UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

Desenvolvimento de um plug-in para a replicação de dados entre os sistemas NetBox e ServiceNow CMDB

Relatório Técnico relativo ao Trabalho de Conclusão de Curso do Bacharelado em Sistemas de Informação na modalidade Empresa

---

Aluno
Manassés Júlio da Silva Júnior

Orientadora
Roberta Macêdo Marques Gouveia
Departamento de Estatística e Informática - DEINFO

Maio de 2023

Manassés Júlio da Silva Júnior

Desenvolvimento de um plug-in para a replicação de dados entre os sistemas NetBox e ServiceNow CMDB

Relatório Técnico apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Universidade Federal Rural de Pernambuco – UFRPE
Departamento de Estatística e Informática Curso de Bacharelado em Sistemas de Informação

Orientadora
Roberta Macêdo Marques Gouveia
Departamento de Estatística e Informática - DEINFO

Recife
Maio de 2023

Resumo

Este trabalho apresenta um novo plugin desenvolvido para integrar o software de configuração de rede de código aberto NetBox com o ServiceNow CMDB. Esse plugin estende a funcionalidade do NetBox, permitindo que os usuários enviem dados do NetBox para a *Application programming interface* (API) do ServiceNow. O NetBox é um software open-source de configuração de rede que oferece uma modelagem e documentação de redes moderna. O projeto é desenvolvido publicamente no GitHub e também age como um repositório centralizado para informações de infraestrutura de rede, incluindo inventário de dispositivos, gerenciamento de endereços IP, gerenciamento de cabos e gerenciamento de energia. Por outro lado, o ServiceNow CMDB é um repositório central que contém informações sobre os ativos e itens de configuração na infraestrutura de TI de uma organização. A integração entre essas plataformas é feita por meio da criação de plugins que ampliam a funcionalidade do NetBox, permitindo que ele trabalhe em conjunto com o ServiceNow CMDB. O projeto usa Python como a linguagem de programação principal, o framework web Django e o Docker para criar o ambiente de desenvolvimento. Em geral, esse projeto fornece uma ferramenta poderosa e flexível para que administradores e operadores de rede gerenciem sua infraestrutura de rede. A arquitetura do plugin segue a arquitetura Django MTV (Model-Template-View), em que o Model representa os dados e o esquema do banco de dados, o View lida com solicitações e respostas e o Template gera a saída HTML. A principal funcionalidade do projeto é a replicação automática das modificações *Create, Read, Update, Delete* (CRUD) em objetos selecionados do NetBox para o ServiceNow CMDB, feita por meio da API do ServiceNow. Esse recurso de replicação automática usa Webhooks para monitorar modificações de objetos, e o plugin lida automaticamente com a criação e exclusão deles. Webhook é um sinal enviado para um domínio de servidor especificado sempre que um evento especificado é acionado (Bai, 2022). Outros recursos incluem um lote manual e simulação para replicar dados para o CMDB. A interface visual do plugin é simples e focada em suas funcionalidades.

Palavra-chave: Webhook, NetBox, ServiceNow, Django.

Abstract

This work presents a new plugin developed to integrate the open source network configuration software NetBox with the ServiceNow CMDB. This plugin extends the functionality of NetBox, allowing users to send NetBox data to the ServiceNow API. NetBox is an open source web application that helps manage and document computer networks. It acts as a centralized repository for network infrastructure information, including device inventory, IP address management, cable management, and power management. On the other hand, the ServiceNow CMDB is a central repository that contains information about all assets and configuration items in an organization's IT infrastructure. The integration between these platforms is achieved through the creation of plugins that extend the functionality of NetBox, allowing it to work together with the ServiceNow CMDB.
The project uses Python as the main language, the Django web framework, and Docker to create the development environment. Overall, this project provides a powerful and flexible tool for network administrators and operators to manage their network infrastructure. The plugin architecture follows the Django MTV (Model-View-Template) architecture, where the Model represents the data and database schema, the View handles requests and responses, and the Template generates the HTML output. The main functionality of the project is the automatic replication of Create, Read, Update and Delete (CRUD) modifications in selected objects from NetBox to the ServiceNow CMDB, done through the ServiceNow API. This automatic replication feature uses Webhooks to monitor object modifications, and the plugin automatically handles the creation and deletion of Webhooks. Other features include manual batch and simulation for replicating data to the CMDB. The visual interface of the plugin is simple and focused on its functionalities.

keywords: Webhook, NetBox, ServiceNow, Django.

Sumário

1 Introdução

Este trabalho desenvolve um plugin que integra o NetBox com o ServiceNow Configuration Management Database (CMDB) utilizando a estrutura do Django. Isso permite que os administradores de rede gerenciem e documentem a infraestrutura de rede de forma mais intuitiva e automatizada. O NetBox é um gerenciador de infraestrutura de rede que fornece recursos para gerenciar e documentar equipamentos de rede, endereços IP, circuitos e conexões de fibra, entre outros elementos. Por sua vez, o ServiceNow CMDB é um dos recursos da plataforma ServiceNow que permite às organizações gerenciar seus recursos de TI, manter seus ativos de TI atualizados e precisos, além de gerenciar processos como gestão de mudanças, incidentes e problemas.

Com o plugin, os administradores e operadores de rede têm acesso a informações precisas e atualizadas sobre todos os ativos e itens de configuração na infraestrutura de TI da organização. O plugin oferece recursos como auto-replicação, batch manual e dry run, permitindo a gestão mais eficiente da infraestrutura de TI. A CMDB do ServiceNow fornece uma visão única e centralizada dos relacionamentos e dependências entre todos os ativos de TI, permitindo que as equipes de TI gerenciem melhor a complexidade de seus ambientes de TI. Isso proporciona uma gestão mais eficiente dos processos de TI, fornecendo informações precisas para tomada de decisão e planejamento de capacidade. Em suma, o plugin desenvolvido neste trabalho melhora a eficiência e eficácia das operações de TI, permitindo que as equipes de Tecnologia da Informação (TI) tomem decisões informadas e gerenciem melhor a infraestrutura de rede.

Em 2020, o Netbox introduziu na sua versão 2.8 o suporte para *plugins* personalizados, o que tornou possível estender a funcionalidade do NetBox além do que o produto principal oferece (NetBox, 2020). A arquitetura do Django MTV (Model-View-Template) é implementada no NetBox, onde o Model representa os dados e o esquema do banco de dados do aplicativo, a View é responsável por processar as requisições dos usuários e gerar respostas, e o Template é responsável por gerar a saída *Hyper Text Markup Language* (HTML) que é enviada ao navegador do usuário (DJANGO, 2023).

Com a integração do NetBox com o ServiceNow CMDB, é possível gerenciar e documentar redes de computadores de forma mais intuitiva e automatizada, melhorando a eficiência e eficácia das operações de TI (ITSM, 2021). Além disso, o plugin desenvolvido permite a visualização de informações precisas e atualizadas sobre todos os ativos e itens de configuração na infraestrutura de TI da organização, o que facilita o planejamento e a tomada de decisões informadas pelas equipes de TI (SN, 2023).

1.1 Problema e Justificativa

O problema surge em decorrência da empresa ter como meta agregar mais valor de mercado ao NetBox. Tal busca por maneiras de agregar mais valor a um software é uma estratégia comum no mercado de tecnologia, haja vista que o tamanho do mercado global de integração de sistemas foi avaliado em US$ 396,41 bilhões em 2022 e espera-se que atinja cerca de US$ 1.492,95 bilhões até 2032, com um crescimento a uma taxa de crescimento anual composto de 14,18% no período de previsão de 2023 a 2032, como mostra na figura 1 (Precedenceresearch, 2023).

No segmento de empresas que fornecem soluções de software, a integração com outras ferramentas e serviços é uma das maneiras mais eficazes de aumentar a percepção de valor por parte dos clientes. De acordo com a MuleSoft (2020), "A estratégia de integração correta permite que as organizações reduzam a complexidade e gerem valor comercial tangível". Com base no que foi solicitado pelo cliente, foi definido que o sistema Netbox deverá ser integrado ao ServiceNow CMDB. No entanto, é importante destacar que de acordo com uma pesquisa realizada por Peerspot (2023), embora o ServiceNow CMDB seja classificado como a segunda opção mais popular em termos de bancos de dados de gerenciamento de configuração, os usuários da plataforma avaliam-na positivamente, dando uma classificação média de 8,8 em uma escala de 10 no PeerSpot. Portanto, mesmo que existam outras opções disponíveis no mercado, o ServiceNow CMDB é altamente avaliado pelos seus usuários.

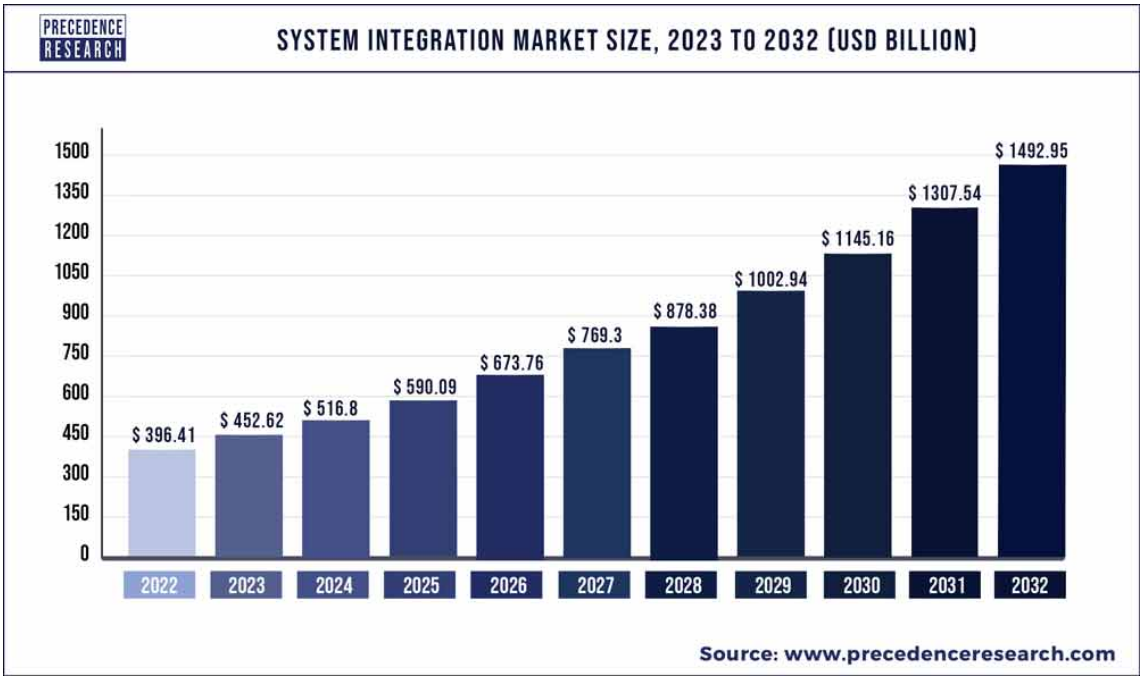
Além disso, ter integrações eficazes entre SIs tornou-se um dos aspectos críticos para a operação dos ambientes de negócios, sobretudo aqueles de natureza altamente colaborativa (Osvaldo et al., 2005).

A decisão de integrar o NetBox com o ServiceNow foi tomada por meio da criação de um plugin que será incorporado diretamente no NetBox. Este plugin foi requisitado pelo cliente devido às

funcionalidades propostas. Devido à sua dependência dos dados do Netbox, é a única opção viável. Para ter acesso ao banco de dados do sistema, é necessário que o administrador do Netbox conceda acesso aos dados. Essa abordagem é possível porque a versão 2.8 do sistema introduziu a capacidade de desenvolver plugins para o software. Outro exemplo de integração seria por meio da criação de conectores, os conectores serviriam de ponte entre os dois sistemas.

A integração com o ServiceNow pode ser vantajosa para empresas que usam o NetBox em conjunto com outras ferramentas e sistemas que estão integrados ao ServiceNow. A integração possibilita uma gestão mais centralizada e simplificada de diversos processos e informações relacionados à infraestrutura de TI.

Figura 1. Tamanho do mercado de integração de sistemas



Fonte: Precedenceresearch (2023)

2 Desenvolvimento realizado na empresa

A organização na qual sou empregado é uma empresa global de software e engenharia digital que oferece serviços de última geração, como análise preditiva, inteligência artificial e aprendizado de máquina, Internet of Things (IoT), nuvem e automação de testes. Possui mais de 9000 funcionários ao redor do mundo, 40 escritórios e laboratórios de inovação e está presente em mais de 14 países. A empresa também possui os prêmios de *great place to work, best teams engineering, best company outlook, best company global culture, best teams hr, best company career growth*.

Comecei a trabalhar na empresa em janeiro de 2022 e durante o meu trajeto estive em contato com 4 projetos, o primeiro deles eu tive uma participação pequena devido ao fato de ser estagiário e também devido ao projeto que estava na reta final. No projeto trabalhei no *back-end* utilizando a linguagem GO, como atividades principais realizei a implementação de uma nova variável de ambiente que facilitaria o trabalho no setor de *Quality Assurance* além de também implementar diversos testes unitários que, embora o sistema possuísse alguns anos não tinha sido implementado.

Este trabalho está sendo realizado como parte do meu segundo projeto na empresa. Fui convidado para participar da equipe com base na minha experiência prévia e recomendação do meu trabalho no projeto anterior. O motivo principal para o convite foi o meu conhecimento na criação de APIs e na linguagem Python.

Durante o meu estágio, tive a oportunidade de participar de um projeto de grande escala que envolvia equipes de desenvolvedores de diversas partes do mundo. O projeto era de escala internacional e, como parte da equipe de desenvolvimento, passei por vários times, incluindo o time de Technical Writing. O software em questão trabalhava em um nível bastante baixo, o que exigia

conhecimento avançado do sistema operacional Ubuntu para navegar pelo código-fonte.

Como o projeto era de escala internacional, não fui responsável por desenvolver novas funcionalidades, mas sim por testar e corrigir bugs. A minha principal tarefa consistia em corrigir um bug em um dos processos do sistema. O setor de Quality Assurance relatou que o processo deveria retornar um erro após a execução de determinadas etapas, mas isso não estava ocorrendo. Realizei uma busca detalhada no código-fonte para identificar o problema e corrigi-lo. Para esse projeto, utilizei minhas habilidades de resolução de problemas e de navegação em sistemas operacionais de baixo nível, o que me permitiu adquirir um conhecimento mais profundo em tecnologias de larga escala.

Estou trabalhando atualmente no quarto projeto e estou na fase de *onboard*, essa fase se define pela introdução ao projeto e estudo das tecnologias que a envolvem, nesse sistemas irei trabalhar como desenvolvedor *full-stack* utilizando as tecnologias NodeJS, Javascript, Amazon Web Services, Graphql e Docker.

2.1 NetBox e seus Plugins

O NetBox é uma aplicação de código aberto projetada para ajudar, gerenciar e documentar redes de computadores. Ele serve como um repositório centralizado para informações de infraestrutura de rede, como inventário de dispositivos, gerenciamento de endereços IP, gerenciamento de cabos e gerenciamento de energia. NetBox fornece uma interface intuitiva e amigável (Figura 1) para visualizar dados de rede e automatizar tarefas comuns de gerenciamento de rede.

O NetBox é escrito em Python e construído sobre a estrutura da Web Django, o que o torna altamente personalizável e extensível. Ele pode se integrar a outras ferramentas de gerenciamento de rede por meio de sua *Application Programming Interface RESTful*, permitindo que ele se encaixe perfeitamente na infraestrutura de rede existente.No geral, o NetBox é uma ferramenta poderosa para administradores e operadores de rede que precisam acompanhar sua infraestrutura de rede, desde pequenas redes locais até ambientes corporativos de grande escala.

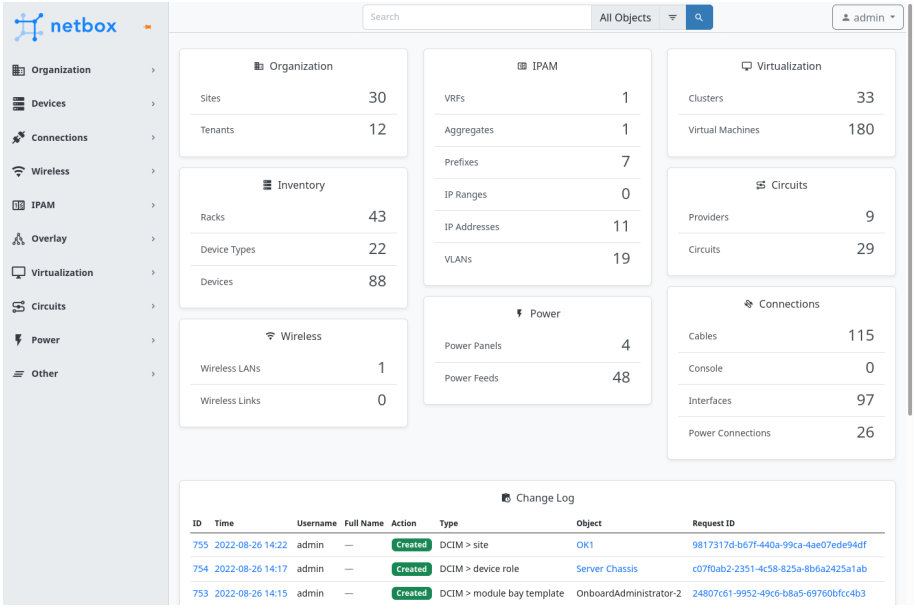
No NetBox, um plug-in (Netbox, 2020) é um módulo ou extensão personalizada que pode ser adicionada à plataforma para fornecer funcionalidade adicional além de seus recursos principais. Um *plug-in* pode ser considerado como um código de *software* que estende a funcionalidade do NetBox sem exigir alterações em sua base de código principal.

Os plug-ins do NetBox são normalmente criados para atender a casos de uso específicos ou para adicionar novos recursos à plataforma. Por exemplo, um plug-in pode ser criado para adicionar suporte a um novo tipo de dispositivo ou para integração com uma ferramenta de terceiros. Os plugins podem ser desenvolvidos em Python ou outras linguagens de programação que suportam a integração com o framework web Django, que é utilizado pelo NetBox.

Os plug-ins do NetBox podem ser instalados simplesmente copiando o código do *plug-in* para o servidor NetBox e adicionando-o à lista de plugins instalados no arquivo de configuração. Depois de instalados, os plugins podem ser acessados por meio da interface da Web do NetBox e integrados aos fluxos de trabalho e processos conforme necessário.

No geral, os *plug-ins* fornecem uma maneira flexível e extensível de personalizar o NetBox para atender às necessidades exclusivas de uma organização, tornando-o uma ferramenta poderosa para gerenciar a infraestrutura de rede.

Figura 2. Tela inicial NetBox



Fonte: NetBox

2.2 ServiceNow CMDB

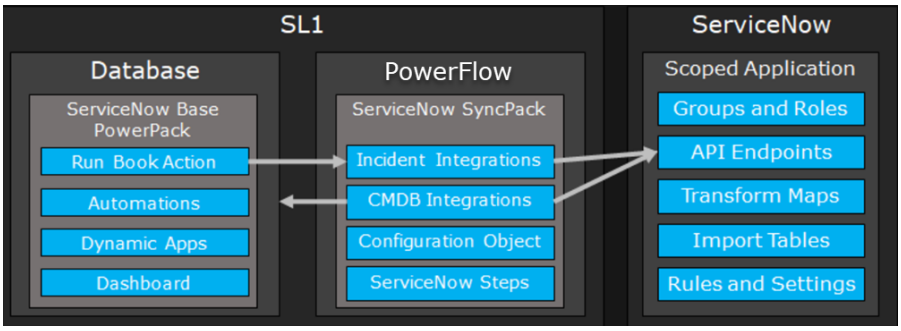
O ServiceNow CMDB é um repositório central que contém informações sobre todos os ativos e itens de configuração na infraestrutura de Tecnologia da Informação de uma organização (SN, 2023). É um componente essencial do conjunto IT Service Management (ITSM) da ServiceNow.

O CMDB fornece uma fonte única de verdade para todos os ativos de infraestrutura de TI e seus relacionamentos, incluindo hardware, software, dispositivos de rede, aplicativos e serviços. Ele permite que as equipes de TI rastreiem mudanças, monitorem relacionamentos e gerenciem as dependências entre diferentes componentes de TI.

O ServiceNow CMDB fornece uma base para outros aplicativos da ServiceNow, como gerenciamento de incidentes, gerenciamento de problemas, gerenciamento de mudanças e gerenciamento de ativos, para trabalharem juntos de forma integrada. Ele permite que as equipes de TI obtenham insights sobre o ambiente de TI, tomem decisões informadas e melhorem a eficiência e eficácia das operações de TI.

Na figura 2 é possível ver um exemplo de como a integração com o ServiceNow CMDB funciona, o sistema SL1 se conecta com os ServiceNow API através do CMDB integrations onde é possui fazer as requisições para os *endpoints* do sistema. Essa integração pode ser vista na seção 2.4.1 onde vai conter a conexão e validação com o ServiceNow e no código 2 o envio do objeto para o *endpoint*.

Figura 3. Exemplo de integração com o ServiceNow



Fonte: Sciencelogic

2.3 Tecnologias utilizadas

Para o desenvolvimento do plugin, com vistas a integrar o software de configuração de rede NetBox com o ServiceNow CMDB, foram utilizadas várias tecnologias, dentre elas, tem-se: Docker, Python, Django e jQuery.

O Docker é uma plataforma de código aberto que permite aos desenvolvedores criar, implantar e executar aplicativos em contêineres. Os contêineres são um pacote de software leve e executável autônomo que inclui tudo o que é necessário para executar um aplicativo, incluindo código, bibliotecas, ferramentas do sistema e configurações. O Docker fornece uma maneira de empacotar e executar aplicativos de maneira consistente, repetível e portátil, facilitando a implantação de aplicativos em diferentes ambientes, como desenvolvimento, teste e produção. No projeto o Docker foi utilizado para criar o ambiente de desenvolvimento, na execução do Docker ele cria um container com o NetBox instalado e também sobre o código do *plug-in* que vai ser acoplado no NetBox.

Já a linguagem utilizada no projeto foi Python, onde é uma linguagem de programação versátil e poderosa amplamente utilizada na indústria, academia e pesquisa. Sua popularidade se deve em parte à sua simplicidade, legibilidade e facilidade de uso, bem como à sua capacidade de integração com outras linguagens e sistemas de programação.

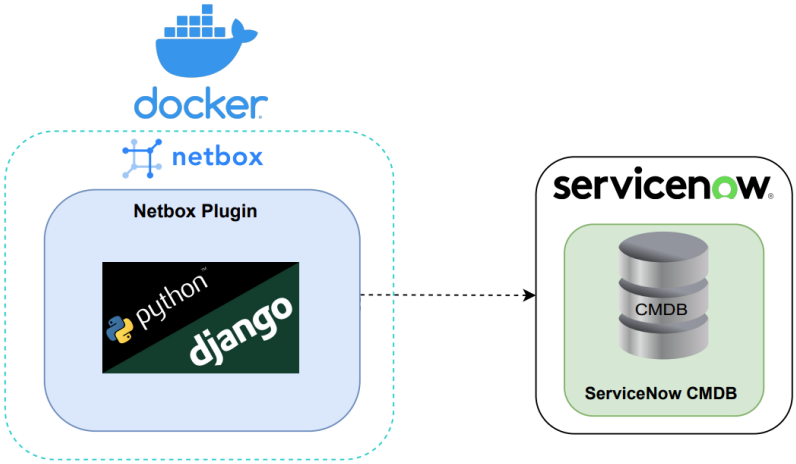
O Django é um framework web poderoso e flexível que permite aos desenvolvedores construir aplicações web complexas de forma rápida e eficiente, seguindo as melhores práticas e mantendo a alta qualidade do código. O Django é o *framework* requerido no NetBox para o desenvolvimento do *plug-in,* já que o sistema também foi criado sobre o mesmo *framework.*

O jQuery também foi utilizado no projeto junto com HTML na parte visual. O Query é uma biblioteca JavaScript poderosa e flexível que simplifica o desenvolvimento da Web, fornecendo um conjunto de métodos consistente e fácil de usar para trabalhar com elementos HTML, eventos e animações. Sua popularidade e comunidade ativa o tornam uma ferramenta valiosa para qualquer desenvolvedor web.

2.4 Desenvolvimento

Para desenvolver o projeto, seguimos um fluxo simples de trabalho (FIGURA 3). Primeiramente, criamos um ambiente de desenvolvimento utilizando o Docker, como detalhado na seção 2.3 do relatório. Em seguida, iniciamos a codificação do plugin, começando pela conexão com o ServiceNow CMDB. Depois, seguimos para a implementação das funcionalidades propostas, que incluíram auto replicação, execução em lote manual e execução de teste ("dry run").

Figura 3. Diagrama do projeto

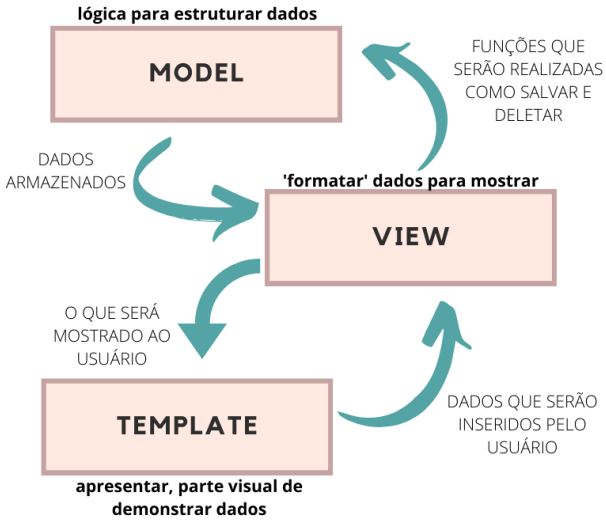


Fonte: De autoria própria (2023).

De acordo com a documentação do NetBox, para desenvolver um plug-in para o sistema, é

necessário seguir a arquitetura *Model-Template-View* (MTV) do Django (FIGURA 4). Essa estrutura é fundamental para garantir uma separação clara de responsabilidades e facilitar a manutenção e escalabilidade do projeto.

Figura 4. Arquitetura MTV (Django)



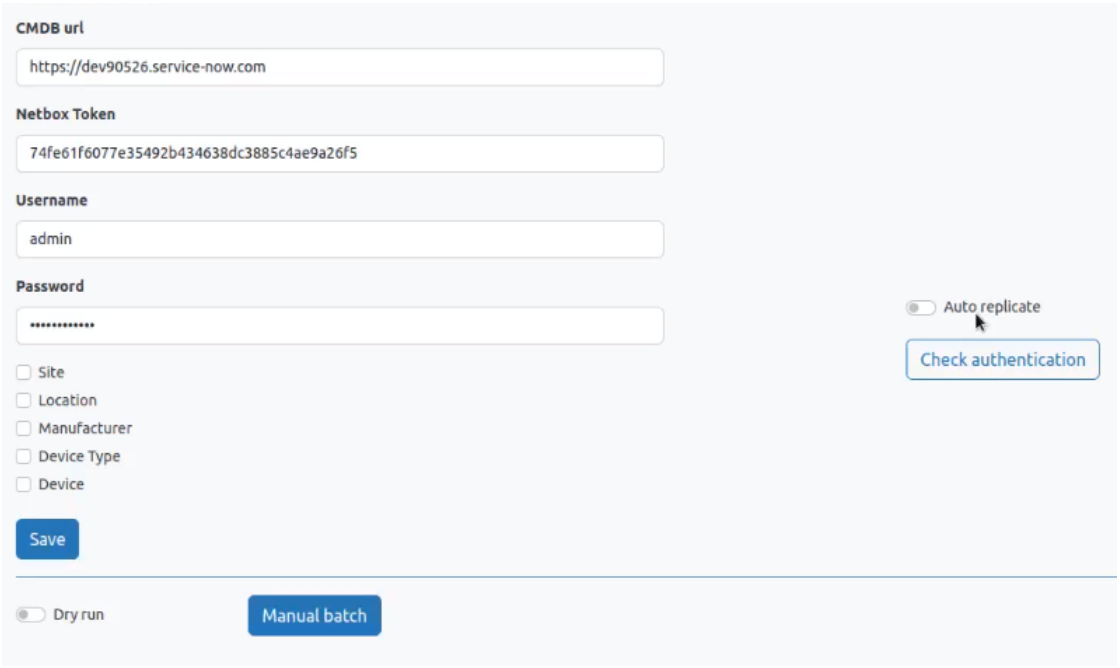
Fonte – Medium (2020)

Django MTV é uma variação da arquitetura tradicional Model-View-Controller (MVC) usada no desenvolvimento de aplicativos da web. Esta estrutura funcionará com base no *Model* como um banco de dados e *View* como uma funcionalidade de controle e os *Templates* vão trabalhar ao lado do usuário para comunicação e interação (Kumar et al, 2021). No Django, essa arquitetura é implementada da seguinte forma:

- Model: O *model* representa os dados e o esquema de banco de dados do aplicativo. Ele define a estrutura das tabelas do banco de dados e os campos que elas contêm. No Django, os models são criados como classes Python, com cada classe representando uma tabela de banco de dados.
- View: A *View* é responsável por processar as requisições dos usuários e gerar respostas. Ele atua como intermediário entre o Modelo e o Modelo, consultando o Modelo em busca de dados e passando-os para o Modelo para renderização. No Django, as visualizações são implementadas como funções ou classes Python que recebem solicitações HTTP e retornam respostas HTTP.
- Template: O Template é responsável por gerar a saída HTML que é enviada ao navegador do usuário. Ele define a estrutura e a aparência da interface do usuário, incluindo *layout*, estilos e conteúdo dinâmico. No Django, os modelos são implementados usando um mecanismo de modelagem que suporta tags, filtros e variáveis para gerar conteúdo dinâmico.

O sistema teve foco principal nas suas funcionalidades, enquanto a parte visual foi trabalhada de forma simples conforme figura 5.

Figura 5. Parte visual do sistema



Fonte: De autoria própria (2023).

2.4.1 Conexão com ServiceNow

A primeira parte do projeto foi a conexão com o ServiceNow CMDB, essa conexão é feita através da API de instância do CMDB que fornece *endpoints* para criar, ler, atualizar e excluir operações em tabelas existentes do banco de dados de gerenciamento de configuração (CMDB). A API do ServiceNow refere-se a um conjunto de interfaces e protocolos baseados na Web que permitem que sistemas externos interajam e acessem dados de uma instância do ServiceNow.

Usando a API ServiceNow, os desenvolvedores podem criar aplicativos personalizados, integrações e automações que ampliam os recursos da plataforma ServiceNow. A API oferece suporte a vários métodos de autenticação, incluindo OAuth, autenticação básica e chaves de API.

Algumas funções foram feitas para realizar a conexão na API de instâncias do CMDB, um exemplo delas é a função que realiza a autenticação das credenciais do usuário que pode ser vista abaixo. O código 1 é bem simples, recebe as credenciais do usuário e faz uma requisição para a API.

Código 1. Função de autenticação

```
@staticmethod
def do_authentication(credentials: Credentials):
    url = f"{credentials.cmdb_url}/{CMDB_OVERALL_ROUTE}"
    return requests.get(url=url, auth=(credentials.username, credentials.password))
```

Fonte: De autoria própria (2023).

Inicialmente, o projeto propõe a replicação de alguns objetos do NetBox para observar como essa ação será refletida no ServiceNow. No entanto, o objetivo futuro do plugin é realizar a replicação dinâmica e automática de todos os objetos do NetBox. No atual sistema os objetos que serão replicados são: *Site, Location, Manufacturer, Device Type e Device*.. Para poder fazer o *Create, Read, Update, Delete* (CRUD) desses objetos para o CMDB é preciso de algumas funções especiais conforme códigos 2 e 3, essas funções tem como objetivo realizar as requisições para a API.

Código 2. Função de salvar do objeto *Location*

```
@staticmethod
def save_cmn_location(credentials: Credentials, location: CmnLocation):
    url = f"{credentials.cmdb_url}/{CMDB_TABLE_ROUTE}/{location.table_name}"
    content = SnowCmdbProxy.CmnLocation.Schema().dump(location)
    return requests.post(
        url=url,
        headers=HEADERS,
        auth=(credentials.username, credentials.password),
        json=content,
    )
```

Fonte: De autoria própria (2023).

Código 3. Função de editar do objeto *Location*

```
@staticmethod
def update_cmn_location(credentials: Credentials, location: CmnLocation):
    url = f"{credentials.cmdb_url}/{CMDB_TABLE_ROUTE}/{location.table_name}/{location.sys_id}"
    content = SnowCmdbProxy.CmnLocation.Schema().dump(location)
    return requests.put(
        url=url,
        headers=HEADERS,
        auth=(credentials.username, credentials.password),
        json=content,
    )
```

Fonte: De autoria própria (2023).

Além das funções de salvar e editar mostradas acima também existe a função de deletar, não só para o objeto *Location*, mas para todos os objetos citados acima.

2.4.2 Auto replicate

A replicação automática é a principal funcionalidade do sistema. Com esta opção ativada, todas as modificações de criação, leitura, atualização e exclusão (CRUD) nos objetos selecionados pelo usuário serão monitoradas, e quando uma dessas ações ocorrer, o sistema replicará automaticamente a alteração. O monitoramento dos objetos é realizado pelo próprio NetBox, por meio de *Webhooks*. Um *webhook* é um mecanismo para transmitir uma alteração ocorrida no NetBox a um sistema externo. Por exemplo, é possível notificar um sistema de monitoramento sempre que o status de um dispositivo for atualizado no NetBox. O uso dos Webhooks permite que as alterações feitas no NetBox sejam refletidas em tempo real em outros sistemas, proporcionando uma maior eficiência e integridade dos dados.

O usuário não precisa lidar com o *Webhook*, o *plug-in* realiza todas a ações de forma automática, desde a criação do Webhook até a sua deleção, conforme código 4 e 5:

Código 4. Configuração do Webhook

```
def snowWebookConfig(self,netbox_token, auto_replicate, site, location, manufacturer, device_type, device):
    contentTypes = self.buildContentTypes(site, location, manufacturer, device_type, device)
    isWebhookCreated = self.getWebhook(netbox_token)
    if auto_replicate == False:
        self.deleteWebhook(isWebhookCreated[1], netbox_token)
        logger.info("Snow CMDB - Wevhook deleted successfully")
    else:
        self.setAutoReplicate(netbox_token, site, location, manufacturer, device_type, contentTypes, isWebhookCreated)
```

Fonte: De autoria própria (2023).

Código 5. Configuração *auto replicate*

```
def setAutoReplicate(self, netbox_token, site, location, manufacturer, device_type, contentTypes, isWebhookCreated):
    if isWebhookCreated[0] == False:
        self.createUpdateWebhook(contentTypes, netbox_token, "POST")
        logger.info("Snow CMDB - Wevhook created successfully")
    else:
        if site == False and location == False and manufacturer == False and device_type == False:
            self.deleteWebhook(isWebhookCreated[1], netbox_token)
            logger.info("Snow CMDB - Wevhook deleted successfully")
        else:
            self.createUpdateWebhook(contentTypes, netbox_token, "PUT", id=isWebhookCreated[1])
            logger.info("Snow CMDB - Wevhook updated successfully")
```

Fonte: De autoria própria (2023).

A função "createUpdateWebhook" é responsável tanto pela criação quanto pela edição do *Webhook*. O terceiro parâmetro da função determina o tipo de ação (Post, Put) a ser realizada.

Uma das configurações do *Webhook* é a "payload_url". Quando ocorre uma das operações do CRUD nos objetos monitorados pelo Webhook, uma requisição é enviada para a "payload_url" com informações sobre a operação que foi realizada (Post, Put, Delete, Update) e os dados do objeto modificado. A *View* é responsável por receber essa requisição, recuperar o objeto do banco de dados e enviá-lo para o ServiceNow CMDB conforme código 6.

Código 6. Envio do objeto *Location* para o ServiceNow

```
if whookBody['event'] == "created":
    if model == "site":
        netbox_site = Site.objects.get(pk=obj_id)
        result = webhook_cmdb.add_update_snow(
        "Site",
        netbox_site,
        user_credentials,
        cmdb_model_func= SnowCmdbProxy.CmnLocation.from_site,
        cmdb_save_func= SnowCmdbProxy.save_cmn_location,
        )
        return result
```

Fonte: De autoria própria (2023).

Todos os outros objetos e operações seguem a mesma estrutura do código da figura acima, a *View* pega o objeto do banco através do seu ID e depois envia a requisição para o ServiceNow CMDB.

2.4.3 Manual batch

A funcionalidade de "manual batch" permite que o usuário faça a replicação de dados de forma manual. Ao contrário da replicação automática, que replica apenas as informações mais recentes, o manual batch replica todos os dados dos objetos selecionados. Para realizar essa tarefa, essa funcionalidade utiliza os "workers" do NetBox, que são processos que geralmente são executados em segundo plano para lidar com tarefas demoradas ou bloqueantes.

No entanto, como o processo manual batch pode enviar uma grande quantidade de dados, é necessário que ele seja executado por um dos *workers* do NetBox (CÓDIGO 7 e 8). Isso ocorre porque essa funcionalidade pode sobrecarregar a capacidade de processamento do sistema, o que pode afetar negativamente a sua performance. Portanto, a utilização dos workers permite que a replicação dos dados seja feita de forma mais segura e eficiente.

Código 7. *Manual batch* dos objetos *Site*

```
if site == "true":
    get_queue("default").enqueue(
        worker.manual_batch,
        dry_run=dry_run,
        credentials=user_credentials,
        model_name="Site",
        model_object=Site,
        cmdb_model_func= SnowCmdbProxy.CmnLocation.from_site,
        cmdb_save_func= SnowCmdbProxy.save_cmn_location
        )
```

Fonte: De autoria própria (2023).

Código 8. Função que roda no *worker*

```
@job("default")
def manual_batch(dry_run,model_name, model_object, credentials, cmdb_model_func, cmdb_save_func):
    netbox_data = model_object.objects.all()
    credential: SnowCmdbProxy.Credentials = SnowCmdbProxy.Credentials(credentials.username, credentials.password, credentials.cmdb_url)
    logger.info("- Snow CMDB - START: Manual replication  of %s.", str(model_name))
    for object in netbox_data:
        logger.info("- Snow CMDB - Replicating object %s with name: %s.", str(model_name), str(object.name))
        if dry_run == "false":
            try:
                cmdb_save_func(credential, cmdb_model_func(object))
                logger.info("- Snow CMDB - Success replicating the object %s with name: %s.", str(model_name), str(object.name))
            except:
                logger.info("- Snow CMDB - Error replicating the object %s with name: %s.", str(model_name), str(object.name))
    logger.info("- Snow CMDB - End of replicating of object %s.", str(model_name))
```

Fonte: De autoria própria (2023).

2.4.4 Dry run

O dry run é uma funcionalidade que permite ao usuário visualizar uma prévia dos dados que serão replicados por meio dos logs. É possível identificar a linha que contém o trecho “if dry_run == ‘false’”. Essa linha verifica se a opção de dry run foi selecionada. Caso não tenha sido, o sistema prossegue com o manual batch normalmente. Porém, se o dry run tiver sido ativado, o sistema não executará a última etapa do manual batch - que consiste no envio dos dados - e, em vez disso, apenas registrará as informações nos logs (CÓDIGO 9). Essa funcionalidade é útil para verificar previamente os dados que serão replicados e evitar possíveis problemas ou erros.

Código 9. *Feature dry run* em funcionamento

```
worker_1    | 17:09:47 INFO - Snow CMDB - START: Manual replication  of Site. - rq.worker - worker - /source/netbox_snow_cmdb/worker.py:12
worker_1    | 17:09:47 INFO - Snow CMDB - Replicating object Site with name: [redacted]. - rq.worker - worker - /source/netbox_snow_cmdb/worker.py:14
worker_1    | 17:09:47 INFO - Snow CMDB - Replicating object Site with name: instagram. - rq.worker - worker - /source/netbox_snow_cmdb/worker.py:14
worker_1    | 17:09:47 INFO - Snow CMDB - Replicating object Site with name: teams. - rq.worker - worker - /source/netbox_snow_cmdb/worker.py:14
worker_1    | 17:09:47 INFO - Snow CMDB - End of replicating of object Site. - rq.worker - worker - /source/netbox_snow_cmdb/worker.py:21
```

Fonte: De autoria própria (2023).

2.5 Contribuição e dificuldades encontradas

O sistema ainda é uma versão inicial que será apresentada à liderança da empresa e, se aprovada, passará por mais algumas etapas de ajustes antes de entrar em ambiente de produção. Os ajustes ainda não estão completamente definidos devido a espera pela resposta da empresa, mas dentre os ajustes confirmados estão a melhoria na parte visual, replicação de todos os objetos no NetBox e replicação seletiva dos objetos.

Na minha percepção, o impacto do projeto é significativo, considerando a importância e a magnitude da integração do NetBox com um sistema tão relevante como o ServiceNow. Espero que a integração desperte o interesse da liderança em prosseguir com o projeto. Estou otimista com relação ao potencial deste projeto e espero que, com os ajustes necessários, ele possa ser implementado com sucesso e atender às expectativas da empresa.

A primeira dificuldade que encontrei ao trabalhar no sistema foi que originalmente haveria vários desenvolvedores envolvidos no projeto, mas, infelizmente, devido a outros projetos que exigiam mais atenção, a equipe foi redirecionada e acabei sendo o único programador trabalhando no sistema. Essa situação me colocou em uma posição desafiadora, pois tive que lidar com ambos os aspectos, front-end e back-end, sendo que eu tinha pouco conhecimento em front-end. Apesar desse desafio, consegui entregar um bom trabalho.

Embora eu tenha uma boa experiência em Python, nunca havia trabalhado com o framework Django antes, então precisei aprender sobre ele enquanto desenvolvia o projeto. À medida que fui desenvolvendo o sistema, comecei a me familiarizar mais com o framework e a utilizá-lo de forma mais eficiente.

Durante o desenvolvimento, enfrentei outra dificuldade relacionada à comunicação. Trabalhando sozinho, em alguns momentos, fiquei preso sem saber quais seriam os próximos passos a serem dados e tive dificuldades para comunicar de forma clara quaisquer problemas ou bloqueios que estava enfrentando. Isso me fez refletir sobre a importância da comunicação em um projeto, especialmente quando se trabalha em uma equipe reduzida ou individualmente.

No início da carreira profissional, especialmente como estagiário, muitas dificuldades surgem. No meu caso, uma das maiores dificuldades foi a comunicação. No entanto, é importante lembrar que essa fase inicial é de aprendizado e que é fundamental ser transparente com a empresa, informando quando você não se sente confortável.

3 Impactos da sua formação no seu trabalho

Um dos assuntos no qual estudei e que também utilizei em boa parte dos projetos da universidade foi o *framework* Scrum. O Scrum é uma estrutura usada no gerenciamento de projetos Agile que enfatiza colaboração, flexibilidade e melhoria contínua. Ele é projetado para ajudar as equipes a desenvolver, entregar e sustentar produtos complexos.

Ao longo de todos os projetos nos quais participei na empresa, o Scrum foi o método utilizado para gerenciar o processo de desenvolvimento. Além disso, a resolução de problemas também teve uma influência significativa no dia a dia da empresa. No projeto em que trabalhei recentemente, a habilidade de resolver problemas foi essencial em muitos aspectos do desenvolvimento.

No frontend, por exemplo, encontrei dificuldades com o Django e precisei utilizar jQuery e AJAX para solucionar o problema e tornar o sistema mais amigável para o usuário. No backend, o desenvolvimento do webhook também foi um desafio que exigiu muita habilidade em resolução de problemas.

Outro ponto que me ajudou a superar esses desafios foi o conhecimento de diversas linguagens, recursos e conceitos específicos, como a linguagem de programação Python, banco de dados, orientação a objetos e AWS. Essas habilidades foram valiosas em muitos projetos e me permitiram trabalhar com mais eficiência e precisão.

Em resumo, a habilidade de resolver problemas e a capacidade de aprender e dominar novas linguagens e recursos foram fundamentais para o sucesso do projeto em que trabalhei.

4 Conclusão

O desenvolvimento de projetos de software bem-sucedidos requer um conjunto de habilidades que vão além das habilidades técnicas e práticas, como a capacidade de se comunicar efetivamente, trabalhar em equipe e liderar. Além disso, a capacidade de resolver problemas e se adaptar a novos desafios e conhecimentos em metodologias ágeis são essenciais para um trabalho de qualidade e para atender às expectativas da empresa.

A integração de sistemas também é crucial para otimizar processos e aumentar a produtividade da equipe. A capacidade de integrar diferentes sistemas e tecnologias é uma habilidade valiosa e necessária para o desenvolvimento de software em um ambiente de negócios cada vez mais complexo.

No entanto, a criação de valor em um software pode ser um desafio complexo e delicado. É

necessário ter conhecimentos e experiência em tecnologias específicas, bem como em metodologias de desenvolvimento de software, para desenvolver soluções inovadoras que atendam às expectativas do cliente.

O problema em agregar mais valor a um software é algo delicado e difícil de se pensar. A integração do Netbox com o ServiceNow foi algo interessante e difícil de desenvolver, principalmente por ter pouca experiência com as tecnologias. Embora o projeto ainda esteja sendo analisado pela empresa, a solução proposta no meu ponto de vista foi muito bem elaborada e desenvolvida, o *plugin* possui uma interface simples e entrega as funcionalidades propostas.

No momento, a minha principal contribuição para a empresa foi o desenvolvimento desse projeto, pois é através dele que a liderança vai decidir se vai seguir ou não com o desenvolvimento dele. Além disso, estou envolvido em outro projeto que, apesar de ainda estar na fase de integração, parece ser muito empolgante. Tenho grandes expectativas de adquirir valiosa experiência e progredir profissionalmente dentro da empresa.

Sendo uma empresa de renome internacional, tanto em termos de clientes como de colaboradores, tenho a oportunidade de interagir constantemente com pessoas de diversas culturas e com variadas experiências. Tenho um grande interesse no mercado global e espero que minha participação e envolvimento nas atividades da empresa me permitam adquirir experiência de trabalho em outros países.

Referências

Osvaldo, José. Medeiros, Gildo. 2005. ABORDAGEM SISTÊMICA PARA INTEGRAÇÃO ENTRE SISTEMAS DE INFORMAÇÃO E SUA IMPORTÂNCIA À GESTÃO DA OPERAÇÃO: ANÁLISE DO CASO GVT. **GESTÃO & PRODUÇÃO.v.13, n.1, p.105-116, jan.-abr. 2006**. Disponível em: https://www.scielo.br/j/gp/a/Tz6ccWKtt4hqkWp9MwGTW9q/?lang=pt&format=pdf

Bai, Hao. 2022. **GoAutoBash: Golang-based Multi-Thread Automatic Pull-Execute Framework with GitHub Webhooks And Queuing Strategy**. Disponível em: https://arxiv.org/ftp/arxiv/papers/2206/2206.06401.pdf

Kumar, Rakesh. Gore, Himanshu. Singh, Ashutosh. Pratap, Arnav. 2021.Django Web Development Simple & Fast. **INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS(IJCRT)**. Disponível em: https://ijcrt.org/papers/IJCRT2105197.pdf

NetBox (2020). Custom plugins. **NETBOX DOCUMENTATION**. Disponível em: https://netbox.readthedocs.io/en/stable/plugins/.

DJANGO (2023). The Django MTV Development Pattern. **DJANGO DOCUMENTATION**. Disponível em: https://docs.djangoproject.com/en/4.2/

ITSM (2021). IT Service Management (ITSM). **SERVICENOW**. Disponível em: https://www.servicenow.com/products/it-service-management.html.

SN (2023). Configuration Management Database (CMDB). **SERVICENOW**. Disponível em: https://www.servicenow.com/products/servicenow-platform/configuration-management-database.html

MuleSoft. How to articulate the value of integration. **MYECOLE**. Disponível em: https://www.myecole.it/biblio/wp-content/uploads/2020/11/1_DB_13_MVC_MS-value-of-integration.pdf

Peerspot. ServiceNow CMDB Reviews. **Peerspot**. Disponível em: https://www.peerspot.com/products/servicenow-cmdb-reviews