **não é** a substituição de trabalho de pessoas, e **sim** uma **possibilidade de redirecionamento do foco dos executores para processos que agregam maior valor ao cliente de processo**. Além disso, ao unir a automatização com as práticas de BPM, existe a possibilidade de **aumento da cultura da mudança** dentro de ambientes de negócio, o que acarreta, consequentemente, um aumento na produtividade.

[9] Python é uma linguagem de programação que possibilita uma alta complexidade de tratamento e análise de dados e, com certeza, pode atender qualquer aplicação de mercado.
[10] Fatores tratados no tópico de “Transformação de Processos” (Tópico 2.1.6)

5 CONCLUSÃO

Recapitulando o que foi apresentado: as práticas de BPM apresentam as bases para se entender, documentar, analisar, mensurar e transformar um processo de negócio de uma maneira robusta e padronizada. Após sua aplicação, *python* se mostra como uma forma de transformação de processo, focada na automatização de atividades e tarefas, bem como uma proposta de mudança de paradigma: redirecionamento do foco para processos de maior valor; e, consequentemente, a abdicação da execução de atividades e tarefas burocráticas e repetitivas. Finalmente, é apresentada uma aplicação de automatização do processo de download, tratamento, análise e envio de dados por e-mail, concluindo-se o trabalho.

A apresentação dos temas se mostrou de forma qualitativa e didática ao longo de todo o trabalho, assim como programado nos objetivos do trabalho. O referencial teórico prepara o leitor para o entendimento da hipótese de junção BPM/*python* bem como a aplicação de automatização de processo.

Cabe salientar que a utilização da linguagem de programação *Python* é somente uma das diversas possibilidades de melhoria de processos observadas a partir da aplicação das práticas de BPM, uma vez que tais práticas possibilitam uma visão holística de processo e apontam diversos tipos de gargalos diferentes. Portanto, como possibilidade de estudo futuro, pode-se explorar a utilização de tecnologias diferentes da computação para a melhoria de processos, tais como a *inteligência artificial* e *machine learning*, bem como a utilização de metodologias ágeis de gerenciamento como o *scrum* e o *lean six sigma* na área de gerenciamento de processos.

A aplicação do conteúdo presente neste documento pode se enquadrar em qualquer tipo de negócio, uma vez que todo negócio carrega consigo a necessidade de gerenciamento de processos. Com isso, o tema desse projeto se enquadra dentro da disciplina de Engenharia Clínica, um dos ramos da Engenharia Biomédica que trata da gestão, não só dentro do setor de Engenharia Clínica, mas em um estabelecimento assistencial de saúde como um todo.

Diante disso, para trabalhos futuros, existe o interesse do autor em aplicar tal hipótese em estabelecimentos assistenciais de saúde, uma vez que são ambientes que carecem, de certa forma, deste tipo de tecnologia, além de poder proporcionar uma maior satisfação do cliente final (pacientes), uma vez que profissionais da saúde poderão focar em processos que agregam maior valor a ele.

REFERÊNCIAS BIBLIOGRÁFICAS

ABPMP. Institucional | ABPMP Brasil, 2022. Página Institucional. Disponível em < https://www.abpmp-br.org/sobre-a-abpmp/institucional/>. Acesso em: 17 de jun. de 2022.

ABPMP. **BPM CBOK V.3.0: Guide to the Business Process Management Common Book of Knowledge**. ABPMP International, p. 35-40, 72, 79, 99, 127, 129, 134, 135, 143, 145, 152, 154, 158, 169, 183, 187, 188, 191-194, 197, 200, 201, 207, 233, 234, 236, 240, 242, 244, 248, 249, 251, 254, 255, 264, 267, 268, 271, 272, 274, 278. 2013.

AUTO-PY-TO-EXE. Auto-py-to-exe PyPl, 2022. Disponível em: https://pypi.org/project/auto-py-to-exe/. Acesso em 22 de jul. de 2022.

BPMN Quick Guide, 2022. Página Inicial. Disponível em <https://www.bpmnquickguide.com/>. Acesso em: 22 de jul. de 2022

CARDOSO, I. D. **Adoção do BPM na Melhoria de Processos Administrativos: Um Estudo de Caso Sobre o Processo de Criação e Registro de Empresas Juniores na UFOP**. Ouro Preto, 2022.

CHROMEDRIVER. Webdriver for Chrom – Downloads, 2022. Disponível em: https://chromedriver.chromium.org/downloads. Acesso em: 22 de jul. de 2022.

CURSO DE PYTHON ONLINE. Curso de python online – Aprenda com a prática – Hashtag, 2022. Disponível em: https://www.hashtagtreinamentos.com/curso-python. Acesso em 22 de jul. de 2022.

GIL, A. C.; **Como Elaborar Projetos de Pesquisa.** São Paulo, Brasil, 4ª edição, 2002. Editora Atlas S.A.

GÓMEZ, G.; WHITTINGHAM-JONES, L.; GULATI, S.; RIOUX, S.; VÁZQUEZ, A.; BRENNAN, R.; EHMEN, S.; GEORGE, A.; DAGLEY, S.; NEUBAUER, J.; QUINTANA, F.; PEREIRÓ, E.; SANCHEZ, J.; MANSER, M. **Bizagi Modeler**: Software de Mapeamento de

Processos. Versão 4.0.0.014, 2022. Disponível em: https://www.bizagi.com/pt/plataforma/modeler. Acesso em: 22 de jul. de 2022.

MICROSOFT. **Agendador de Tarefas**: Software para agendamento de tarefas. Versão 1.0, 2022.

MUTHUKADAN, B. Selenium with python – Selenium python bindings 2 documentation, 2022. Disponível em: https://selenium-python.readthedocs.io/index.html. Acesso em: 22 de jul. de 2022

MUTHUKADAN, B. Locating Elements. Selenium with python – Selenium python bindings 2 documentation, 2022. Disponível em: https://selenium-python.readthedocs.io/locating-elements.html. Acesso em 22 de jul. de 2022.

NETO, J. A. **Automatização da Geração de Ressuprimento de Materiais por Meio de Webscraping do Portal do Painel de Preços e Integração com Sistema Laborti**. Brasília, 2021.

NETSUPPORT, 2022. **IFRAME: ENTENDA O QUE É, PARA QUE SERVE E COMO UTILIZAR**. Disponível em: <https://netsupport.com.br/iframe-entenda-o-que-e-para-que-serve-e-como utilizar/#:~:text=Iframe%20%C3%A9%20a%20abrevia%C3%A7%C3%A3o%20de,outra%20p%C3%A1gina%20em%20determinado%20box>. Acesso em: 18 de Agosto de 2022

PANDAS DOCUMENTATION. Pandas 1.4.3 documentation, 2022. Disponível em: https://pandas.pydata.org/docs/index.html. Acesso em: 22 de jul. de 2022.

PATHLIB. Object-oriented filesystem paths, 2022. Disponível em: https://docs.python.org/3/library/pathlib.html. Acesso em: 22 de jul. de 2022

PYODBC WIKI. Home – mkleehammer/Pyodbc Wiki – GitHub, 2022. Disponível em: https://github.com/mkleehammer/pyodbc/wiki. Acesso em: 22 de jul. de 2022

PRODANOV, C. C.; FREITAS, E. C. **Metodologia do Trabalho Científico: Métodos e Técnicas de Pesquisa e do Trabalho Acadêmico.** Novo Hamburgo, Brasil, 2ª edição, 2013. Editora Feevale.

SMTPLIB. **Smtplib – SMTP protocol cliente – Python documentation**, 2022. Disponível em: <https://docs.python.org/3/library/smtplib.html>. Acesso em: 19 de Ago. de 2022

SCHWAB, Klaus. **The Fourth Industrial Revolution**. Suíça: World Economic Forum, 2016

SILVA, I. R. S.; SILVA, R. O. **Linguagem de Programação Python.** Revista Tecnologias em Projeção. Brasília, v. 10, n. 1, p. 56, 57 e 59, 2019.

APÊNDICE A – CÓDIGO *PYTHON* DO EXEMPLO DE APLICAÇÃO

Exemplo de Aplicação

Este código pertence a **Lucas Guaragna Guedes** e foi apresentado em seu trabalho de conclusão de curso (Bacharelado em Engenharia Biomédica) da Universidade Federal de Uberlândia em Agosto de 2022

Bibliotecas Utilizadas:

- **Selenium**: manipulação de dados web (webscrapping)
- **Pathlib**: manipulação de arquivos
- **Pandas**: manipulação de dados

Instalação das Bibliotecas:

- As bibliotecas devem ser instaladas no prompt de comando: *pip install biblioteca_x*

Importando Bibliotecas

In [ ]:

```
# Bibliotecas utilizadas em todo o código
import time

# Bibliotecas necessárias para utilização do Selenium:
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager

# Bibliotecas necessárias para a utilização do Pandas:
from datetime import datetime
import pandas as pd
import matplotlib.pyplot as plt

# Bibliotecas necessárias para utilização do Pathlib:
from pathlib import Path
import shutil as sh
import os

# Bibliotecas necessárias para a utilização do Smtplib:
import smtplib
import email.message
import mimetypes
from email.message import EmailMessage

# Biblioteca necessária para abrir a caixa de texto final do programa
from tkinter import *
import tkinter.filedialog
from tkinter import messagebox
```

1. Selenium

Passos:

- Entrar no Google Chrome
- Inserir o site https://br.investing.com/currencies/eur-brl-historical-data
- Desativar os pop-ups
- Realizar login no site
- Baixar planilha de dados
- Fechar navegador

In [ ]:

```
# Abre o Chrome e certifica o download do chromedriver de versão adequada
# chrome_options = webdriver.ChromeOptions()
# chrome_options.add_argument('headless')
driver = webdriver.Chrome(service = Service(ChromeDriverManager().install()))
```

```
time.sleep(5)

# Abre o site da Investing
driver.get("https://br.investing.com/currencies/eur-brl-historical-data")

# Aguarda 5 segundos para carregar o site por completo
#time.sleep(5)

# Clica no botão "aceito" do pop-up
while len(driver.find_elements(By.ID, "onetrust-accept-btn-handler")) == 0:
    time.sleep(1)
time.sleep(1)
driver.find_element(By.ID, "onetrust-accept-btn-handler").click()

# Clica no botão "Login"
while len(driver.find_elements(By.CLASS_NAME, "user-area_link__1RH0V")) == 0:
    time.sleep(1)
time.sleep(1)
driver.find_element(By.CLASS_NAME, "user-area_link__1RH0V").click()


## PREENCHENDO CAMPO DE LOGIN e SENHA ------------------------------------------------
try:
    # iframe é uma lista dos iframes presentes na página
    iframe = driver.find_elements(By.TAG_NAME, "iframe")
    # Percorrendo a lista de iframes
    for i in range(len(iframe)):
        # Mudando para o iframe "i"
        driver.switch_to.frame(iframe[i])
        # Procura o ID 'loginFormUser_email' no iFrame "i" e armazena na lista "teste"
        teste = driver.find_elements(By.ID, 'loginFormUser_email')
        # Se a lista estiver vazia:
        if not teste:
            driver.switch_to.default_content() # Retorne para a página inicial
            pass # passa para o próximo item do for (i + 1)
        # Agora, se a lista tiver o ID que procuramos:
        else:
            while len(driver.find_elements(By.ID, 'loginFormUser_email')) == 0:
                time.sleep(1)
            time.sleep(1)
            # Escreva o '[redacted]@gmail.com" no campo desejado
            driver.find_element(By.ID, 'loginFormUser_email').send_keys([redacted])
            # Como o campo de senha está no mesmo iframe do campo de email, escreva a senha:
            driver.find_element(By.ID, "loginForm_password").send_keys([redacted])
            # Clique no botão "Login":
            driver.find_element(By.CLASS_NAME, "newButton").click()
            break
except:
    time.sleep(4) # Aguarda carregar o site por 4 segundos
    iframe = driver.find_elements(By.TAG_NAME, "iframe")
    # Percorrendo a lista de iframes
    for i in range(len(iframe)):
        # Mudando para o iframe "i"
        driver.switch_to.frame(iframe[i])
        # Procura o ID 'loginFormUser_email' no iFrame "i" e armazena na lista "teste"
        teste = driver.find_elements(By.ID, 'loginFormUser_email')
        # Se a lista estiver vazia:
        if not teste:
            driver.switch_to.default_content() # Retorne para a página inicial
            pass # passa para o próximo item do for (i + 1)
        # Agora, se a lista tiver o ID que procuramos:
        else:
            while len(driver.find_elements(By.ID, 'loginFormUser_email')) == 0:
                time.sleep(1)
            time.sleep(1)
            # Escreva o '[redacted]@gmail.com" no campo desejado
            driver.find_element(By.ID, 'loginFormUser_email').send_keys([redacted])
            # Como o campo de senha está no mesmo iframe do campo de email, escreva a senha:
            driver.find_element(By.ID, "loginForm_password").send_keys([redacted])
            # Clique no botão "Login":
            driver.find_element(By.CLASS_NAME, "newButton").click()
            break
## ------------------------------------------------------------------------------------
time.sleep(5)

# Clica no botão "Baixar Dados"
```

```
# Caso ele não encontre o elemento, ele aguarda 1 segundo. Depois de encontrado,
# o elemento é clickado:
while len(driver.find_elements(By.CLASS_NAME, 'download-data_text__1gHMw')) == 0:
    time.sleep(1)
time.sleep(1)
driver.find_element(By.CLASS_NAME, 'download-data_text__1gHMw').click()

# Espera 4 segundos para realizar o download
time.sleep(4)

# Fecha navegador
driver.close()
```

2. Pathlib

In [ ]:

```
def descobrir_caminho_ultimo_arquivo(caminho, ext):
    '''
    Descobre o caminho do ultimo arquivo da pasta desejada
    - caminho: str. (caminho da pasta do arquivo)
    - ext: str. (extensão do arquivo. Ex: ext=".csv")
    '''
    # lista_arquivos recebe a lista de arquivos da pasta Downloads:
    lista_arquivos = os.listdir(caminho)
    lista_datas = []
    for arquivo in lista_arquivos:
        # descobrindo a data do arquivo:
        if ext in arquivo:
            data = os.path.getmtime(f"{caminho}/{arquivo}")
            lista_datas.append((data, arquivo))
    # Ordenando a lista de datas
    lista_datas.sort(reverse=True)
    # o ultimo arquivo será o primeiro item da lista de datas
    ultimo_arquivo = lista_datas[0]
    # ultimo arquivo retorna [data, arquivo] e, como
    # queremos o nome do ultimo arquivo, queremos ultimo_arquivo[1]
    nome_ultimo_arquivo = ultimo_arquivo[1]
    # Portanto, o caminho do ultimo arquivo pode ser descrito:
    caminho_ultimo_arquivo = caminho / Path(nome_ultimo_arquivo)
    return caminho_ultimo_arquivo


caminho_arquivo = descobrir_caminho_ultimo_arquivo("C:/Users/lucas/Downloads", ext=".csv")
caminho_atual = Path.cwd()

# Criando uma nova pasta no caminho atual chamada 'AutoAnalise':
if 'AutoAnalise' not in os.listdir(caminho_atual):
    caminho_pasta = caminho_atual / Path('AutoAnalise')
    Path(caminho_pasta).mkdir()
else:
    caminho_pasta = caminho_atual / Path('AutoAnalise')

# Movendo o arquivo para o caminho atual:
sh.move(str(caminho_arquivo), str(caminho_pasta))
```

Out[ ]:

```
'c:\\Users\\lucas\\OneDrive\\Documentos\\Engenharia Biomédica\\9º Período\\TCC\\AutoAnalise\\EUR_BRL Dados Históricos (9).csv'
```

3. Pandas

Tratamento dos Dados

In [ ]:

```
# Importando a Planilha e armazenando-a em 'dados_df'
dados_df = pd.read_csv(descobrir_caminho_ultimo_arquivo(caminho_pasta, ext=".csv"))

# Manipulando a string presente na coluna Data para obter Dia, Mês e Ano
lista_dia = []
lista_mes = []
lista_ano = []
for data in dados_df['Data']:
    lista_dia.append(data[:2])
    lista_mes.append(data[6:9])
```

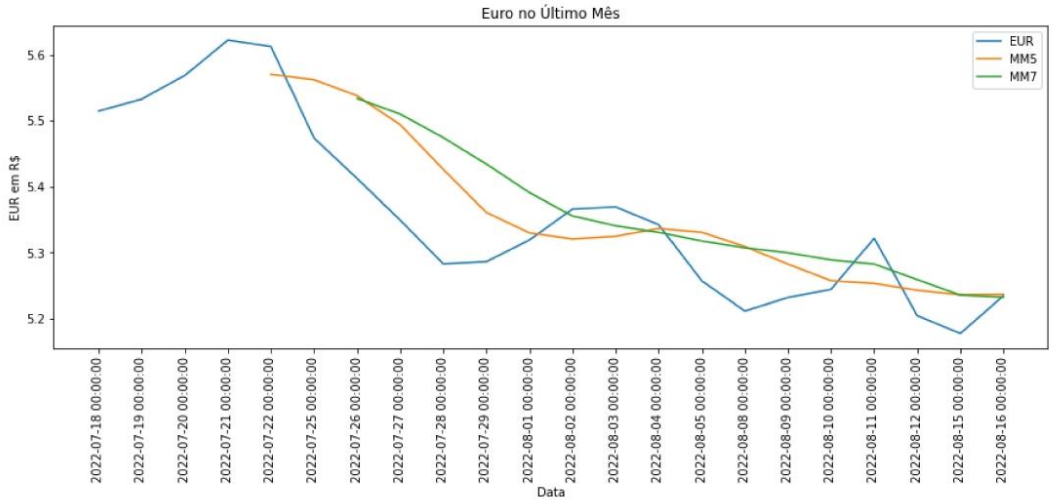
```
    lista_ano.append(data[-4:])

# Criando as colunas Dia, Mês e Ano:
dados_df['Dia'] = lista_dia
dados_df['Mês'] = lista_mes
dados_df['Ano'] = lista_ano

# Substituindo mês str por mês int:
dic_mes = {'jan': 1, 'fev': 2, 'mar': 3, 'abr': 4, 'mai': 5, 'jun': 6, 'jul': 7, \
           'ago': 8, 'set': 9, 'out': 10, 'nov': 11, 'dez': 12}
dados_df['Mês'] = dados_df['Mês'].replace(dic_mes)

# Transformando a coluna ano e dia em numeral
dados_df['Ano'] = pd.to_numeric(dados_df['Ano'])
dados_df['Dia'] = pd.to_numeric(dados_df['Dia'])

# Excluindo a coluna Data
dados_df = dados_df.drop('Data', axis=1)

# Transformando as colunas Dia, Mes e Ano em strings
dados_df['Ano'] = dados_df['Ano'].map(str)
dados_df['Dia'] = dados_df['Dia'].map(str)
dados_df['Mês'] = dados_df['Mês'].map(str)

# Criando uma nova coluna Data e unindo Dia/Mês/Ano
dados_df['Data'] = dados_df['Dia'] + '/' +  dados_df['Mês'] + '/' + dados_df['Ano']

# Transformando a coluna Data para o tipo 'DateTime'
dados_df['Data'] = pd.to_datetime(dados_df['Data'], format='%d/%m/%Y')

# Excluindo as colunas Dia, Mês e Ano:
dados_df = dados_df.drop(['Dia', 'Mês', 'Ano'], axis=1)

# Reordenando as colunas
dados_df = dados_df[['Data', 'Último', 'Abertura', 'Máxima', 'Mínima', 'Vol.', 'Var%']]

# Convertendo as colunas 'Último', 'Abertura', 'Máxima' e 'Mínima' para float
aux_df = dados_df[['Último', 'Abertura', 'Máxima', 'Mínima']]
aux_df = aux_df.replace(',', '.', regex=True ).astype(float)

# Tratando a coluna Vol (K = x1000 e passando para float)
dados_df['Vol.'] = dados_df['Vol.'].replace('K', '',\
     regex=True).replace(',', '.', regex=True).astype(float) * 1000

# Recriando a coluna Var% e transformando em float
dados_df['Var%'] = dados_df['Var%'].replace('%', '', regex=True)
var_list = []
for num in dados_df['Var%']:
    if '-' in num:
        num = num.replace('-','').replace(',','.')
        num = float(num)
        num = 100 - num
        var_list.append(num)
    else:
        num = num.replace(',','.')
        num = float(num)
        num = 100 + num
        var_list.append(num)

dados_df = dados_df.drop('Var%', axis=1) # Excluindo a antiga coluna Var%
dados_df['Var%'] = var_list

# Reordenando a tabela: dia mais antigo para dia mais recente
dados_df = dados_df.sort_values('Data')
dados_df = dados_df.reset_index(drop=True) # Resetando o índice

# Passando a planilha para o Excel:
dados_df.to_excel(caminho_pasta / Path('Dados_Analisados.xlsx'), index = False)
```

Análise dos Dados

In [ ]:

```
# Obtendo valores máx e min:
max = dados_df['Último'].max()
min = dados_df['Último'].min()
```

```
# Esboçando o gráfico da variação do Euro bem como as médias móveis 5 e 7
dados_df['Último'].plot(figsize=(15, 5), label = 'EUR', xlabel = 'Data', ylabel = 'EUR em R$')
dados_df['Último'].rolling(5).mean().plot(label='MM5')
dados_df['Último'].rolling(7).mean().plot(label='MM7')
plt.xticks(range(len(dados_df['Data'])), dados_df['Data'])
plt.xticks(rotation = 90)
plt.title('Euro no Último Mês')
plt.legend()
plt.savefig(caminho_pasta / Path('grafico_EUR.jpg'), bbox_inches = 'tight')

def alta_ou_queda():
    '''Calcula, com base na planilha dados_df se existe uma alta do Euro'''
    x = len(dados_df['Último'])
    if dados_df['Último'][0] > dados_df['Último'][x-1]:
        return 'Queda'
    else:
        return 'Alta'
```



4. Smtplib

In [ ]:

```
def enviar_email():
    caminho_planilha = str(caminho_pasta / Path('Dados_Analisados.xlsx'))
    caminho_grafico = str(caminho_pasta / Path('grafico_EUR.jpg'))

    # Criação do Corpo do E-mail
    corpo_email = """
    <p>Olá, Lucas! Tudo bom?</p>
    <br>
    <p>Segue a análise do Euro no último mês, como solicitado</p>
    <p>O euro esteve, aparentemente, em <b>{}</b> nesse último mês: \
        <b>máximo de R${:.2f}</b> e <b>mínimo de R${:.2f}</b></p>
    <p>Em anexo, segue o gráfico do euro neste mês contendo a média \
        móvel de 5 e 7 dias, bem como a planilha utilizada para a análise</p>
    <p>Qualquer coisa estou à disposição!</p>
    <br><br>
    <p>Att.</p>
    <p>Lucas Guaragna</p>
    <br>
    <p align='right'><font size='1'><i>este e-mail foi enviado automaticamente \
        utilizando <b>Python</b></i></font></p>
    """.format(alta_ou_queda(), max, min)

    # Criação de uma mensagem
    msg = EmailMessage()
    # Assunto da mensagem
    msg['Subject'] = "Análise EUR/R$"
    # Remetente
    msg['From'] = '[redacted]'
    # Destinatário
```

```
    msg['To'] = '[redacted]'
    # Senha - Gerada pelo google em "Senhas de App"
    password = '[redacted]'
    msg.add_header('Content-Type', 'text/html')
    # Corpo de e-mail
    msg.set_payload(corpo_email )

    #Anexando os arquivos:
    mime_type, encoding = mimetypes.guess_type(caminho_planilha)
    with open(caminho_planilha, 'rb') as fp:
        dados = fp.read()
    msg.add_attachment(dados, maintype=mime_type.split("/")[0], \
         subtype=mime_type.split("/")[1], filename='Dados_Analisados.xlsx')

    mime_type, encoding = mimetypes.guess_type(caminho_grafico)
    with open(caminho_grafico, 'rb') as fp:
        dados = fp.read()
    msg.add_attachment(dados, maintype=mime_type.split("/")[0], \
        subtype=mime_type.split("/")[1], filename='grafico_EUR.jpg')

    #
    s = smtplib.SMTP('smtp.gmail.com: 587')
    s.starttls()
    # Login Credentials for sending the mail
    s.login(msg['From'], password)
    s.sendmail(msg['From'], [msg['To']], msg.as_string().encode('utf-8'))
    print('Email enviado')

enviar_email()
```

```
Email enviado
```

In [ ]:

```
# Abrindo a mensagem final do programa:

janela = Tk()
messagebox.showinfo('Status do Programa', 'Programa concluído com sucesso! \
    Os arquivos enviados por e-mail se encontram na pasta AutoaAnalise, \
        localizada na mesma pasta do arquivo executável deste programa')
janela.destroy()
```

Fonte: Autoria Própria