UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

PABLO MARTINEZ SLOMP

DESENVOLVIMENTO DE UM SISTEMA EM MICROSSERVIÇOS PARA O CONTROLE E GERENCIAMENTO DE RPAS

GUARAPUAVA

2023

PABLO MARTINEZ SLOMP

DESENVOLVIMENTO DE UM SISTEMA EM MICROSSERVIÇOS PARA O CONTROLE E GERENCIAMENTO DE RPAS

Development of a Microservices System for RPA Control and Management

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Tecnólogo em Tecnologia em Sistemas para Internet do Curso Superior de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Paulo Henrique Soares

GUARAPUAVA

2023

PABLO MARTINEZ SLOMP

DESENVOLVIMENTO DE UM SISTEMA EM MICROSSERVIÇOS PARA O CONTROLE E GERENCIAMENTO DE RPAS

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Tecnológo em Sistemas para Internet do Curso de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná (UTFPR).

Data da aprovação: 22/Novembro/2023

Prof. Paulo Henrique Soares
Doutor
Universidade Tecnológica Federal do Paraná - Campus Guarapuava

---

Prof. Emerson André Fedechen
Doutor
Universidade Tecnológica Federal do Paraná - Campus Guarapuava

---

Prof. Luciano Ogiboski
Doutor
Universidade Tecnológica Federal do Paraná - Campus Guarapuava

GUARAPUAVA
2023

Quem nunca cometeu um erro nunca tentou
nada novo.
(Autor(a) desconhecido)

RESUMO

As automações de processos tornou-se algo essencial em linhas de produção para as empresas manterem-se competitivas no mercado. A diferenciação tem sido na robotização de procedimentos administrativos por meio de RPAs (Robotic Process Automation). A partir do momento que eles vão sendo criados e o número de tarefas automatizadas cresce, a controle sobre eles e sobre as informações que eles geram torna-se cada vez mais necessário. Este documento demonstra a estrutura de um sistema em microsserviços para o controle de RPAs e visualização de dados, dentro de uma empresa do ramo financeiro, de forma que possa ser escalável e adaptável ao crescimento de robôs desenvolvidos ao longo do tempo.

Palavras-chave: rpa; automação; microsserviço; desenvolvimento.

ABSTRACT

Process automation has become essential in production lines for companies to remain competitive in the market. Differentiation has been achieved through the robotization of administrative procedures using RPAs (Robotic Process Automation). As they are created and the number of automated tasks increases, control over them and the information they generate becomes increasingly necessary. This document outlines the structure of a microservices system for RPA control and data visualization within a financial company, aiming to be scalable and adaptable to the growth of robots developed over time.

Keywords: rpa; automation; microservice; developmet.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

Siglas

API	Application Programming Interface
BD	Banco de Dados
JSON	JavaScript Object Notation
ODM	Object Document Mapping
RPA	Robotic Process Automation
SGBD	Sistema de Gerenciamento de Banco de Dados

SUMÁRIO

1 INTRODUÇÃO

A formação de organizações se dá pelo relacionamento entre recursos, esforços e pessoas, as quais realizam tarefas diferentes com o intuito de alcançar objetivos específicos. Para que essa interação ocorra, a comunicação entre setores e indivíduos é primordial, seja ela visual, oral ou tátil. Com o objetivo de melhorar controle nas relações entre os envolvidos, é de vital importância as divisões de hierarquia e responsabilidades dentro da empresa. Essa disposição ocorre por meio de um organograma, o qual transmite a informação a respeito da incumbência de cada um dos envolvidos, bem como a divisão de trabalho entre eles. Além disso, com base nesses dados, cada colaborador sabe com quem se comunicar dependendo do processo que está sendo realizado (SOUZA, 2021).

Todavia, durante a troca de informações, podem ocorrer interferências vindas de diversos elementos, os quais são chamados de “causadores de distorções”. Os problemas oriundos das diferenças entre níveis de hierarquia, das falhas na linguagem, e do entendimento do que foi transmitido são alguns exemplos de perturbações que podem ocorrer na comunicação (SOUZA, 2021). Essas deturpações podem impactar na eficiência do trabalho, tanto no fluxo quanto em seu resultado.

Na busca por melhorias na realização de processos, empresas vêm há anos automatizando tarefas com a finalidade de se manterem competitiva no mercado. Esse cenário pode ser fortemente observado no segmento industrial, onde as empresas já demonstraram que é possível ter um maior controle sobre as etapas existentes, ganho de eficiência e tempo, além da possibilidade de extração de dados estatísticos e de desempenho, permitindo uma melhor avaliação dos processos (SANTOS BRUNA APARECIDA DE OLIVEIRA, 2017). Porém, não é apenas em linhas de produção industrial que essa mudança tem ocorrido. Empresas também têm buscado e selecionado atividades administrativas e rotineiras que possam ser automatizadas (MANZUETO, 2016).

O uso de softwares específicos, como um ERP ou um programa de controle e análise de dados, permitem que algumas etapas de um processo sejam executadas de maneira automática, tal como a geração de gráficos dinâmicos ou o cálculo proveniente de uma grande quantidade de dados. Contudo, se o objetivo é a automatização de várias etapas, diminuindo consideravelmente a intervenção humana, uma solução é a Automação Robótica de Processos (Robotic Process Automation (RPA)).

Uma RPA é um software que tem a capacidade de realizar tarefas rotineiras e que não necessitam de análises complexas, como se fosse uma pessoa, só que com mais precisão e velocidade (SILVA, 2018). Ele é capaz de interagir com sistemas on-line e offline, lendo e interpretando dados para os quais foi programado. Para que isso ocorra, é necessário que o robô seja parametrizado com as regras do processo, além dispor do acesso a uma máquina e a um usuário com as devidas permissões. Dentro deste contexto, uma RPA pode ser vista como um funcionário não humano dentro das rotinas e fluxos de trabalho. Contudo, ao se decidir imple-

mentar uma automação robótica de processo, é importante que sejam respondidas algumas questões, como: Qual a responsabilidade dela? Quem é o responsável por ela? Como se dará a comunicação entre as pessoas envolvidas e a automação?

Um sistema que consiga controlar as RPAs, filtrar e disponibilizar informações por elas geradas é de crucial importância para que a empresa saiba o que está acontecendo com os processos, e assim possa tomar decisões com base em dados existentes. A falta de uma aplicação que gerencie as questões mencionadas acima pode provocar uma série de problemas, tais como a execução de tarefas que já foram realizadas, a impossibilidade nas alterações de parâmetros do robô pela área responsável pelo processo, e a inviabilidade ou incremento de complexidade para a obtenção de informações pela gerência, pois todos os dados estariam exclusivamente dentro do algoritmo dos robôs.

Esse projeto ocorre dentro de uma empresa do ramo financeiro no interior do Paraná, a qual já possui algumas RPAs em funcionamento. Contudo, não dispõe de nenhuma ferramenta de controle e gerenciamento delas.

1.1 Considerações iniciais

Na instituição onde esse projeto foi realizado, todo o controle e informações referentes às RPAs está centralizado na área de desenvolvimento de automações da empresa. Se ocorrer qualquer falha ou um cenário que se diferencie do normal, apenas essa equipe terá o conhecimento sobre a situação. Dessa forma, mesmo que o robô esteja alocado em outra área, fica a cargo dos desenvolvedores verificar a situação de cada RPA, sobrecarregando o time de desenvolvimento com tarefas que saem do seu escopo. O intuito deste projeto é o desenvolvimento de uma aplicação em microsserviços que organize esses robôs, facilitando a interação deles com as pessoas envolvidas nos processos. Esse sistema será dividido em 3 módulos:

- **Controlador:** responsável pelo controle de todos os dados das tarefas que serão realizadas pelas RPAs, e pela interação com alguns sistemas de terceiros;
- **Serviço:** módulo acoplado à RPA, responsável por filtrar e armazenar os dados da tarefa a ser executada;
- **Visualizador:** uma aplicação web. A plataforma que expõe informações sobre as tarefas que foram ou serão realizadas de forma automatizada, e que permitirá o envio dados novos para a execução delas.

Toda a comunicação entre os módulos será realizada por meio de uma Application Programming Interface (API) bem definida e cada módulo será implementado dentro de uma máquina virtual (VM - *virtual machine*) e/ou de uma máquina física.

Por ser um projeto desenvolvido dentro de uma instituição financeira, este trabalho manterá sigilo sobre algumas informações, limitando a quantidade delas.

1.2 Objetivos

1.2.1 Objetivo geral

Desenvolver uma aplicação de microsserviços para o controle e gerenciamento de informações de RPAs utilizados por uma instituição financeira específica, localizada no estado do Paraná.

1.2.2 Objetivos específicos

1. Levantar os dados de entrada e saída dos RPAs
    a) Padronizar os dados de entrada e saída dos RPAs no módulo de serviço;
    b) Padronizar o envio e recebimento de dados no módulo de controle.
2. Desenvolver um Banco de Dados (BD) principal
    a) Centralizar o BD no módulo de controle;
    b) Criar uma API para a escrita e leitura do BD pelo módulo de visualização;
    c) Criar uma API para a escrita e leitura do BD pelo módulo de serviço.
3. Analisar a comunicação dos RPAs com aplicações de terceiros
    a) Analisar a necessidade de centralização da comunicação à aplicações de terceiros;
        i. Centralizar a comunicação com aplicações de terceiros ao módulo de controle.
4. Desenvolver o módulo de visualização
    a) Criar o *layout*;
    b) Criar um nome e logotipo.
5. Dimensionar o servidor
    a) Máquina física *versus* Máquina virtual;
    b) Escolher o sistema operacional.

1.3 Justificativa

O uso de RPAs para a diminuição no tempo de execução dos processos, assim como o ganho de assertividade, tem chamado a atenção de empresas por ser uma maneira de reduzir

os custos e manter-se competitiva no mercado. Contudo, a implementação de robôs sem o devido controle pode causar falhas na efetividade, as quais podem ocorrer, por exemplo, pela falta de transparência das informações geradas pelas RPAs.

Com a implementação deste sistema, espera-se que qualquer pessoa envolvida no processo tenha acesso às informações das tarefas autômatas, em execução ou programadas, ou ao histórico delas, além de disponibilizar informações para controle gerencial, tais como quantidade de processos automatizados e as datas e horas que foram finalizados. Outro ponto é a redução no número de requisições e acessos paralelos realizados em sistemas de terceiros, pois essas ficarão centralizadas no Controlador. Desta forma obtém-se um maior controle e transparência nas informações geradas por RPAs.

1.4 Estrutura do trabalho

Este trabalho está dividido da seguinte maneira:

- Capítulo 2: apresenta o cenário atual da empresa onde o projeto será desenvolvido;
- Capítulo 3: apresenta o panorama atual do mercado de automações;
- Capítulo 4: disserta sobre o referencial teórico
- Capítulo 5: descreve a metodologia de desenvolvimento;
- Capítulo 6: descreve o desenvolvimento;
- Capítulo 7: expõe os resultados do projeto e suas estruturas;
- Capítulo 8: apresenta as considerações finais.

2 ANÁLISE DE DOMÍNIO

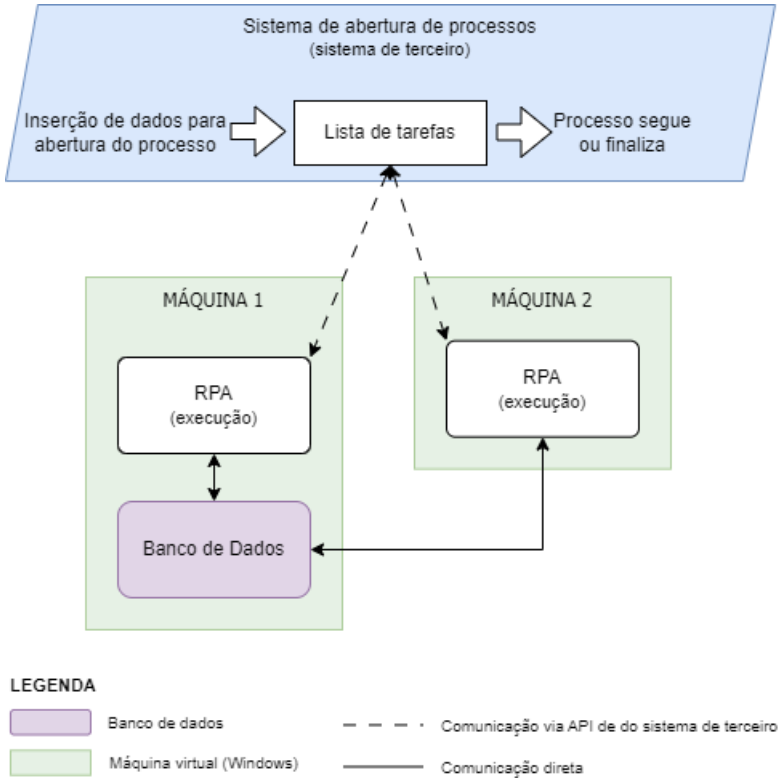
A empresa na qual esse projeto foi desenvolvido tem mais de 120 anos de história e está presente em todo o Brasil. Para conseguir atender todo o território de maneira eficiente, existem várias sedes em diversos estados. A sede na qual haverá a implementação deste projeto possui, atualmente, 40 anos de funcionamento.

No início do projeto, a sede possuía duas RPAs que eram responsáveis pela execução de 15 processos automatizados. Cada um dos robôs possui credenciais de acesso (usuário e senha), assim como os demais funcionários, para acessar os sistemas, além de um computador próprio, necessário para a interação com as aplicações, seja coletando, tratando ou enviando informações.

Sempre que uma nova automação entra em produção, é importante que a equipe de desenvolvedores esteja preparada para constantes manutenções e atualizações, já que é possível que um ou mais processos executados pelos robôs passem por alterações, seja no fluxo ou mesmo na aplicação necessária para sua realização. Devido a isso, os algoritmos das automações ficam disponíveis para toda a equipe de automação, dessa maneira é possível que refatorações no código sejam feitas em um ritmo sustentável e que atualizações possam ser integradas às RPAs continuamente.

A estrutura de funcionamento básica das RPAs, no início desse projeto, pode ser visualizada na Figura 1 e segue os seguintes passos:

Figura 1 – Estrutura atual de funcionamento dos RPAs



Fonte: Autor.

1. A tarefa é aberta dentro de um sistema de processos, em que são preenchidas todas as informações necessárias e onde ocorre o envio delas para a próxima etapa;

2. As RPAs verificam suas listas de tarefas, dentro do sistema citado no item acima, e as coletam, selecionam e armazenam os dados em um BD central;

3. As RPAs lêem o BD e selecionam uma tarefa para ser executada. Tal seleção ocorre com base na data e hora de salvamento das tarefas no BD, da mais antiga para a mais nova;

4. Com base na tarefa selecionada, as RPAs fazem uma requisição ao "sistema de abertura de processos" e coletam os dados contidos dela;

5. Com base nesses dados, são acessados os sistemas necessários, onde o processo é executados e algumas informações são armazenadas no BD;

6. As RPAs retornam ao "sistema de abertura de processos" e dão prosseguimento à tarefa, a qual pode seguir para outra área ou ser finalizada.

Como pode ser observado, o fluxo de funcionamento é simples e por mais que seja funcional, não é eficiente e muito menos acessível. Todas as informações geradas pelas RPAs estão no BD, onde o acesso é restrito. Apenas a equipe que desenvolve as automações tem acesso a esses dados. Desta maneira, apenas os desenvolvedores sabem o que está sendo feito, o quanto foi feito ou se houve falha ou não. Ou seja, se alguém da empresa necessitar de uma informação, deve solicitar aos desenvolvedores, e esses devem realizar essa busca. Isso resulta em uma sobrecarga de tarefas que estão além do escopo de trabalho dos programadores, pois eles precisam verificar continuamente a lista de tarefas automatizadas que foram executadas, validando se elas foram concluídas ou se ocorreu alguma falha. Diante deste cenário, é possível observar que além de desenvolver as RPAs, a equipe de automações fica responsável por todo o gerenciamento dos robôs, e das informações geradas por eles.

Ademais, manutenções e atualizações podem gerar grandes pausas nas execuções automáticas, pois se uma tarefa tiver passado por alteração em seu fluxo, deve-se acessar cada um dos robôs e pausar esse processo até que sejam realizadas as devidas mudanças. Após efetuadas, cada um deles é acessado novamente para que a tarefa seja colocada em produção outra vez. Enquanto são apenas duas RPAs, isso não aparenta ser um problema, mas o aumento nessa quantidade pode causar um gargalo na execução dos processos.

3 TRABALHOS RELACIONADOS

Já é natural as empresas automatizarem linhas de produção com a utilização de robôs e/ou máquinas (SANTOS BRUNA APARECIDA DE OLIVEIRA, 2017), mas a competitividade tem impulsionado a busca pela automatização de processos administrativos, na tentativa de reduzir os custos e aumentar a eficiência e assertividade, resultando em uma vantagem competitiva. Essa demanda resultou em um crescimento de 31% mundialmente em 2021 no mercado de RPA, e espera-se um aumento de mais de 17% para 2023. Com o objetivo de aproveitar esse crescimento, companhias desse setor têm investido cada vez mais no desenvolvimento das suas aplicações de automação (STAMFORD, 2022).

Cabe ao futuro cliente dessas companhias definir qual escolher, visto que, sendo criados processos automatizados que se utilizam a solução de uma dessas empresas, não é possível adquirir um módulo de controle ou de visualização de informações de outra. Isso ocorre pelo fato que os softwares de criação de RPA são desenvolvidos para funcionarem exclusivamente com os módulos que os compõem. Tal fato se origina da quantidade de variáveis que existem ao se desenvolver uma automação, tornando-se praticamente impossível a criação de subsistemas genéricos que funcionem para diferentes aplicações de desenvolvimento RPA.

3.1 Mercado de RPA

No cenário atual, CXTODAY (2022) cita 15 empresas integrantes nesse mercado por meio do Quadrante Mágico do Gartner[1] (Figura 2), no qual 5 delas são consideradas líderes de mercado. Dessas, são três as estão em evidência atualmente.

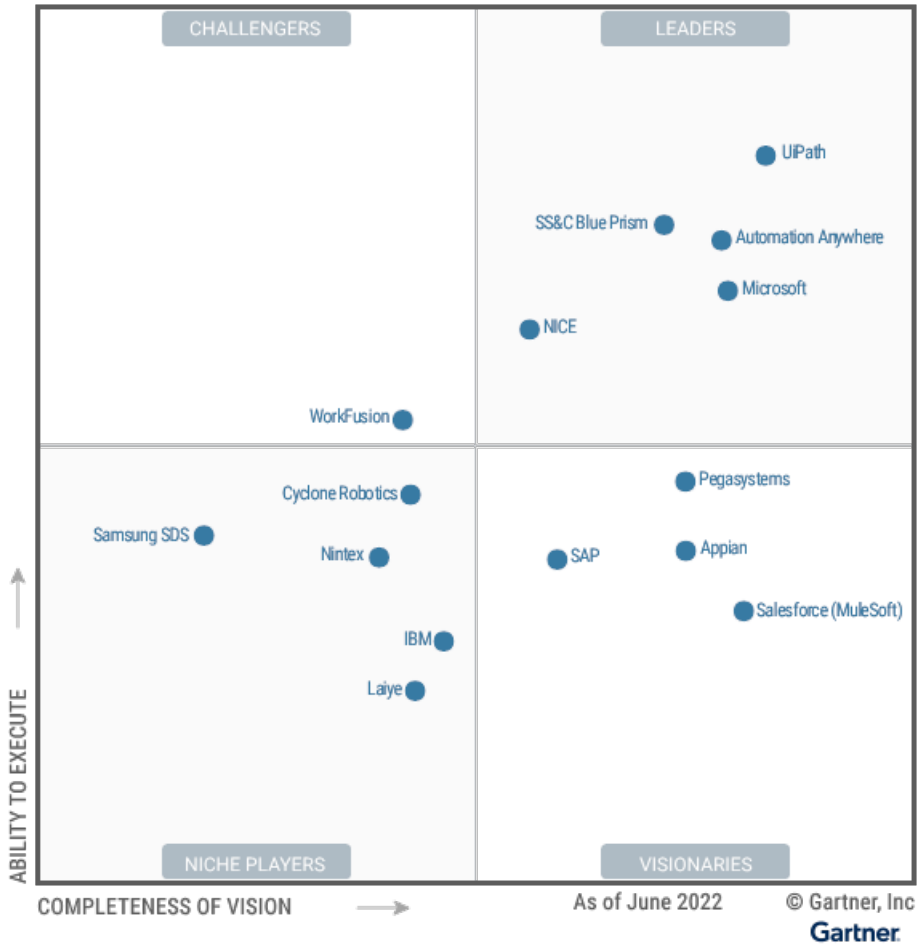
3.1.1 UIPath

Empresa fundada na Romênia em 2005, é a líder do segmento (CXTODAY, 2022). Possui um ecossistema diversificado, com diversas integrações e bibliotecas próprias, possibilitando sua aplicação nos mais diferentes segmentos empresariais (UIPATH, 2023). Sua interface por meio de blocos possibilita que pessoas sem conhecimento aprofundado em programação consigam automatizar tarefas simples.

Para o controle dos RPAs desenvolvidos por meio da plataforma, o UiPath possui um sistema on-line no qual é possível gerenciar as automações, verificando se estão em funcionamento, agendando tarefas, controlando versões. Uma limitação ocorre no fato que essa plataforma funciona plenamente apenas em RPAs que são executados de maneira automática,

[1] *Magic Quadrant do Gartner*, é um gráfico que exibe o posicionamento competitivo entre empresas de desenvolvimento de um mesmo mercado, dividindo-as em Líderes, Visionários, Operdores de Nicho e Competidores (GARTNER, 2023).

Figura 2 – Quadrante Mágico do Gartner



Fonte: CXTODAY (2022).

os quais possuem um valor de licença maior do que os que necessitam de uma inicialização manual.

3.1.2 Automation Anywhere

Empresa americana, fundada em 2003, tem como seu principal diferencial o foco em nuvem, podendo ser implantado tanto, por exemplo, no Google Cloud e no AWS (CXTODAY, 2022). Está presente em mais de 90 países e mais de 2,8 milhões de RPAs implementados (ANYWHERE, 2023).

Sua plataforma de gerenciamento permite a inserção de dados financeiros sobre a mão de obra que está sendo economizada com a automação, e em conjunto com o tempo de execução de cada automação, é possível a visualização, por exemplo, de informações sobre a economia gerada e do tempo ganho devida a rapidez da execução, por meio dos RPAs.

3.1.3 Power Automate

Pertencente à Microsoft, surgiu em resposta ao mercado. Anteriormente a empresa tinha apenas o Power RPA, que era incorporada a sua plataforma em nuvem Azure. O Power Automate possui uma versão gratuita para todos os usuários de Windows 10 e 11 (CXTODAY, 2022). Por pertencer à Microsoft, ele possui uma poderosa integração com todos os softwares dela, fazendo disso um grande diferencial em relação aos seu concorrentes.

A visualização e controle de informações do Power Automate é bem completa, trazendo informações como pontos de gargalos, gráficos sobre tempo e verificação de pontos de melhoria (AUTOMATE, 2023). Contudo, é importante salientar que determinadas funções ficam disponíveis apenas em planos pagos.

4 REFERENCIAL TEÓRICO

Este capítulo apresenta as definições necessárias para um melhor entendimento do projeto. Primeiramente é descrito sobre *Robotic Process Automation* (RPA), passando pelo conceito de microsserviço e sua comparação com um sistema monobloco, para então descrever alguns detalhes sobre a linguagem Python o Sistema de Gerenciamento de Banco de Dados (SGBD) MongoDB.

4.1 RPA - *Robotic Process Automation*

A automação robótica de processos pode ser descrita como a utilização de um ou mais robôs virtuais, que executam as tarefas para as quais foram programados, podendo realizá-las utilizando-se da mesma interface utilizada pelos humanos. É possível abrir, interagir, ler e enviar informações para quaisquer sistemas que se tenha acesso (SILVA, 2018). Em outras palavras, é um software previamente configurado para imitar a ação humana em tarefas repetitivas e estruturadas (JúNIOR, 2021). Esses dois pontos são cruciais, pois, ainda, não é possível para os robôs julgar situações e tomar decisões com base nelas. De acordo com Cabral Ariel Behr (2022) as melhores tarefas para serem automatizadas possuem as seguintes características:

- Baixo requisitos cognitivos;
- Alto volume;
- Acesso a muitos sistemas;
- Baixo índice de exceções;
- Alto índice de falha humana.

Porém, a principal motivação para a implementação de RPAs está na otimização dos custos (CABRAL ARIEL BEHR, 2022). Devidamente projetado, um robô de processo pode executar uma tarefa de maneira muito mais rápida e assertiva se comparado com um ser humano. Contudo, para que se desfrute dos benefícios, é necessário que haja um alinhamento bem estruturado dentro da empresa para que ocorra a implementação de um RPA.

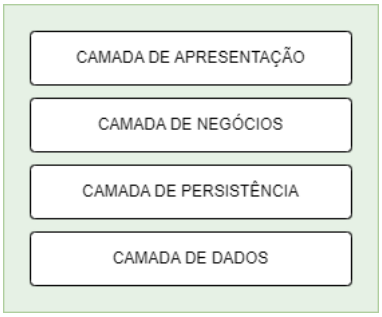
Como observado por MARIANO (2021), automações não devem ser desenvolvidas sem a devida orientação da equipe de TI, pois isso pode ocasionar problemas de segurança e desempenho, impactando, por exemplo, na rede da empresa. Outro ponto importante é a necessidade de documentação, item necessário para a realização de auditorias internas e/ou disseminação das informações presentes no algoritmo.

4.2 Microsserviços

Antes de se desenvolver uma aplicação, é importante elaborar sua estrutura. Quantos e quais os elementos que deve conter, como vão se comunicar entre si e quais são as propriedades de cada um deles (MEDES, 2021), são exemplo que dados que devem ser devidamente arquitetados. Dentro dessa ordenação, dois padrões se destacam: monolítico e microsserviços.

Um sistema monolítico caracteriza-se por conter toda sua base de código compreendida em um só lugar, e os dados que o alimentam ficam centralizados em uma base compartilhada (JúNIOR, 2019). Ou seja, todas as camadas que o compõem fazem parte da mesma estrutura (Figura 3). Ele pode conter um único ou vários processos, e, no segundo caso, todos os processos precisam ser implementados juntos.

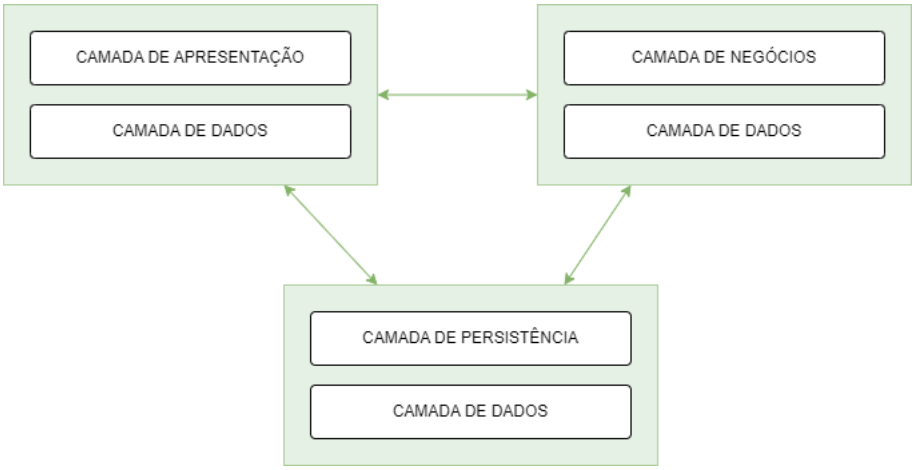
Figura 3 – Arquitetura monolítico



Fonte: (JúNIOR, 2019) - Editada.

Por ser altamente acoplada, uma aplicação monolítica não é facilmente escalável. Diferente da sua criação, que tende a ser de fácil desenvolvimento, cada alteração realizada em uma parte do código pode impactar no todo, tornando sua ampliação uma tarefa custosa.

Figura 4 – Arquitetura de microsserviços



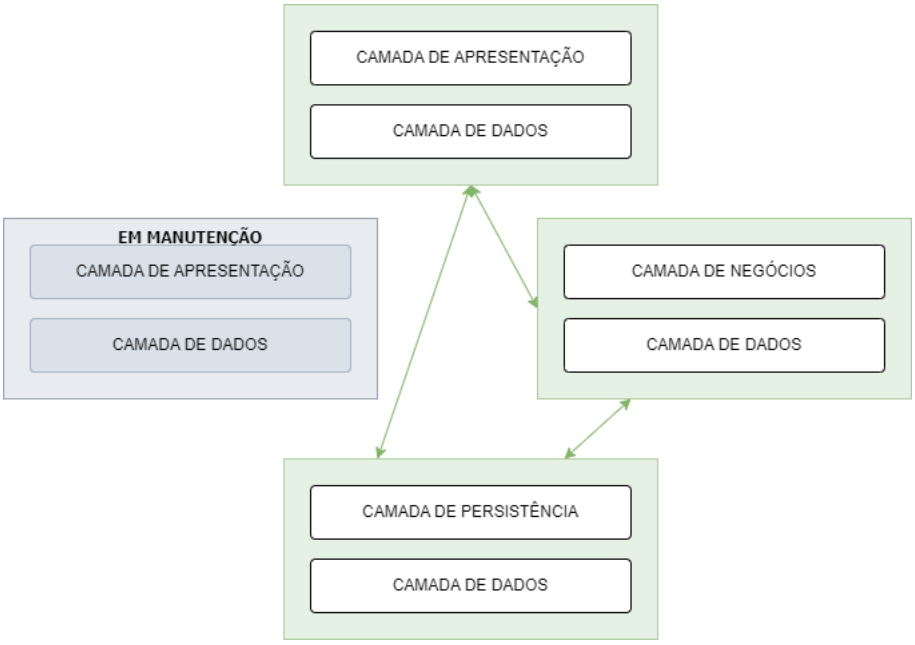
Fonte: Autor.

Por outro lado, um sistema em microsserviços tem suas camadas independentes entre si. A AWS define microsserviços como "uma abordagem arquitetônica e organizacional do desenvolvimento de software na qual o software consiste em pequenos serviços independentes

que se comunicam usando APIs bem definidas". Em outras palavras, essa arquitetura é composta por pequenas partes autônomas que se comunicam entre si para realizarem a entrega (Figura 4).

Diferentemente da estrutura monolítica, um sistema de microsserviços é altamente escalável. Como cada módulo trabalha de maneira independente, é possível realizar upgrades em um deles sem impactar os outros. Isso é devido a capacidade de replicação de cada um dos serviços. Por exemplo, se for necessário fazer alguma alteração na camada de negócios, é possível replicá-la, mantendo-a em funcionamento enquanto se modifica a cópia. Isso sem interferência aos demais serviços (Figura 5).

Figura 5 – Manutenção em microsserviços



Fonte: Autor.

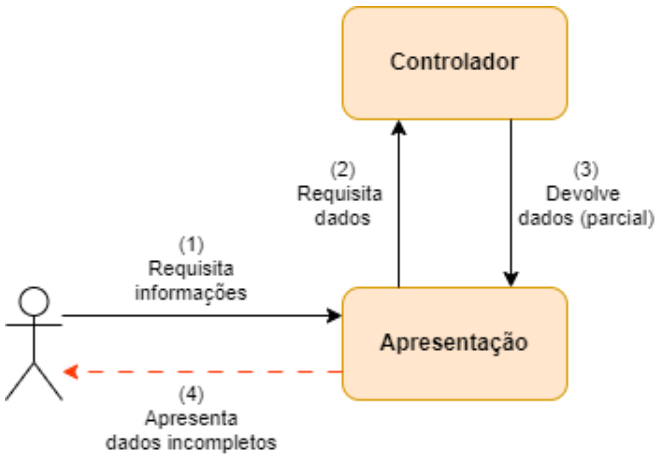
Contudo, para um funcionamento fluído, a comunicação entre os serviços deve ser coesa (JúNIOR, 2019). Mesmo que cada camada realize suas tarefas individualmente, é necessário que haja trocas de dados entre elas para que o objetivo do processo seja alcançado. É impossível que a camada de apresentação exiba os dados que foram requisitados se o controlador não realizou o envio correto dos dados (Figura 6).

4.3 Python

A linguagem Python é considerada uma linguagem democrática, muitas vezes sendo usada como primeiro contato à programação em escolas (DIVINO, 2021). Um dos motivos que contribui para isso é o fato dela ser flexível, facilitando o aprendizado e garantido ao desenvolvedor diferentes maneiras de programação:

1. Procedural, onde código é executado na sequência na qual foi criado;

Figura 6 – Envio de dados



Fonte: Autor.

2. Orientada a objetos, a qual é organizada em estruturadas denominadas classes e objetos, permite programação multiplataforma;

3. Funcional, que organiza em blocos chamados de funções, as quais realizam tarefas específicas.

Outro ponto que merece destaque, é o fato dela ser muito reconhecida na área de análise de dados, possuindo diversas bibliotecas voltadas para esse cenário (DIVINO, 2021). Isso a torna super versátil quando se trabalha com uma grande quantidade de dados. É possível encontrar soluções para os mais diversos cenários, entre as milhares bibliotecas disponíveis na linguagem, tais como:

- Rocketry: framework para agendamento de tarefas e processos, altamente customizável;
- Pandas: provavelmente a biblioteca mais famosa do Python, o Pandas é uma poderosa ferramenta de análise e manipulação de dados;
- Litestar: biblioteca para a criação de APIs tanto síncronas quanto assíncronas;
- PyAutoGUI: possui scripts responsáveis por simular o uso do mouse e teclado, de maneira que seja possível a interação com outras aplicações;
- Beanie: é um ODM assíncrono para o MongoDB

4.4 MongoDB

Os bancos relacionais são os mais conhecidos, principalmente por implementarem o modelo ACID, que é a sigla para atomicidade, consistência, isolamento e durabilidade. Esses

quatro pilares asseguram que os dados continuarão consistentes após a execução de uma transação (SANTOS, 2019).

Contudo, quando essas características não são necessárias, mesmo que haja relação entre diferentes grupos de dados, o uso de SGBD relacionais se torna ilógico. Para esse cenário existem os bancos NoSQL. Estes tem como sua principal característica o armazenamento feito por meio de documentos, os quais normalmente utilizam a formatação JavaScript Object Notation (JSON) (LIMA, 2021). Cada um desses documentos pode ser diferente dos outros, e, mesmo assim, todos podem ser armazenados em um mesmo local. Esse local, definido como ‘tabela’ em um modelo relacional, é conhecido como *collection*.

5 METODOLOGIA DE DESENVOLVIMENTO

A ênfase deste capítulo está em reportar o que foi realizado para alcançar o objetivo do trabalho. Este capítulo está subdividido em duas seções, sendo uma para os materiais e outra para os métodos.

5.1 Materiais

Analisando-se as informações dos RPAs existentes na empresa, e os dados que são e/ou devem ser armazenados, dois pontos importantes para o desenvolvimento foram decididos: a linguagem de programação e o modelo de banco de dados.

Quanto à linguagem, foi escolhida a Python. Essa decisão foi julgada como a melhor pelas seguintes questões:

1. Atualmente o Python é usado pelo time que desenvolve às automações dentro da empresa. Por esse motivo, não é necessário que as pessoas que o compõem precisem aprender uma outra linguagem de programação, o que facilita a implementação do projeto, bem como a comunicação entre o sistema e os RPAs;

2. Possui uma sintaxe de fácil aprendizado. Isso pode facilitar futuras contratações pela empresa, a qual pode vir a aceitar pessoas que conheçam outras linguagens, pois estas podem aprendê-la com relativa rapidez;

3. Como o sistema vai ter múltiplos serviços, é possível que eles precisem trabalhar em estruturas de código diferentes, e o Python permite ao programador desenvolver códigos de modos distintos, tal como o procedural e orientado a objetos;

4. Dentro da empresa utilizam-se diversos relatórios, e muitos deles podem ser importantes para que a execução de tarefas seja realizada mais rapidamente pelos robôs. Para que as informações nelas contidas sejam enviadas, é importante que os dados estejam filtrados e organizados. Deste modo, o Python se destaca por ter várias bibliotecas que facilitam esse trabalho.

No que diz respeito ao modelo de BD, atualmente as RPAs utilizam o SQL, por meio do MySQL. Contudo, foi analisado que uma das principais características desse tipo de BD não está sendo e nem precisa ser utilizada: a de dependências e relacionamento entre os dados. Como as informações que serão armazenadas são apenas de transferência, ou seja, estão em um módulo, que precisa repassá-las para outro, tradas e depois enviadas para seu destino, não há necessidade de todos os benefícios o que um modelo de BD relacional disponibiliza. Outro ponto é que as quantidades e os tipos de dados podem ser diferentes e, mesmo assim, é interessante que sejam armazenados em um mesmo local, ou seja, é necessário que o SGBD

seja flexível, possibilitando reunir diferentes informações com diferentes dados em um mesmo local.

Em decorrência a esse cenário, optou-se pela utilização de um SGBD NoSQL, mais precisamente pelo uso do MongoDB. Visto que a maioria das informações que se necessita armazenamento são referentes às APIs de comunicação entre os módulos do sistema, as quais são dados organizados em um JSON, é natural que elas sejam acondicionadas nesse mesmo formato, não havendo necessidade de conversões e/ou formatações.

5.2 Métodos

O presente estudo caracteriza-se pela coleta de dados a partir de uma análise observatória e documental das RPAs existentes na empresa onde este projeto foi realizado.

Para que desenvolvimento do sistema seja viável, as seguintes informações foram coletadas:

1. **Informações utilizadas pelas automações:** quais são as informações mais frequentemente utilizadas;
2. **Tipos de automação:** quais são os tipos de automações possíveis e existentes;
3. **Aplicações acessadas:** quais são as aplicações de terceiros que as RPAs acessam, tal como de qual tipo são (web, embarcado, remoto...), além do levantamento das mais acessadas;
4. **Informações dos RPAs:** quais são as informações que a empresa deseja dispor sobre os robôs e sobre as tarefas que eles realizam;
5. **Padrão de Armazenagem de Dados:** se existe um padrão de como são dados os resultados dos processos executados pelos RPAs;
6. **Inicialização dos Robôs:** como é realizado o start de cada processo executado de forma automatizada;
7. **Frequência de Atualizações dos RPAs:** com que frequência as RPAs são paralisadas em decorrência de uma atualização ou manutenção;
8. **Recursos Disponíveis:** os recursos que a empresa disponibiliza para a realização e implementação do desenvolvimento do sistema proposto neste projeto;

Na elaboração da estrutura entre os módulos do sistema de microsserviços (Controlador, Serviço e Visualizador) e na elaboração dos esquemas das APIs, é essencial realizar o levantamento de quais as informações que devem ser salvas e quais delas deverão transitar entre os serviços. Em vista disso, foram realizados dois tipos de investigações:

- Acompanhamento de processos automatizados: durante várias semanas, foram observadas diversas tarefas autônomas, com o objetivo de captar as informações demandadas necessárias para serem realizadas. Em paralelo foram coletados quais dados são disponibilizados após a finalização de cada tarefa. Com base nos resultados dessas observações, foi possível analisar quais são as informações comuns entre os diferentes processos.

- Projeção de futuras automações: baseando-se em conversas com diferentes áreas da empresa e em uma análise geral dos trabalhos que são realizados dentro de cada área, optou-se por idealizar cenários futuros. O objetivo dessa projeção foi dimensionar as informações das APIs, para evitar muitas modificações futuramente.

Com base nos dados que devem compor a API, é possível prever como a comunicação entre os serviços ocorrerá, e quais os filtros cada um deverá possuir para que os dados contidos na API sejam enviados, lidos e devolvidos de maneira eficiente.

5.3 Arquitetura Base do Sistema

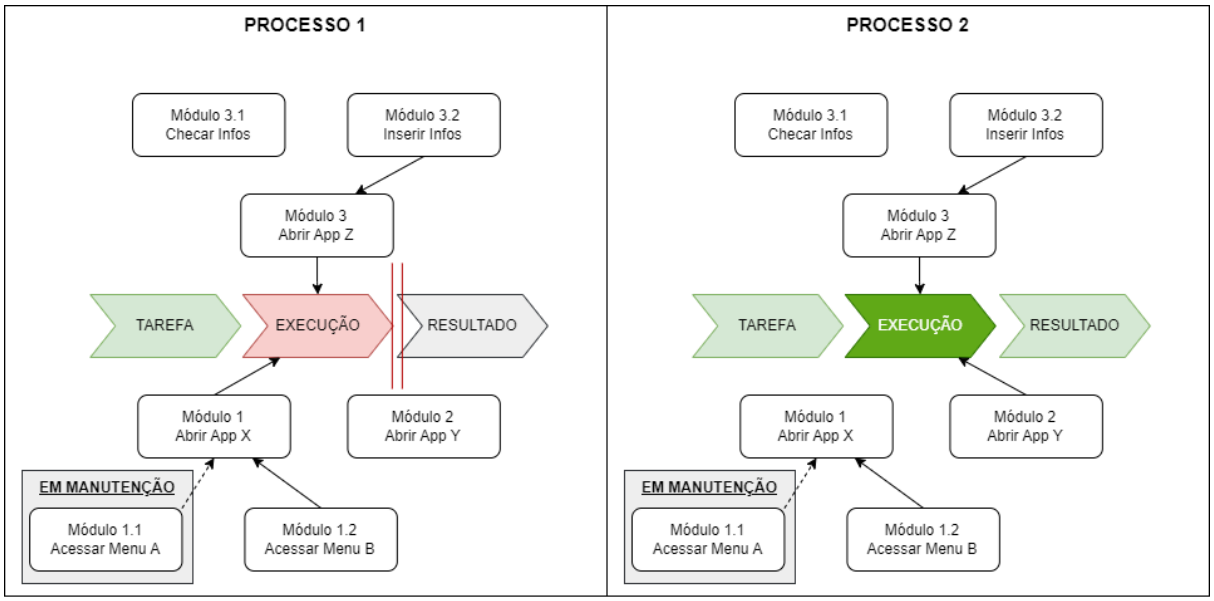
O módulo de controle deve ser o primeiro a ser implantado. Por ele ser o centro do sistema de microsserviços, é fundamental que sua estrutura esteja construída de maneira que defina as APIs de envio e recebimento de informações, que serão utilizadas pelos módulos de serviço e visualização. É importante salientar que uma vez iniciado o desenvolvimento, ele deverá ser flexível o suficiente para se adequar a particularidades dos RPAs, ao mesmo tempo que determina o padrão de fluxo de informações entres os módulos. Assim que forem englobados os vários cenários de processos e automações diferentes, os futuros robôs desenvolvidos deverão seguir o modelo concebido de leitura e escrita das APIs entre os módulos.

Ocorrendo constantes mudanças em processos, aplicações e/ou RPAs, torna-se crucial a elaboração de pequenos módulos de códigos, os quais são chamados por um outro maior, responsável por estruturar e organizá-los. Dessa forma, na ocorrência de manutenções, pode-se realizá-las apenas em determinados módulos, evitando a interferência em outros processos.

A figura 7 exemplifica esse cenário: O processo 1 precisa acessar o Menu A do Módulo 1 para ser executado, contudo, esse está em manutenção. Já o processo 2 não faz uso desse módulo, então pode continuar funcionando normalmente. Havendo a necessidade de reescrever um código, é possível reaproveitar todos os outros módulos que o compõem, reformulando-se apenas o(s) necessário(s). No caso da figura, caso seja necessário reescrever o Módulo 3, se for feito o uso dos seus submódulos, contribui-se para um resultado mais rápido.

Visto isso, é importante que a estrutura interna de cada parte do sistema de microsserviços seja composta de módulos, possibilitando a realização de manutenções e incrementos, evitando o impacto em outros serviços que estejam sendo executados.

Figura 7 – Desenvolvimento em módulos



Fonte: Autor.

6 DESENVOLVIMENTO

Esse capítulo descreve a estrutura do sistema, bem como o relacionamento entre os módulos dos serviços e o esquema base dos dados (API) a ser transitado entre eles.

6.1 Estrutura do Sistema

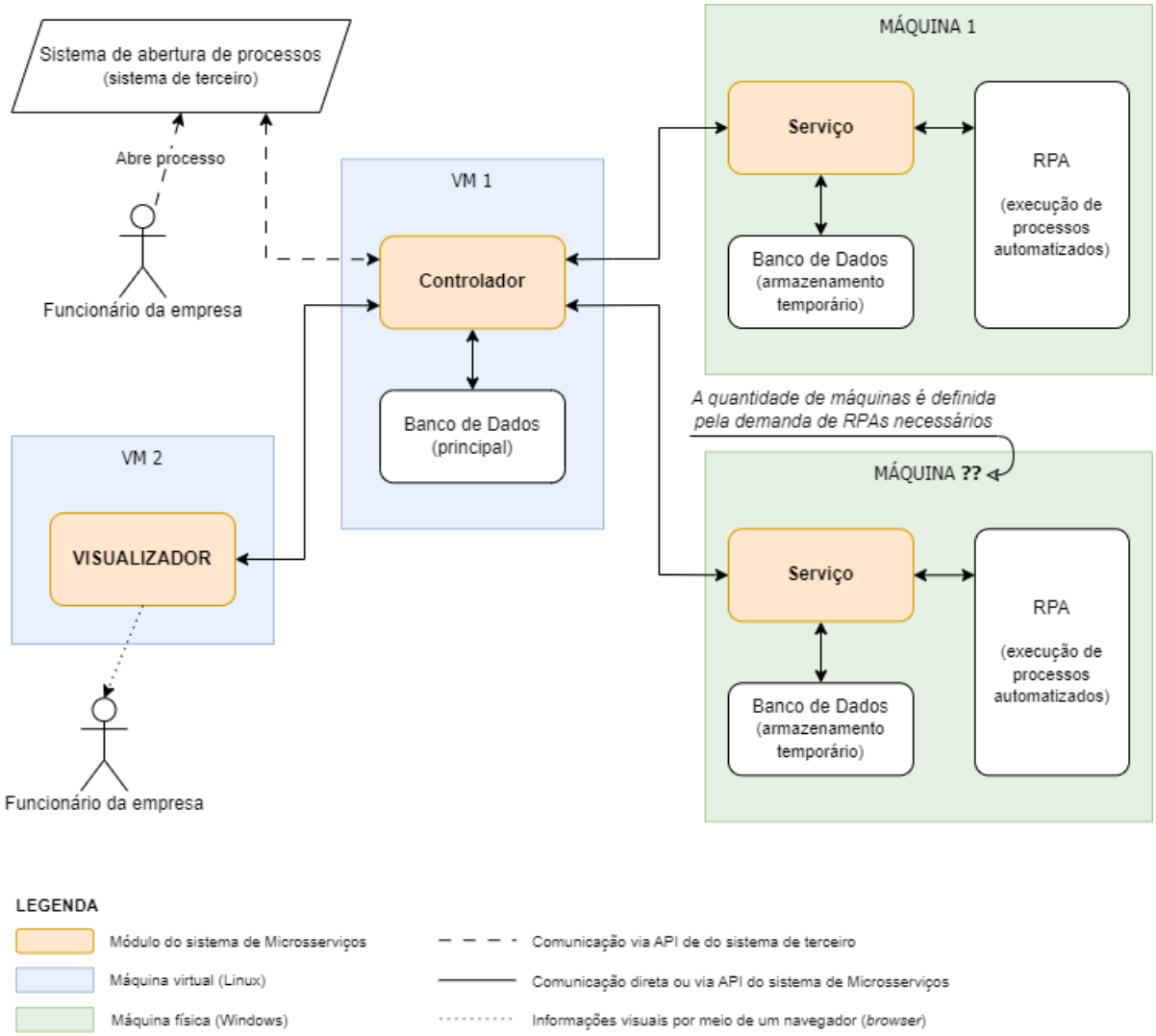
Analisando a estrutura da empresa, bem como os recursos e o atual funcionamento das automações, chegou-se à configuração exibida na figura 8. Com base nela, é possível visualizar os seguintes itens:

- Os módulos Controlador e Visualizador, por não necessitarem de estrutura gráfica nem grande desempenho, serão alocados em máquinas virtuais dentro do servidor da empresa;
- Haverá um BD central, o qual apenas o Controlador tem acesso;
- O módulo de serviço será implementado em todas as máquinas onde houver um RPA;
- Diferentemente das máquinas dos robôs, os módulos Controlador e Visualizador podem ser implementados em com Linux;
- A comunicação com o sistema responsável pela abertura de processos ficará a cargo do Controlador;
- Cada módulo de serviço terá um BD próprio, que servirá para armazenamento de informações necessárias apenas para o processo corrente.

Devido ao fato do Controlador ser o módulo central, o desenvolvimento deverá começar por ele, pois nele serão definidas as regras de como serão estruturadas as informações de envio e recebimento da API. É nele que as informações serão filtradas para o envio ao módulo de Serviço para cada tipo de operação automatizada. Dito isso, deverá ser necessário o controle das tarefas, tais como, por exemplo: qual será enviada antes, qual está pendente ou concluída e qual robô está livre para recebe-la.

Dentro da empresa existem dois tipos de tarefas que são automatizadas: as que são agendadas e as que se iniciam pelo sistema de apertura de processos. No primeiro caso, podem existir uma tarefa que deve ser executada todo dia 10 e outra todos os dias às 9 horas. Ficará a cargo do Controlador realizar esse agendamento e enviar elas para execução no dia e hora corretos. Outro ponto é que uma tarefa não pode interferir na outra. Se uma não puder ser executada, seja por estar em manutenção ou algum problema no código, isso não pode impedir que outra seja enviada para o módulo de Serviço. Da mesma maneira que as tarefas não podem impactar no envio de informações para o módulo de Visualização. Por esse motivo,

Figura 8 – Configuração dos módulos do sistema de microsserviço

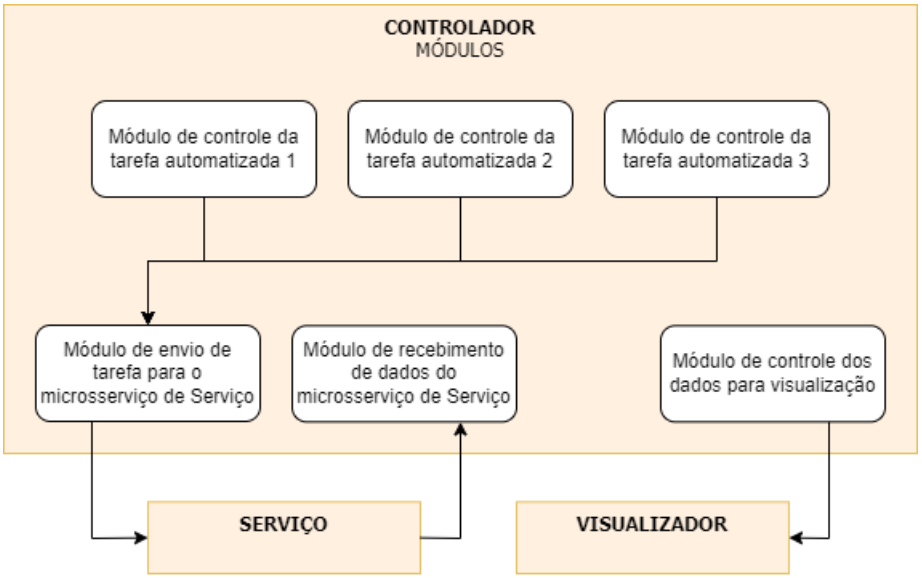


Fonte: Autor.

cada estrutura responsável por uma atividade específica dentro do controlador deverá ser independente das outras, como mostrado na figura 9. Com base nela, se o "módulo de controle da tarefa automatizada 1"estiver em manutenção, não vai impactar nas outras tarefas. O Mesmo pode ser visto para o que envia os dados para a visualização. As exceções estão nos códigos que controlam o envio e o recebimento do módulo de serviço, já que esses são em comum a todas as tarefas. Se caso ele estiver em manutenção, significa que realmente nenhum tarefa poderá ser enviada ou recebida, por motivo de segurança.

Outro ponto é a comunicação com a aplicação de abertura de processos. Esse programa é um software de terceiro, que possui uma API própria. Para que cada robô não fique acessando ele, gerando trafego em paralelo e impactando no desempenho do mesmo, optou-se que apenas o Controlador fizesse essa comunicação. Desse modo, ele que acessa e coleta das tarefas designadas para os RPAs, bem como faz o protocolo delas quando forem finalizadas pelos robôs.

Figura 9 – Estrutura do Controlador



Fonte: Autor.

A decisão de centralizar o BD foi tomada para uma melhor organização dos dados. Caso seja necessário informações sobre um determinado tipo de tarefa realizada, não é necessário qual robô fez cada uma e ver o BD deles. Centralizando no Controlador, fica fácil a manutenção e a obtenção de dados, bem como o *backup* deles.

Os módulos de **Serviço** terão como função o controle e filtro de informações enviadas para o RPA, sejam elas vindas do Controlador ou do BD próprio. Por esse motivo, haverá tantos quanto o número de RPAs existentes. Dessa maneira, tem como manter os dados coesos e organizados, além de sempre devolvidos ao controlador ou enviados para o RPA de maneira correta.

Quanto ao BD próprio, cada Serviço terá acesso a um. Isso é necessário para que as informações não sejam perdidas no decorrer de uma automação. Isso pode ser necessário em casos do processo ser muito longo e o RPA precise armazenar dados durante ele, ou na situação em que ocorrer algum problema durante a execução do robô. Ao ser reexecutado, é possível que continue o processo do ponto que parou.

O módulo **Visualizador** terá como função a exibição de informações para a empresa por meio de um navegador. A projeção do seu layout pode ser vista na figura 10. Seus dados virão diretamente do Controlador, o qual vai acessar o BD e enviar as informações requisitadas pelo Visualizador. O tipo e a quantidade de informações disponíveis dependerão da demanda da empresa, e serão organizadas por área e/ou tipo.

Todo o sistema deverá ser de fácil manutenção e ampliação, pois a cada nova automação os módulos deverão ser atualizados para recebe-la. Outro ponto é que deverão ser de fácil migração para o caso de uma máquina der problema. Ocorrendo isso, e havendo outra máquina disponível, os módulos serão implementados nessa nova, já podendo se comunicar entre si com mínimas alterações no código.

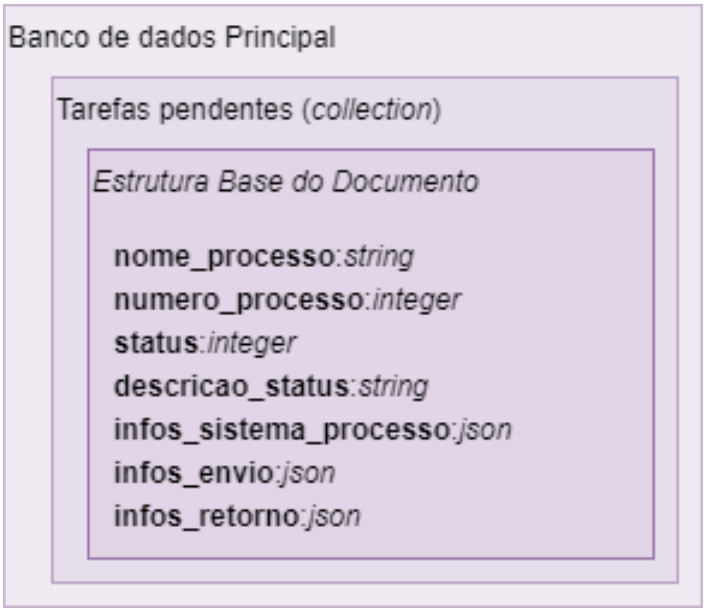
Figura 10 – Projeção do layout do módulo Visualizador

Fonte: Autor.

6.2 Esquema Base da API

Cada automação pode necessitar de diferentes dados para que sejam executadas, bem como podem gerar elementos distintos após sua execução. Devido a isso torna-se essencial que as informações enviadas pela API para Entrada e Saída do módulo de Serviço possam ser dinâmicas. O SGBD MongoDB colabora com esse cenário, pois os documentos dentro de uma mesma *collection* não precisam serem todos iguais. Na figura 11 é mostrado uma organização base de chaves para os documentos, para que haja a possibilidade de armazenar dados diferentes sem divergir de uma estrutura em comum.

Figura 11 – Estrutura base dos documentos do BD



Fonte: Autor.

Dentre as chaves, três requerem uma descrição:

- **infos_sistema_processo:** contém as informações vindas do sistema de terceiro onde são abertos os processos, tais como hora em que a tarefa foi aberta, quem fez a requisição e para onde deverá ser enviada após a conclusão;
- **infos_envio:** informações relacionadas à tarefa, para que sejam possível ela ser executada. Um exemplo é um processo de contratação de cartão, onde é necessário saber qual a bandeira do cartão.
- **infos_retorno:** os dados que foram gerados pelo RPA após a execução.

Essas três chaves citadas acima são do tipo *json*, o que as tornam dinâmica. Ou seja, é possível modificar o conteúdo delas com base na tarefa que está sendo executada. Isso simplifica os filtros dos dados nos módulos Controlador e de Serviço, ao mesmo tempo que viabiliza a transmissão de diferentes dados entre eles.

7 RESULTADOS

Aqui são apresentados os módulos que foram implementados e o funcionamento de cada um.

7.1 Módulo Controlador

O módulo Controlador é o cerne do sistema. É nele onde:

- Separa-se as tarefas que devem ser automatizadas;
- Realiza-se a comunicação com o sistema onde são criadas as demandas de processos;
- Controla-se o BD
- Filtra-se os dados para envio para outros módulos;

7.1.1 Tarefas

Toda a base de inicialização de tarefas é feita por meio de agendamentos, sejam eles diários, semanais, mensais, ou, até mesmo, que seja executado várias vezes em um mesmo dia sempre dentro de um intervalo de tempo (de hora em hora por exemplo). Por meio da biblioteca Rocketry, as tarefas podem ser inicializadas de maneira independente uma da outra ou de maneira dependente.

Figura 12 – Agendamento de tarefas

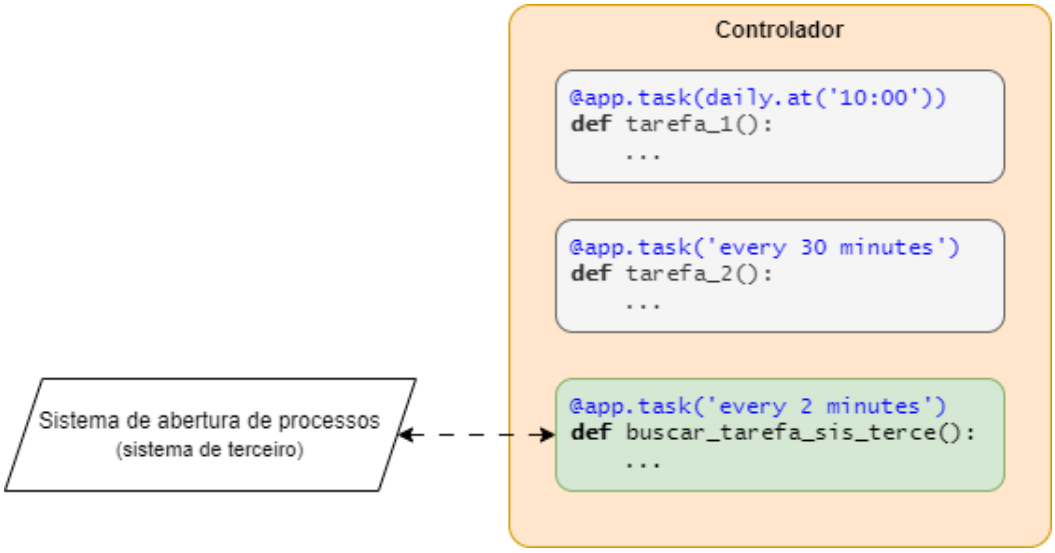
Fonte: Autor.

Como pode ser visto na figura 12 as funções A e B são executadas, cada uma, em seu respectivo tempo. No caso da A, sua execução ocorre diariamente às 10 horas da manhã e a B

a cada 30 minutos. Contudo, a função C tem sua execução dependente da A. Ou seja, ela só será executada após a função "tarefa_1()"ser finalizada. Dessa maneira, sabe-se a "tarefa_3()"é executada diariamente, mas sem um horário programado, pois depende da conclusão de outra função.

O agendamento é importante até mesmo para processos que são requisitados a partir de um sistema terceiro. Isso se deve ao fato que a verificação de tarefas novas é feita de maneira ativa. Ou seja, é necessário que o Controlador faça uma chamada (via API) para verificar se existem trabalhos a serem feitos. Dessa maneira torna-se essencial que exista uma função que seja executada de maneira recorrente. Por exemplo: a cada 2 minutos é feito uma requisição para a plataforma terceirizada buscado-se todos os processos novos (Figura 13).

Figura 13 – Agendamento de Tarefas e Comunicação com Sistemas Externos



Fonte: Autor.

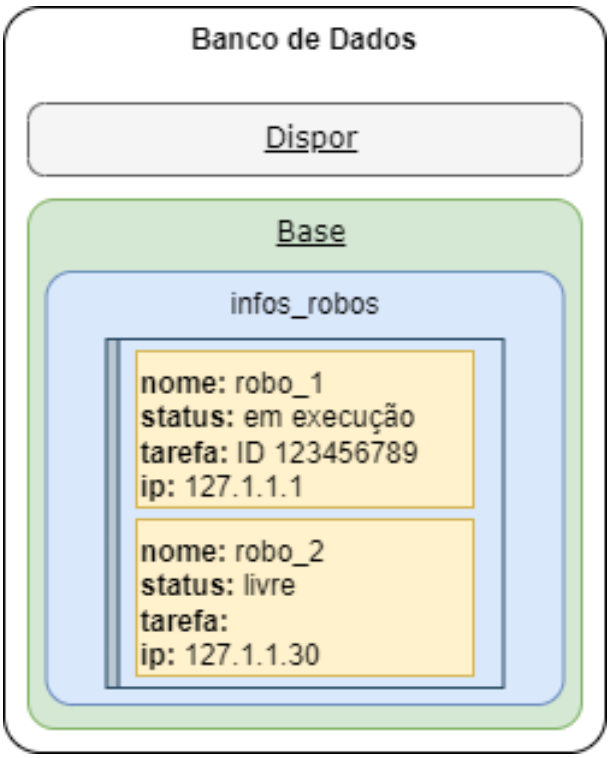
7.1.2 Banco de dados

É possível separar os dados em dois grupos distintos: os necessários para iniciar uma tarefa e os que são gerados após a execução delas. Dessa maneira foram criados dois BD, nomeados de **Base** e **Dispor**, onde cada um armazena as respectivas *collections* do antes e depois da execução dos processos.

Os dados contidos no banco Base servem estrutura para a inicialização de tarefas. Um exemplo é a *collection* 'infos_robos', onde estão contidas informações como o nome do robô, o IP da máquina onde ele instalado, a tarefa que está executando e o seu status atual.

Ao verificar que existe uma tarefa para ser executada, o Controlador busca uma RPA para o qual possa enviá-la. Na figura 14 é possível visualizar a existência de dois robôs. Com base nela, a tarefa encontrada seria enviada para a máquina cujo IP é 127.1.1.30, pois é nela que existe um robô livre, ou seja, ele pode receber um trabalho para executar.

Figura 14 – Banco de Dados - Base



Fonte: Autor.

O BD Dispor é responsável por armazenar dados que já foram tratados parcial ou totalmente. Por exemplo, existe uma tarefa onde é necessário verificar se existe algum cliente está com sua cadastro está atualizado ou não, e, caso não esteja, deve enviar para o funcionário que é responsável por ele. O Controlador inicializa esse processo todos dias, filtrando os dados de uma relatório e vinculando cada um dos clientes ao seu respectivo responsável, e salva essas informações no BD Dispor, dentro de uma *collection* 'tarefas_realizar'. Essas tarefas serão posteriormente enviadas para as RPAs, que irão fazer a verificação de cadastro e definir se o cadastro está atualizado ou não, informação que também será salva no Dispor (Figura 15).

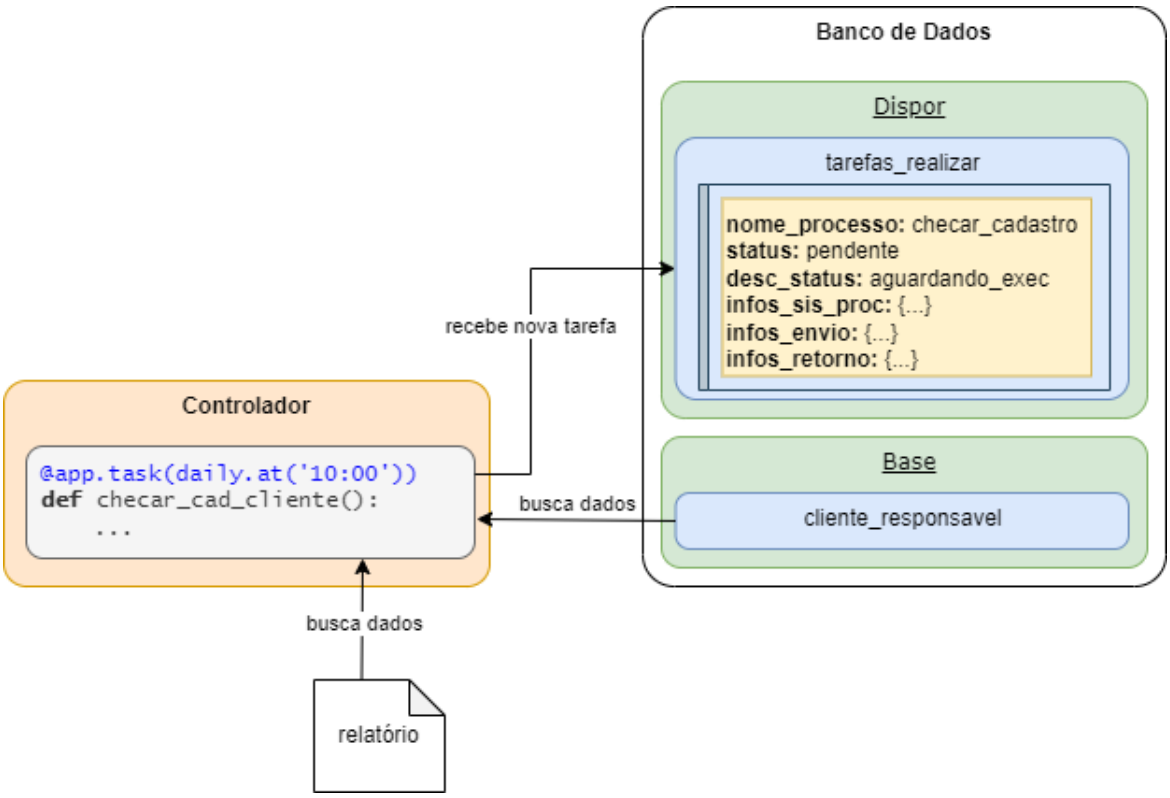
7.1.3 Envio de dados para outros módulos

Outra importante função do módulo Controlador é o envio (e recebimento) de informações para os demais módulos que compõe o sistema (Serviço e Visualizador). Contudo, o ponto significativo não está no envio o recebimento, mas sim no fato de que é ele quem fará os filtros de todas os dados.

Os envios podem ocorrer tanto de maneira ativa quanto passiva. A passiva ocorre, por exemplo, quando o módulo visualizador requisita certas informações. Ou seja, o Controlador é chamado para acessar os dados no BD e então enviar o que lhe for pedido (Figura 16)

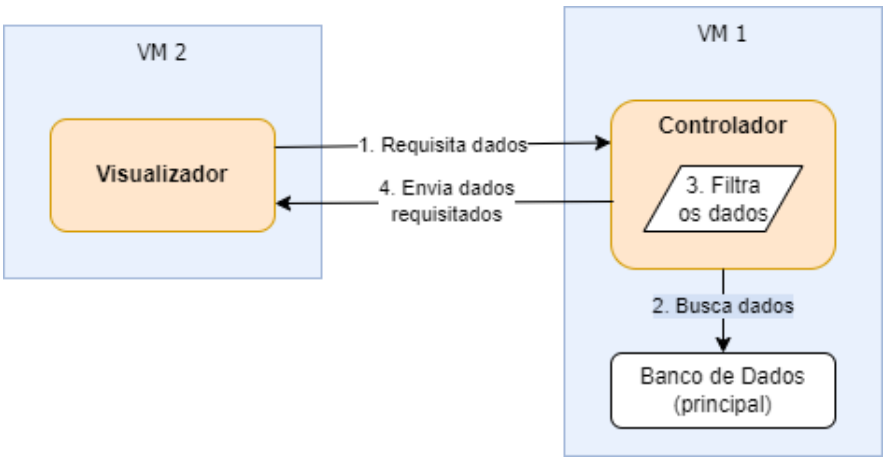
Os envios do tipo Ativos ocorrem com os módulos de serviço. Como é incumbência do Controlador definir quais tarefas serão realizadas e para qual RPA enviar, é ele quem filtra os

Figura 15 – Banco de Dados - Dispor



Fonte: Autor.

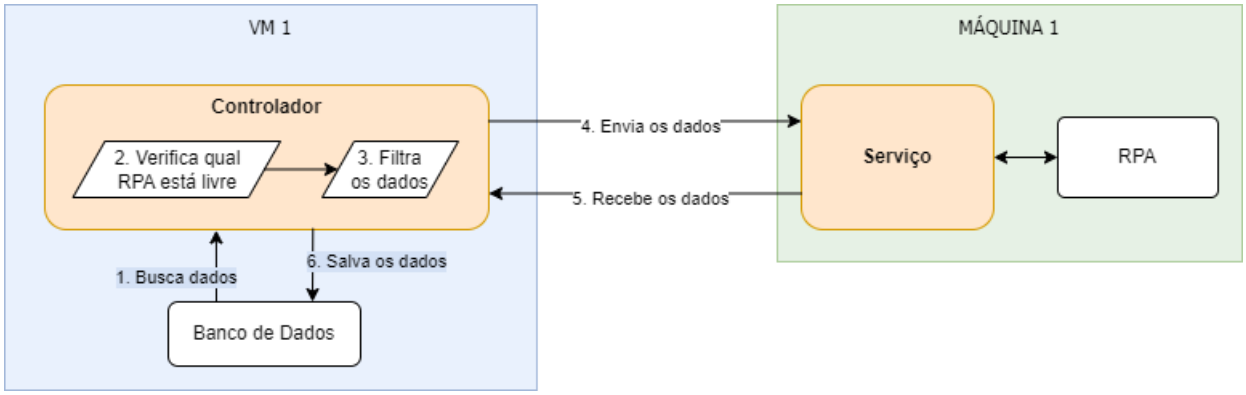
Figura 16 – Envio passivo



Fonte: Autor.

dados necessários para envio. Após a tarefa ser realizada pelo robô, as informações retornam para que sejam salvas no BD, havendo assim um registro do que foi realizado e dos dados obtidos após a execução (Figura 17).

Figura 17 – Envio ativo

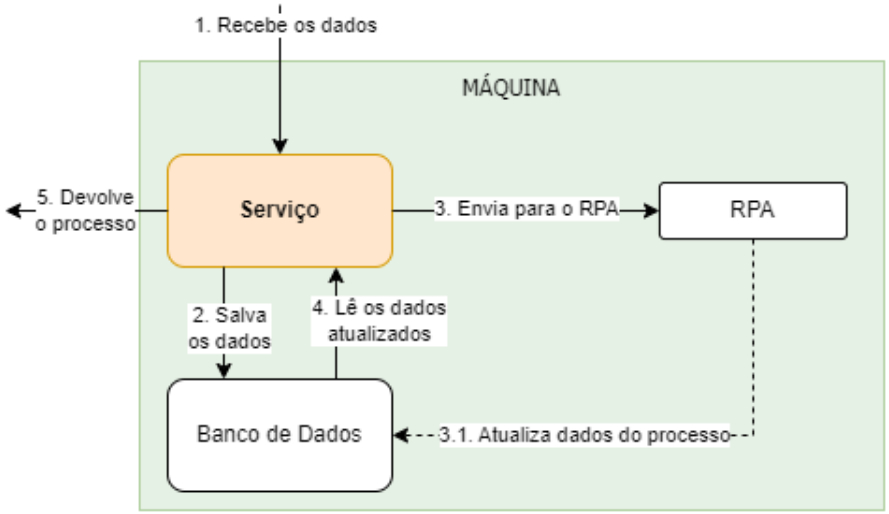


Fonte: Autor.

7.2 Módulo Serviço

A função do módulo de Serviço é ser o intermediário entre o Controlador e a RPA. Assim que recebe uma requisição do Controlador, armazena os dados em um BD temporário, com o qual o robô irá trabalhar. Após isso, verifica qual é a tarefa que deve ser executada e chama o código responsável por realiza-la. Uma vez que foi executada, os dados no BD são organizados e enviados para o Controlador (Figura 18).

Figura 18 – Funcionamento do Módulo de Serviço



Fonte: Autor.

7.3 Módulo Visualizador

A base de funcionamento do módulo Visualizador pode ser vista na figura 16. A definição de quais informações ficam disponíveis nesse módulo é de responsabilidade de cada um dos setores que possuem algum tipo de automação em seus processos. Havendo necessidade

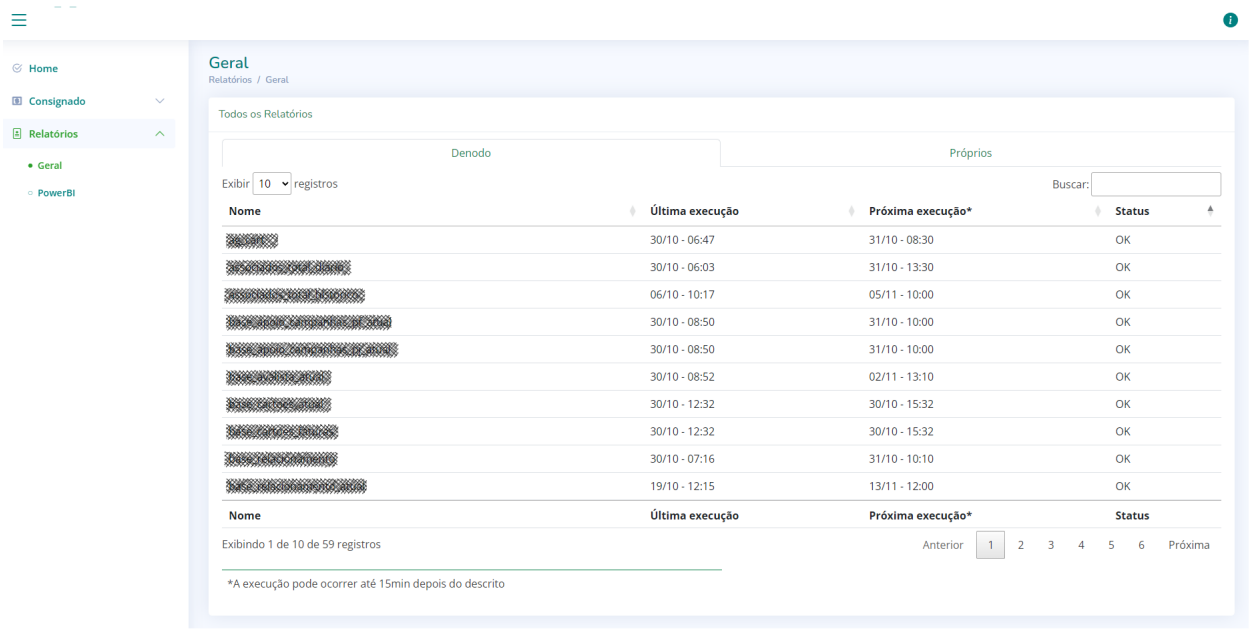
e importância que que determinados dados possam ser acessados por qualquer funcionário, a busca deles é implementada no Controlador, de modo que possam ser acessadas pelo Visualizador.

Referente autenticação para acesso à informações, não há necessidade. Visto que o sistema só funciona na intranet da empresa e que esse módulo tem como objetivo apenas a exibição de dados gerados pelas RPAs, ou seja, não existe o input de dados via o Visualizador, foi definido junto ao setor de TI da empresa que não seria necessária autenticação de usuários. Essa decisão se deve também ao fato que a própria rede já possui o registro dos usuários que a acessam.

O layout final do módulo pode ser visto[1] na figura 19. Dentre as características presentes no Visualizador, salientam-se algumas:

- Como as informações são exibidas em tabelas, é possível clicar no ícone de "menu hambúrguer"para que o menu se retraia, havendo, então, mais espaço para a tabela;
- Para uma melhor exibição dos dados, é possível exibir até 100 item na mesma página;
- É possível realizar uma busca na tabela de forma dinâmica. Ou seja, os dados são filtrados a medida que se escreve;

Figura 19 – Funcionamento do Módulo de Serviço



Fonte: Autor.

[1] Devido a questões de privacidade da empresa, informações foram ocultadas e o menu principal exibido, diminuído

8 CONSIDERAÇÕES FINAIS

8.1 Conclusões

Em 6 meses de funcionamento, 22.146 tarefas já passaram pelo sistema. No decorrer desse tempo ajustes foram realizados para uma melhor performance, mas não houve nenhum erro de processo ocasionado pelo sistema. A receptividade da implementação dele pelo setor de automação foi satisfatório, uma vez que criou-se um padrão para os dados recebidos e enviados para as RPAs, além de uma melhor organização deles. A performance dos robôs também foi aumentada, visto que agora não há necessidade que eles fiquem em *loop* verificando se existem tarefas novas ou não. Existindo, elas apenas são enviadas à eles para execução.

8.2 Trabalhos Futuros

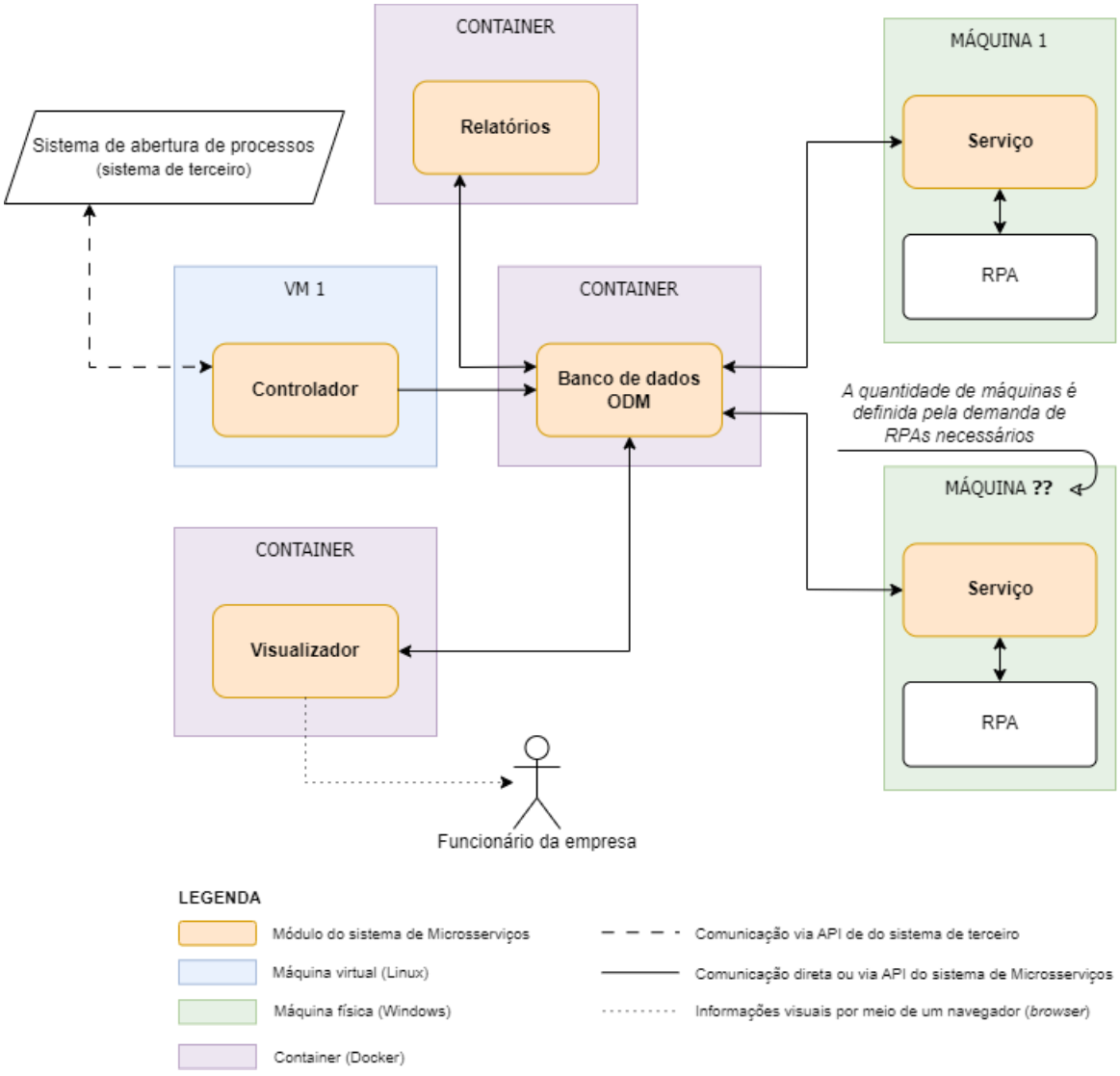
A estrutura na qual todo o sistema foi desenvolvido serve muito bem ao atual cenário da empresa. Contudo, foi percebido que um aumento além do esperado na demanda por automações pode deixar a manutenção dela complicada, além de reduzir sua eficiência. Diante desse cenário, já se discute uma mudança na estrutura do sistema, a fim de melhorar a performance e evitar que o uso dela supere para o que ele foi projetado.

Como pode ser visto na figura 20, é possível ver que algumas alterações já foram desenhadas na estrutura base do sistema. São elas:

1. Utilizar Containers para alguns módulos;
2. Retirar o BD do Controlador, fazendo que esse primeiro torne-se um Object Document Mapping (ODM);
3. Criar um módulo de Relatórios, que teria como única responsabilidade a importação de dados de outras plataformas e criação de relatórios necessários às áreas ou às automações;
4. Retirar os BD temporários dos módulos de Serviço, permitido que eles acessam diretamente o centra;

Ainda são necessários mais estudos e essa estrutura pode vir a ser modificada, mas a ideia é que ela comporte um maior fluxo de dados e de processos sendo executados simultaneamente.

Figura 20 – Esboço da nova estrutura para o sistema



Fonte: Autor.

REFERÊNCIAS

ANYWHERE, A. **About Us**. 2023. Disponível em: https://www.automationanywhere.com/company/about-us. Acesso em: 15 de abril de 2023.

AUTOMATE, P. **Process Advisor**. 2023. Disponível em: https://powerautomate.microsoft.com/pt-br/process-advisor/. Acesso em: 15 de abril de 2023.

CABRAL ARIEL BEHR, G. S. S. P. H. D. **ROBOTIC PROCESS AUTOMATION: PROPOSTA DE DEFINIÇÃO PARA A TECNOLOGIA NO CONTEXTO CONTÁBIL**. Porto Alegre, 2022. 26 f.

CXTODAY. **Gartner Says Worldwide RPA Software Spending to Reach $2.9 Billion in 2022**. 2022. Disponível em: https://www.cxtoday.com/data-analytics/gartner-magic-quadrant-for-robotic-process-automation-rpa-2022. Acesso em: 15 de abril de 2023.

DIVINO, B. **O que é Python?** 2021. Disponível em: https://www.alura.com.br/artigos/python-uma-introducao-a-linguagem. Acesso em: 27 de maio de 2023.

GARTNER. **Gartner Magic Quadrant**. 2023. Disponível em: https://www.gartner.com.br/pt-br/metodologias/magic-quadrant. Acesso em: 22 de abril de 2023.

JúNIOR, L. M. C. M. **ROBOTIC PROCESS AUTOMATION: PROPOSTA DE DEFINIÇÃO PARA A TECNOLOGIA NO CONTEXTO CONTÁBIL**. [*S.l.*], 2019. 17 f.

JúNIOR, V. G. de F. **Automação de Processos de Negócio Utilizando Robotic Process Automation (RPA) em Um Centro de Serviços Compartilhados (CSC): Um Estudo de Caso**. Uberlandia, 2021. 68 f.

LIMA, G. **MongoDB: O banco baseado em documentos**. 2021. Disponível em: https://www.alura.com.br/artigos/mongodb-o-banco-baseado-em-documentos. Acesso em: 27 de maio de 2023.

MANZUETO, M. S. **Automação de processos: a influência dos softwares de automação de processos nas rotinas organizacionais**. 2016. 57 f. Dissertação (Mestrado em Administração de Empresas) — Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2016.

MARIANO, A. F. **AUTOMAÇÃO ROBÓTICA DE PROCESSOS: UMA ANÁLISE SOBRE A GOVERNANÇA DE RPA PARA GRANDES EMPRESAS**. São Pauolo, 2021. 90 p.

MEDES, I. S. **Arquitetura Monolítica vs Microsserviços: uma análise comparativa**. Brasília, 2021. 90 p.

SANTOS BRUNA APARECIDA DE OLIVEIRA, V. H. J. M. A. A. M. P. I. F. G. A. C. Automação industrial em pequenas, médias e grandes empresas: Um estudo teórico. 2017.

SANTOS, R. de C. **Estudo Comparativo dos Sistemas Gerenciadores de Bancos de Dados: Oracle, SQL Server e PostgreSQL**. Campolide, 2019. 15 f.

SILVA, M. C. B. Arthur Marcos da. **Automação Robótica de Processos (RPA): Estudo de Caso Através da Tarefa Administrativa Contas a Pagar**. São Paulo, 2018. 19 f.

SOUZA, A. A. de. **Organização, processos e tomada de decisão**. Florianópolis, 2021. 109 f.

STAMFORD, C. **Gartner Says Worldwide RPA Software Spending to Reach $2.9 Billion in 2022**. 2022. Disponível em: https://www.gartner.com/en/newsroom/press-releases/2022-08-1-rpa-forecast-2022-2q22-press-release. Acesso em: 15 de abril de 2023.

UIPATH. **About Us**. 2023. Disponível em: https://www.uipath.com/company/about-use. Acesso em: 15 de abril de 2023.