

Samuell Henrique de Magalhães Perpétuo

***Software* Supervisorio *Web* para Integração com  
Simulador 3D: Treinamento Remoto para  
Operadores de Caminhão Fora-de-estrada**

Brasil

Julho de 2024

Samuell Henrique de Magalhães Perpétuo

***Software Supervisorio Web para Integração com  
Simulador 3D: Treinamento Remoto para Operadores de  
Caminhão Fora-de-estrada***

Dissertação submetida ao Programa de Pós-Graduação em Ciência e Tecnologia da Computação da Universidade Federal de Itajubá como requisito parcial para a obtenção do título de Mestre em Ciência e Tecnologia da Computação.

Área de Concentração: Matemática da Computação

Universidade Federal de Itajubá – Unifei

Programa de Pós-Graduação em Ciência e Tecnologia da Computação

Orientador: Prof. Dr. Giovani Bernardes Vitor

Coorientador: Prof. Dr. Wendell Fioravante da Silva Diniz

Brasil

Julho de 2024

# Resumo

A Indústria 4.0, caracterizada pela integração de tecnologias digitais aos processos industriais, vem transformando e impactando também o setor de capacitação e treinamento para operadores de caminhão fora-de-estrada na área da mineração. Dentre os desafios existentes, a integração de um *software web* para supervisionar simuladores 3D de caminhão fora-de-estrada constitui uma necessidade para o setor. Busca-se uma solução que proporcione maior flexibilidade e acessibilidade para as plataformas de *hardware* e *software* que executam o sistema de simulação em diferentes regiões. Neste sentido, o presente trabalho apresenta o desenvolvimento de um sistema supervisor *web* para gerenciar um sistema de simulação. O supervisor será responsável pela persistência dos dados em banco de dados e comunicará com o simulador em tempo de execução. A abordagem se deu utilizando uma camada de *back-end* que oferece uma API e uma camada de *front-end* que oferece a interface do usuário. A integração com o simulador se deu através do *framework* ROS2. Como resultados, o trabalho apresenta uma interface amigável, correta troca de mensagens entre os subsistemas supervisor e simulador, bem como o armazenamento dos dados de simulação dos operadores para acompanhamento de desempenho e aprendizado.

**Palavras-chave:** Desenvolvimento *web*. Sistema supervisor. Caminhão fora-de-estrada.

# Abstract

Industry 4.0, characterized by the integration of digital technologies into industrial processes, has also been transforming and impacting the training and qualification sector for off-road truck operators in mining. Among the existing challenges, the integration of web software to supervise 3D off-road truck simulators is a necessity for the sector. A solution is sought that provides greater flexibility and accessibility for the hardware and software platforms that run the simulation system in different regions. In this sense, this work presents the development of a web supervisory system to manage a simulation system. The supervisory system will be responsible for persisting data in a database and will communicate with the simulator at runtime. The approach was made using a back-end layer that offers an API and a front-end layer that offers the user interface. The integration with the simulator was done through the ROS2 framework. As a result, the work presents a user-friendly interface, correct exchange of messages between the supervisory and simulator subsystems, as well as the storage of operator simulation data for performance monitoring and learning.

**Keywords:** Web development. Supervisory system. Off-road truck.

# Lista de ilustrações

Figura 1 – Estação do treinador em destaque no conjunto do simulador. . . . .	10
Figura 2 – Arquitetura do <i>hardware</i> do simulador. . . . .	11
Figura 3 – Sistema supervisorio na indústria. . . . .	18
Figura 4 – Comunicação entre <i>front-end</i> e <i>back-end</i> . . . . .	19
Figura 5 – Padrão arquitetural MVC. . . . .	20
Figura 6 – Tópicos e serviços do ROS. . . . .	23
Figura 7 – <i>Frameworks</i> usados no desenvolvimento. . . . .	25
Figura 8 – Arquitetura do sistema supervisorio. . . . .	26
Figura 9 – Arquitetura recomendada para sistemas Django + PostgreSQL . . . . .	27
Figura 10 – Arquitetura do módulo de integração com o sistema de simulação. . . . .	27
Figura 11 – DER - Projeto Conceitual do BD. . . . .	28
Figura 12 – Casos de uso. . . . .	30
Figura 13 – Diagrama de Classes. . . . .	31
Figura 14 – Casos de uso da integração supervisorio x simulador. . . . .	40
Figura 15 – Casos de uso da geração das tarefas de simulação. . . . .	41
Figura 16 – Tela de listagem dos cenários cadastrados. . . . .	43
Figura 17 – Tela de acesso aos detalhes do cenário. . . . .	44
Figura 18 – Tela de administração de currículos. . . . .	44
Figura 19 – Tela de administração dos erros do operador. . . . .	45
Figura 20 – Tela de administração dos eventos de simulação. . . . .	46
Figura 21 – Tela de visualização dos sítios instalados. . . . .	46
Figura 22 – Tela de administração dos operadores cadastrados. . . . .	47
Figura 23 – Tela de administração dos grupos de operadores. . . . .	47
Figura 24 – Tela de inicialização da sessão de simulação. . . . .	48
Figura 25 – Tela com as simulações realizadas. . . . .	48
Figura 26 – Tela de resultado da sessão, mostrando detalhes, erros, eventos e dados do currículo em diferentes abas. . . . .	49
Figura 27 – Tela de acompanhamento de simulação em tempo de execução. . . . .	50
Figura 28 – Aba Geral da tela de detalhamento do cenário. . . . .	50
Figura 29 – Aba Opções da tela de detalhamento do cenário. . . . .	51
Figura 30 – Aba Requisitos da tela de detalhamento do cenário. . . . .	51
Figura 31 – Aba Condições de Operação da tela de detalhamento do cenário. . . . .	52
Figura 32 – Aba Rotas da tela de detalhamento do cenário. . . . .	52
Figura 33 – Aba Cargas da tela de detalhamento do cenário. . . . .	53
Figura 34 – Aba Eventos da tela de detalhamento do cenário. . . . .	53
Figura 35 – Aba Assistentes da tela de detalhamento do cenário. . . . .	54

Figura 36 – Aba Benchmarks da tela de detalhamento do cenário. . . . .	54
Figura 37 – Aba Erros de operação da tela de detalhamento do cenário. . . . .	55
Figura 38 – Aba Avaliação da tela de detalhamento do cenário. . . . .	55
Figura 39 – API com os <i>endpoints</i> do <i>back-end</i> . . . . .	56

# Lista de tabelas

Tabela 1 – Resumo dos requisitos e regras de negócio. . . . .	32
Tabela 2 – Resumo dos requisitos relacionados aos erros. . . . .	36
Tabela 3 – Resumo dos requisitos relacionados aos eventos. . . . .	37
Tabela 4 – Resumo dos requisitos da integração supervisorio x simulador. . . . .	39
Tabela 5 – Resumo dos requisitos da geração das tarefas de simulação. . . . .	41

# Lista de abreviaturas e siglas

API	<i>Application Programming Interface</i>
CRUD	<i>Create, Read, Update, Delete</i>
IEEE	<i>Institute of Electrical and Electronics Engineers</i>
GPS	<i>Global Positioning System</i>
HTML	<i>Hyper Text Markup Language</i>
HTTP	<i>Hyper Text Transference Protocol</i>
IoT	<i>Internet of Things</i>
MVC	<i>Model-View-Controller</i>
ORM	<i>Object Relational Mapping</i>
ROS	<i>Robot Operating System</i>
SGBD	Sistema Gerenciador de Banco de Dados
SQL	<i>Structure Query Language</i>
TIC	Tecnologia da Informação e Comunicação
URL	<i>Uniform Resource Locator</i>
WEB	<i>World Wide Web</i>

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>10</b>
1.1	Contexto e motivação	12
1.2	Problema	12
1.3	Objetivos	14
1.3.1	Objetivo Geral	14
1.3.2	Objetivos Específicos	14
1.4	Justificativa	14
1.5	Organização do Trabalho	15
<b>2</b>	<b>REVISÃO BIBLIOGRÁFICA</b>	<b>16</b>
2.1	Trabalhos Relacionados	16
2.2	Definições e Conceitos	17
2.3	<i>Robot Operating System</i>	22
<b>3</b>	<b>DESENVOLVIMENTO</b>	<b>24</b>
3.1	Ferramentas Utilizadas	24
3.2	Arquitetura do Sistema Supervisório	26
3.3	Projeto do Banco de Dados	28
3.4	Descrição das Funcionalidades	29
3.5	Requisitos	32
3.5.1	Requisitos funcionais	33
3.5.2	Requisitos não-funcionais	34
3.5.3	Regras de negócio	35
3.5.4	Erros do operador	35
3.5.5	Eventos de simulação	37
3.5.6	Integração supervisório x simulador	38
3.5.7	Geração das tarefas de simulação	40
<b>4</b>	<b>RESULTADOS</b>	<b>42</b>
4.1	Resultados Qualitativos	42
4.1.1	RF-1 – Administrar cenários	42
4.1.2	RF-2 – Administrar currículos	43
4.1.3	RF-3 – Administrar erros	44
4.1.4	RF-4 – Administrar eventos	45
4.1.5	RF-5 – Visualizar sítios	45
4.1.6	RF-6 – Administrar operadores	46

4.1.7	RF-7 – Administrar grupos de operadores . . . . .	47
4.1.8	RF-8 – Administrar simulações . . . . .	48
4.1.9	RF-9 – Fazer backup, importar e exportar e RF-10 Gerar relatórios . . . . .	49
4.1.10	Integração supervisorio x simulador e geração das tarefas de simulação . . . . .	49
<b>4.2</b>	<b>Resultados Quantitativos . . . . .</b>	<b>56</b>
<b>5</b>	<b>CONCLUSÕES . . . . .</b>	<b>57</b>
	<b>REFERÊNCIAS . . . . .</b>	<b>58</b>

# 1 Introdução

Este trabalho faz parte de uma parceria entre a mineradora multinacional brasileira Vale S.A. e a Universidade Federal de Itajubá – Unifei – Campus Itabira. O objetivo dessa parceria é de desenvolver um simulador computacional em três dimensões (3D) de caminhão fora-de-estrada capaz de auxiliar na capacitação dos operadores destas máquinas.

A mineradora possui um parque tecnológico com alguns simuladores distribuídos entre os grandes complexos minerários, conforme exemplo da Figura 1. Alguns destes ativos estão em desuso. Assim, a parceria com a Universidade é uma forma de buscar desenvolver um conjunto de *hardware* disponível com tecnologia nacional, buscando um horizonte de customizações e *upgrades* a curto, médio e longo prazo.

O projeto é considerado de alta complexidade devido ao desenvolvimento de *software* específico e customizado, além da integração do mesmo com o conjunto de *hardware*. Este conjunto disponibilizado pela mineradora foi unido com partes adquiridas ou desenvolvidas pela equipe da frente de *hardware* do projeto.

Figura 1 – Estação do treinador em destaque no conjunto do simulador.

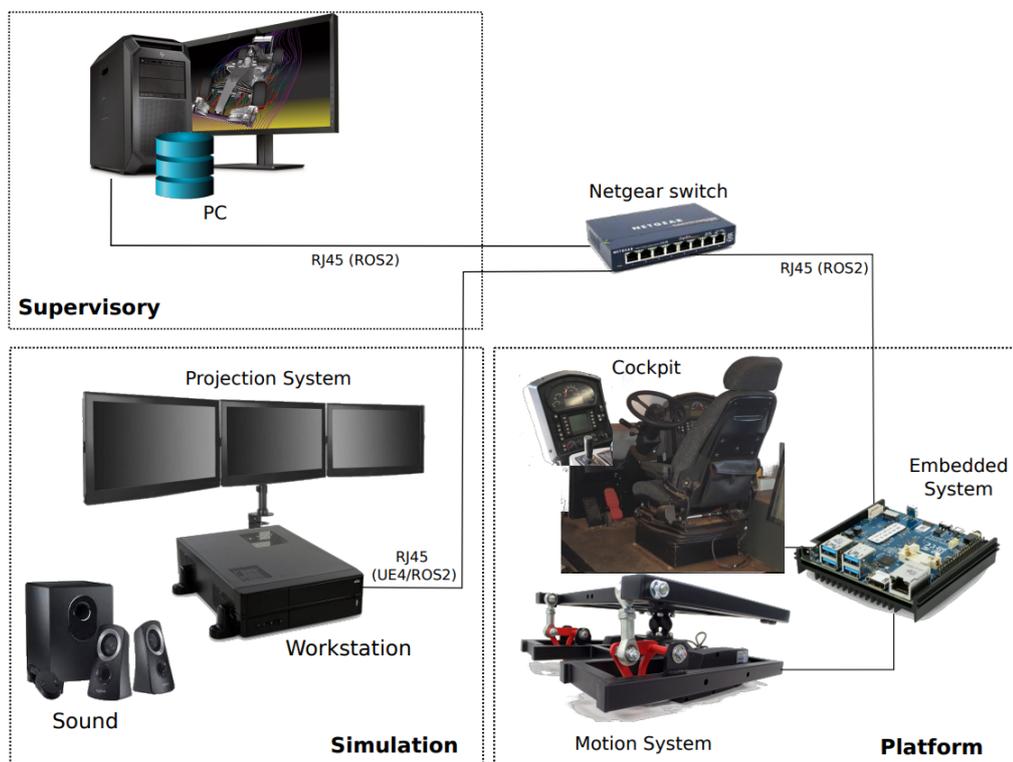


Fonte: [Silva \(2021\)](#).

Devido a complexidade apresentada, o projeto foi decomposto em quatro módulos funcionais de forma a se obter um melhor gerenciamento das atividades. O módulo no qual se enquadra o escopo deste trabalho tem como objetivo o desenvolvimento de um sistema *web* supervisor. Este sistema será usado para gerenciar e controlar as atividades de treinamento no ambiente virtual de simulação. Também são atribuições do sistema o registro de todas as informações em banco de dados, a interação em tempo de execução com o sistema de simulação e a interface principal com o usuário.

O sistema supervisório é hospedado em computador dedicado ao mesmo, com um monitor próprio separado dos monitores utilizados para exibição da simulação. Este computador é a plataforma de trabalho do instrutor, como pode ser visto na Figura 1, na qual está destacada a estação de trabalho do instrutor no conjunto do simulador. Juntamente a Figura 2 é possível entender melhor como o sistema supervisório (*Supervisory* na figura) se comunica com os outros elementos do simulador via rede utilizando o *framework* ROS (*Robot Operating System*). Essa comunicação e os demais detalhes da solução serão abordados ao longo deste trabalho.

Figura 2 – Arquitetura do *hardware* do simulador.



Fonte: Vitor e Diniz (2023).

Como apresentado, o presente trabalho está associado a um contexto maior, já que faz parte de um projeto em parceria entre a Unifei e a mineradora Vale. Assim, o desenvolvimento do sistema completo de simulação, bem como o desenvolvimento de suas partes componentes, possui questões de confidencialidade e segurança da informação. Devido à estas questões, módulos específicos do sistema supervisório não serão abordados de maneira detalhada neste trabalho para não expor as informações que devem ficar resguardadas aos envolvidos no projeto.

## 1.1 Contexto e motivação

A Quarta Revolução Industrial traz mudanças nos paradigmas de produção e é impulsionada pela integração de tecnologias digitais como a Internet das Coisas (IoT), inteligência artificial, automação, análise de *Big Data* e robótica (RIBEIRO et al., 2022; LööW; ABRAHAMSSON; JOHANSSON, 2019; CHEN; ZHOU, 2018; CRUZARA et al., 2020). Embora a Indústria 4.0 seja normalmente associada à produção e às fábricas inteligentes, é igualmente transformadora no contexto da indústria da mineração.

As organizações da área da mineração buscam reduzir os riscos associados às suas atividades. Um desses riscos está associado ao treinamento de mão-de-obra para operar máquinas amplamente utilizadas na mineração. A Tecnologia da Informação e Comunicação (TIC) pode ajudar na redução desses riscos à medida que pode ser aplicada no desenvolvimento de soluções para auxiliar na capacitação de operadores de caminhões fora-de-estrada nas minas.

Um exemplo interessante é o da indústria da aviação, na qual o treinamento dos pilotos de aeronaves inclui várias horas de simulações em sistemas computacionais. Assim, o uso de simuladores é tido como uma boa prática ao se capacitar colaboradores.

A Vale, financiadora do projeto do qual este trabalho faz parte, investe em tecnologia de simulação desde 2005, usando-a para formar e reciclar seus operadores nos seus processos de treinamento. Com isso, a mineradora busca proporcionar um ambiente seguro e controlado, capaz de reproduzir o ambiente da mina dentro de plataformas de simulação. Com a imersão do operador em sistemas simuladores de qualidade é possível trabalhar diversos aspectos da operação real das máquinas.

Especificamente sobre o sistema supervisor, o mesmo se configura como essencial para a utilização do simulador. O supervisor é a principal interface do sistema com o usuário. Ele oferece as ferramentas sem as quais não seria possível que o usuário utilizasse o simulador sem dominar todos os aspectos do seu funcionamento. Assim, abstraindo boa parte dos demais módulos do simulador, além de cumprir com suas responsabilidades, o supervisor fornece ao usuário as funcionalidades necessárias para a utilização efetiva do sistema de simulação como um todo.

## 1.2 Problema

O problema abordado neste trabalho está relacionado com o treinamento de operadores de caminhão fora-de-estrada na mineração a céu aberto, processo crucial para a segurança, produtividade e eficiência das operações na indústria da mineração (GERMANY, 2002).

A mineração a céu aberto é um método de extração de minerais que se caracteriza

pela remoção de camadas de solo e rocha para alcançar o depósito mineral. Essa atividade exige o uso de equipamentos pesados, como caminhões fora-de-estrada, que são responsáveis pelo transporte do material escavado. A operação segura e eficiente dessas máquinas depende da qualificação dos operadores. O treinamento adequado garante que os operadores estejam aptos a manejar os equipamentos com segurança, produtividade e responsabilidade ambiental (GERMANY, 2002).

O treinamento do operador de caminhão fora-de-estrada geralmente segue uma estrutura modular, abrangendo os seguintes tópicos entre conhecimentos teóricos e práticos (SILVA, 2021):

- Noções básicas de mineração a céu aberto: Abrange o ciclo de mineração, tipos de minérios, métodos de extração e equipamentos utilizados;
- Características dos caminhões fora-de-estrada: Detalhes dos componentes principais, sistemas hidráulicos, motor, transmissão e sistemas de segurança;
- Princípios de operação: Aborda os comandos, instrumentos, técnicas de direção, manobras e procedimentos de carga e descarga;
- Normas de segurança: Enfatiza os riscos inerentes à operação, uso de EPIs, procedimentos de trabalho seguros e medidas de prevenção de acidentes.
- Manutenção preventiva: Aborda os itens de verificação diária, lubrificação, troca de filtros e procedimentos básicos de reparo;
- Simuladores: Utilização de simuladores para familiarização com os comandos, instrumentos e situações de operação em um ambiente seguro;
- Treinamento em área controlada: Prática da operação em um local seguro e delimitado, com supervisão constante de instrutores experientes;
- Operação em campo real: Gradual progressão para a operação em condições reais de mineração, sob acompanhamento e avaliação.

Desse modo, este trabalho pretende contribuir para o aprimoramento do uso de simuladores nos processos de treinamento de operadores de caminhão fora-de-estrada, que se configura como um investimento fundamental para a indústria de mineração a céu aberto. Operadores bem treinados garantem a segurança das operações, aumentam a produtividade e reduzem custos com manutenção e reparos. Além disso, contribuem para a preservação do meio ambiente e para a imagem positiva da empresa junto a sociedade.

## 1.3 Objetivos

Os objetivos deste trabalho estão relacionados ao desenvolvimento de *software*. Assim, para definir os objetivos, partiu-se da compreensão de desenvolvimento como um processo de produção de *software* (aplicação e documentação relacionada) que tradicionalmente envolve etapas de concepção, *design*, codificação, testes, implantação e manutenção.

### 1.3.1 Objetivo Geral

O objetivo geral é o desenvolvimento de um sistema *web* supervisor para compor um sistema de simulação de caminhão fora-de-estrada de mineração.

### 1.3.2 Objetivos Específicos

Os objetivos específicos são:

- Levantar e analisar os requisitos funcionais e não-funcionais;
- Projetar o *software*;
- Desenvolver as funcionalidades para listar, criar, atualizar e excluir cenários, currículos, operadores, grupos de operadores, erros operacionais e eventos, relatórios das simulações realizadas, histórico de simulações do operador;
- Implementar a integração do sistema e realizar os testes necessários.

## 1.4 Justificativa

No contexto específico da capacitação de colaboradores da indústria da mineração, pode-se citar o relatório técnico intitulado “A mineração no Brasil” produzido por [Germany \(2002\)](#). O relatório foi elaborado com base na experiência de mais de 42 anos do autor na atividade mineradora. O pesquisador utilizou a cooperação de técnicos especializados, entrevistas e consultas pessoais e à literatura existente. O relatório aponta para a necessidade de se treinar os colaboradores de forma mais eficaz, afim de que se possa tirar o maior benefício do seu trabalho com resultados mais positivos para a indústria.

Apesar da evolução tecnológica, principalmente no que diz respeito à capacidade de processamento dos computadores, os simuladores não evoluíram de forma significativa. Em geral, as principais evoluções perceptíveis estão relacionadas às capacidades gráficas, nas representações dinâmicas do sistema dentro do ambiente simulado e nas filosofias empregadas no processo de treinamento do operador. Ainda hoje no Brasil não foi possível

mapear na literatura, estado da arte, algum tipo de desenvolvimento relacionado à plataformas de simulação voltadas principalmente para simulações de caminhões fora-de-estrada, constituindo-se um campo em aberto para o desenvolvimento e inovação em nível nacional.

A pesquisa por trabalhos similares apresentada resumidamente na seção 2.1 permitiu verificar que há na literatura referências a ferramentas desenvolvidas especificamente para cada contexto de aplicação. Essas ferramentas possuem interface de monitoramento do progresso de simulação e gestão dos treinamentos de maneira unificada com a simulação. Desse modo, o presente trabalho inova ao apresentar uma solução que desenvolve uma ferramenta modular e remota que se comunica com o sistema de simulação de modo a integrá-lo, porém, sendo apartado do mesmo.

## 1.5 Organização do Trabalho

Após a introdução do Capítulo 1, que elucidou o contexto, a motivação, o problema, os objetivos e a justificativa, o trabalho segue apresentando uma revisão bibliográfica no Capítulo 2, trazendo os trabalhos relacionados e o embasamento teórico, a fim de reunir os conceitos necessários para o entendimento do desenvolvimento apresentado no Capítulo 3 e os resultados no Capítulo 4. Finalmente, o Capítulo 5 apresenta as considerações finais, com as limitações, contribuições e possíveis trabalhos futuros.

## 2 Revisão Bibliográfica

Para cumprir com a proposta do trabalho de desenvolver um sistema *web* como supervisor de um sistema de simulação é importante entender do que se trata este tipo de sistemas de informação. Ao desenvolver um sistema é comum revisar definições relacionadas ao trabalho a ser realizado, classificar o sistema almejado e suas características principais. À partir disso, os pesquisadores buscam verificar qual a melhor forma de desenvolver a solução proposta, o que é feito neste capítulo de revisão da literatura relacionada ao tema do trabalho.

### 2.1 Trabalhos Relacionados

Dentro das práticas da Indústria 4.0, a aplicação de tecnologias na integração de informações, modelagem digital, plataformas de simulação, tecnologias de nuvem e aprendizado de máquina são amplamente difundidas. Diversas organizações aplicam essas tecnologias em setores como transporte, engenharia de energia, produção e aviação, gerando otimização de recursos e a consequente economia de capital (ROMERAL *et al.*, 2021; NGOC; LASA; IRIARTE, 2022; SOUZA *et al.*, 2021; BIGLIARDI; FILIPPELLI; TAGLIENTE, 2022; YANG; GU, 2021).

A realidade virtual é uma das tecnologias cuja aplicação está crescendo rapidamente, utilizando o poder cada vez maior dos computadores para simular ambientes e situações do mundo real e imaginários com um alto grau de realismo e interatividade (BELOGLAZOV; PETROV; BAZHIN, 2020).

O trabalho de Wyk e Villiers (2009) traz um exemplo de aplicação, em que a segurança na indústria mineira sul-africana é a maior preocupação. Em média, um trabalhador morre por dia de trabalho e cerca de 16 ficam feridos em acidentes relacionados com minas naquele país. A formação inadequada ou insuficiente é frequentemente citada como a causa principal de muitas mortes na mineração em todo o mundo. Os autores entenderam que as ferramentas de formação baseadas na realidade virtual podem proporcionar uma exposição simulada a condições de trabalho do mundo real, sem os riscos associados.

O trabalho de McMahan *et al.* (2010) traz uma experiência na qual uma equipe de pesquisa desenvolveu aplicações de realidade virtual para melhorar o treinamento de trabalhadores que operam caminhões e sistemas de transporte na indústria de mineração. A eficácia e a relação custo-benefício destas ferramentas foram avaliadas empiricamente e com base nos resultados, as ferramentas de treinamento foram consideradas benéficas para o treinamento de inspeções pré-turno de caminhões de transporte e para o treinamento de

procedimentos operacionais e de segurança para sistemas de transporte.

O artigo [Jain et al. \(2023\)](#) também traz aplicação de ambientes virtuais para treinamento de operadores, sendo este aplicado na indústria da mineração de carvão. Os autores avaliaram a ferramenta de treinamento de forma positiva, ressaltando sua potencialidade de preservar vidas, evitando acidentes e capacitando os colaboradores a lidarem com situações desastrosas nas minas de carvão.

Em 2023, a IEEE realizou uma conferência sobre realidade virtual e interfaces de usuário 3D, na qual foram publicados trabalhos que discutem a evolução destas ferramentas. Neste sentido, pode-se citar o trabalho de [Yu et al. \(2023\)](#), apontando que o futuro desta área é o desenvolvimento de tecnologias cada vez mais sofisticadas, que apresentam funcionalidades mais avançadas, como treinamento remoto em realidade virtual, treinamento adaptativo e co-treinamento em realidade virtual.

## 2.2 Definições e Conceitos

É importante ressaltar as características do sistema supervisorio que definem a metodologia a ser utilizada para alcançar os objetivos deste trabalho. O sistema foi desenvolvido utilizando a plataforma *web*. Desse modo, é de suma importância trazer aqui algumas definições e conceitos, buscando entender as características do sistema almejado, bem como as especificidades do desenvolvimento da categoria de *software* trabalhada.

Um sistema supervisorio é um tipo de sistemas de informação. [Laudon e Laudon \(2011\)](#) definem sistemas de informação como um conjunto de elementos inter-relacionados que coleta, processa, armazena e distribui informações com o intuito de apoiar a tomada de decisão, a coordenação e o controle de uma organização. Os sistemas de informação auxiliam os gestores na hora de analisar problemas, obter uma visão mais ampla de assuntos complexos e até a criar novos produtos.

Um sistema supervisorio é frequentemente referido no contexto de sistemas de controle industrial como sendo uma estrutura de controle de alto nível que supervisiona vários controladores individuais ou malhas de controle. Esses sistemas não são necessariamente obrigatórios para a operação de cada controlador, mas fornecem ao usuário uma visão geral do processo da planta industrial e permite a operação integrada entre os controladores.

O termo “supervisorio” é usado para um Sistema de Controle de Supervisão e Aquisição de Dados (*Supervisory Control and Data Acquisition - SCADA*), que é uma classe de sistema usada no controle de processos. Os sistemas SCADA são normalmente empregados em aplicações menores e remotas, como transporte por dutos, distribuição de água ou sistemas de serviços públicos de águas residuais. Eles coletam e analisam dados, monitoram e controlam equipamentos e fornecem aos operadores uma visão abrangente do

status do sistema por meio de uma interface de usuário (TOTVS, 2023).

A Figura 3 ilustra o posicionamento dos sistemas supervisórios na indústria. Neste esquema, o supervisório encontra-se no terceiro nível da pirâmide de automação, acima dos dispositivos de campo, bem como dos equipamentos de controle. Assim, é frequente a aplicação deste tipo de sistema para realizar a interface entre os níveis organizacionais superiores com os equipamentos instalados em campo e operação da planta industrial.

Figura 3 – Sistema supervisório na indústria.



Fonte: Abacaxi Digital.

O sistema supervisório deste trabalho é aplicado em um simulador. Sistemas simuladores são utilizados para treinamento há muitos anos. Um dos primeiros relatos do uso desses remonta o treinamento de pilotos de aviação em 1910, no qual modelos de aeronaves eram presas ao chão e instrutores passavam orientações de como reagir em situações hipotéticas (PAGE, 2000). Desde então, os simuladores têm evoluído na imersão do operador em ambientes e situações específicas, se firmando como ferramentas essenciais para o treinamento e capacitação desses ou no desenvolvimento de novos equipamentos como forma de validar estruturas no seu estado de concepção, antes mesmo da montagem física de protótipos.

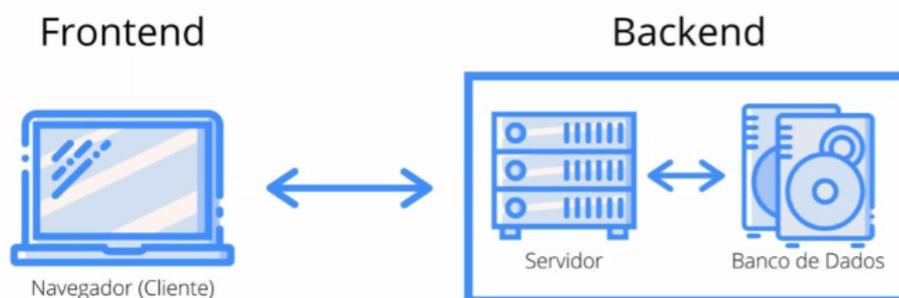
Acompanhando o desenvolvimento tecnológico trazido pelos computadores modernos, os simuladores foram desenvolvidos e adaptados de acordo com as necessidades dos usuários. Eles variaram de estruturas simples envolvendo apenas um computador pessoal, um monitor e controles, até sistemas sofisticados formados por modelos completos com

dinâmicas intrincadas, com representações gráficas de alto nível e bases de movimento.

O sistema supervisorio aqui apresentado foi concebido como um sistema *web*. A combinação de padrões arquitetônicos comprovados com *frameworks* avançados permite a criação de aplicações *web* eficientes e com alto desempenho. Este trabalho aplicou metodologia de desenvolvimento de sistemas *web* utilizando o padrão arquitetural MVC (*Model-View-Controller*), o *framework* Django REST Framework para construção de uma API no *back-end* e o *framework* Vue.js para criação das páginas responsáveis pela interface com o usuário no *front-end*.

No desenvolvimento *web*, o *front-end* é a parte visível de um site ou aplicativo, com a qual o usuário interage diretamente. Já o *back-end* é a parte que fica nos bastidores, responsável por toda a lógica e funcionamento do sistema. O *front-end* e o *back-end* trabalham juntos para oferecer uma experiência completa ao usuário. Quando um usuário interage com um elemento no *front-end*, como clicar em um botão ou preencher um formulário, uma solicitação é enviada ao *back-end*. O *back-end* processa essa solicitação, realiza as ações necessárias e envia uma resposta de volta para o *front-end*, que atualiza a interface para refletir as mudanças. Quando se fala em arquitetura cliente servidor, o *front-end* fica do lado do cliente, enquanto o *back-end* ficaria no lado do servidor, como esquematizado na Figura 4.

Figura 4 – Comunicação entre *front-end* e *back-end*.



Fonte: <<https://www.zup.com.br/>>.

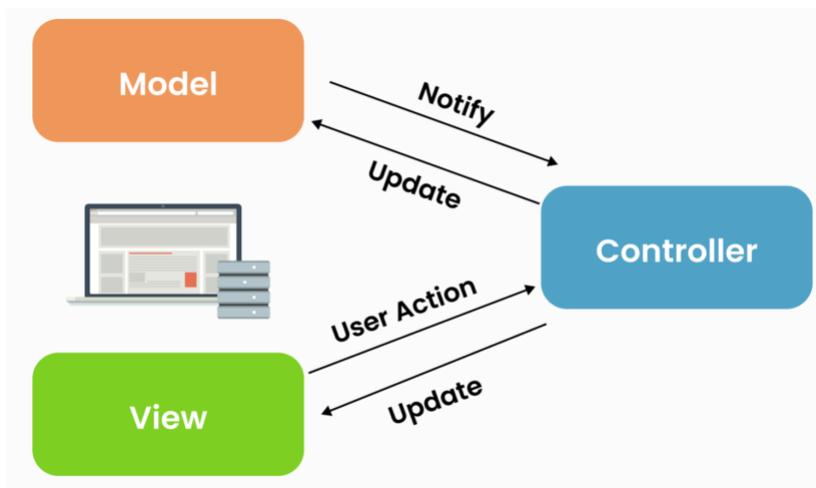
O padrão MVC (*Model-View-Controller*) é um paradigma de design de software amplamente utilizado para estruturar aplicações *web*. Diversos autores (FOWLER, 2003; LARMAN, 2004; SKOVGAARD, 2007) descrevem o padrão MVC e seus benefícios, como a separação de responsabilidades, a modularidade do código e a facilidade de manutenção. Essa separação facilita a reutilização de componentes, a testabilidade e a extensibilidade da aplicação (SKOVGAARD, 2007).

O modelo (*Model*) inclui todos os dados e lógica de negócios relacionada. Basicamente, ele representa os dados transferidos entre o controlador e a visão. Por exemplo, se o controlador ajuda a recuperar informações do cliente do banco de dados, ele manipula

os dados e os usa para renderizar os mesmos dados ou os envia de volta ao banco de dados. A visão (*View*) define a exibição ou interface do usuário do sistema. O controlador (*Controller*) é a interface entre o modelo e a visão e contém lógica de controle interpretando as entradas do usuário (NILAY, 2021).

Como pode ser visto no esquema da Figura 5, no padrão MVC a visão recebe as ações do usuário e transmite para o controlador, que atualiza o modelo, recebe o retorno do mesmo e então atualiza a visão, com a qual o usuário interage.

Figura 5 – Padrão arquitetural MVC.



Fonte: Nilay (2021).

A aplicação do modelo MVC neste trabalho está implícita ao utilizar o *framework* Django em conjunto com o Django REST Framework. Ambos são *frameworks* baseados na linguagem de programação Python e são empregados na construção de APIs (*Application Programming Interface*). Assefaw (2015) e Hyndman (2016) destacam a simplicidade e a flexibilidade do Django REST Framework, além de sua integração perfeita com o *framework* Django. Estes *frameworks* em conjunto fornecem ferramentas para serializar e desserializar dados, definir *endpoints*, autenticação e autorização, validação de entrada e documentação de APIs (STEVENSON, 2013).

Uma API é uma interface de programação de aplicativos, como o próprio nome sugere. Se um projeto tem algum código para ser usado por outra aplicação sem saber tudo sobre o código, recomenda-se o uso de APIs. As APIs fornecem uma camada de abstração e fornecem acesso ao código subjacente. Isso é muito útil ao trabalhar em projetos que envolvem camadas independentes, pois o desenvolvedor de uma das camadas pode usar o código de outra camada sem ter que se preocupar como a mesma foi implementada. Neste trabalho, por exemplo, a camada do *front-end* consome a API produzida no *back-end*.

O *front-end* deste trabalho foi implementado usando diversas tecnologias, das quais se ressalta o Vue. O Vue.js é um framework da linguagem JavaScript para construir

interfaces de usuário. Ele é construído em cima das linguagens padrão de desenvolvimento *web* – HTML, CSS e JavaScript – e provê uma forma de programação declarativa e baseada em componentes que auxilia na criação de interfaces de usuário de qualquer complexidade (YOU, 2024).

O termo componente não é exclusivo do Vue. Ele é comum em muitos frameworks de interface de usuário e representa uma porção de código independente, como um botão ou um *checkbox*. Componentes podem ser combinados para formar componentes maiores. Cada componente pode ser pensado de forma isolada e é comum organizá-los em uma estrutura hierarquizada do tipo árvore. O Vue implementa seu próprio modelo de componente, permitindo encapsular conteúdo personalizado e lógica em cada um deles (YOU, 2024).

Yu (2016) e Bojan (2017) discutem a popularidade do Vue.js devido à sua curva de aprendizado suave, à sua flexibilidade e à sua capacidade de criar interfaces *web* responsivas, combinando a simplicidade do HTML com a flexibilidade do JavaScript, permitindo a construção de interfaces de forma modular (BOJAN, 2017).

Responsividade se refere à habilidade de uma ferramenta de executar ações automaticamente em resposta às mudanças dos dados de entrada (YOU, 2024). Estes dados podem ser tanto as informações que usuário precisa quanto o tamanho do dispositivo. Com o Vue é possível criar facilmente uma interface que se adapta ao tamanho da tela em que está sendo visualizada, independentemente do dispositivo.

O desenvolvimento de sistemas *web* com MVC, Django REST Framework e Vue.js oferece uma solução robusta, escalável e eficiente para a criação de aplicações *web* modernas. Essa abordagem combina os benefícios do padrão MVC com a flexibilidade do Django REST Framework e do Vue.js, permitindo o desenvolvimento de sistemas *web* de alta qualidade. As principais vantagens dessa abordagem incluem:

- **Separação de responsabilidades:** O padrão MVC promove a separação de responsabilidades entre o modelo, a visão e o controlador, tornando o código mais modular e fácil de manter (SKOVGAARD, 2007).
- **Desenvolvimento rápido de APIs:** O Django REST Framework facilita o desenvolvimento de APIs RESTful robustas e escaláveis (STEVENSON, 2013).
- **Interfaces *web* responsivas:** O Vue.js permite a criação de interfaces *web* dinâmicas e responsivas com facilidade (BOJAN, 2017).

Diversos estudos de caso demonstram a efetividade da utilização de MVC, Django REST Framework e Vue.js no desenvolvimento de sistemas *web*. Por exemplo, Yu (2018) apresenta um sistema de gerenciamento de bibliotecas desenvolvido com essa *stack* tecnológica, enquanto Ali et al. (2019) propõem uma plataforma de *e-learning* baseada na mesma abordagem.

## 2.3 Robot Operating System

O sistema supervisorio deste trabalho utilizou o framework *Robot Operating System* (ROS) para se comunicar com o sistema de simulação via rede local cabeada. O ROS é um *framework* de *software* de código aberto amplamente utilizado para o desenvolvimento e a implementação de robôs. Sua arquitetura modular e distribuída facilita a criação de robôs robustos, escaláveis e autônomos (OPEN\_ROBOTICS, 2021). Esta seção tem como objetivo apresentar os pontos principais do ROS, incluindo seus conceitos básicos, arquitetura e principais funcionalidades.

Um robô é qualquer sistema que pode perceber seu ambiente, tomar decisões com base no estado do ambiente e executar as instruções geradas. Os robôs são usados em uma variedade de aplicações, incluindo a indústria, exploração espacial e até mesmo para realizar cirurgias meticulosas (HASAN, 2019). Eles também estão presentes na computação como aplicativos para realizar tarefas repetitivas. Robôs mecânicos diminui riscos associados a operações que envolvem acessar lugares perigosos ou inacessíveis aos seres humanos.

O ROS é um sistema de software que fornece uma estrutura para gerenciar a complexidade de sistemas robóticos. Ele fornece uma maneira padronizada de comunicar diferentes partes do sistema robótico, como sensores, atuadores e algoritmos de controle. O ROS também fornece uma biblioteca de ferramentas e pacotes de software que podem ser usados para construir sistemas que envolvem robôs. A arquitetura do ROS é composta por três elementos principais (OPEN\_ROBOTICS, 2021):

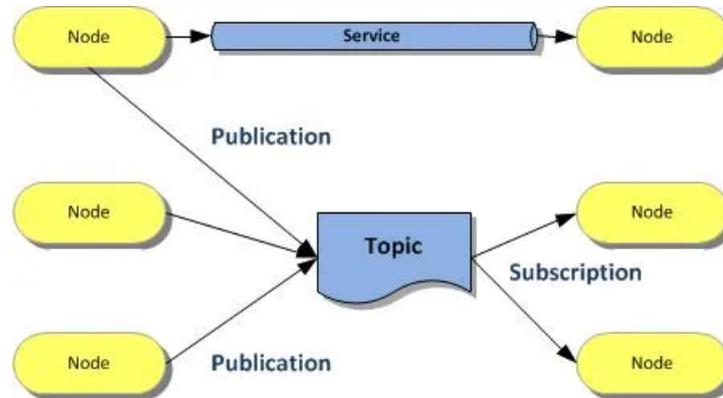
- **Nó:** Um processo individual que executa um ou mais pacotes ROS;
- **Tópico:** Um canal de comunicação pelo qual os nós publicam e recebem mensagens;
- **Mensagem:** Um tipo de dado estruturado que contém informações trocadas entre os nós.

O ROS é baseado em uma arquitetura distribuída que utiliza pacotes independentes para encapsular funcionalidades específicas. Esses pacotes podem ser facilmente instalados, atualizados e reutilizados em diferentes projetos que usam comunicação com robôs. A arquitetura de comunicação do ROS é caracterizada como pub/sub. Isso significa que os diferentes componentes do sistema robótico se comunicam publicando (pub) e assinando (sub) mensagens em tópicos. O ROS também usa um sistema de nomes para que os componentes possam se encontrar e se comunicar.

A Figura 6 esquematiza o funcionamento dos tópicos e serviços do ROS. Um tópico é um canal nomeado pelo qual os nós publicam e assinam mensagens. Os serviços são outra forma de comunicação no ROS. Eles são usados para chamadas de procedimento remoto, em que um nó pode solicitar que outro nó execute uma tarefa. Os pacotes são a unidade

básica de organização no ROS. Um pacote contém todo o código e dados necessários para uma funcionalidade específica, como um nó ou um serviço.

Figura 6 – Tópicos e serviços do ROS.



Fonte: Hasan (2019).

O ROS oferece uma ampla gama de funcionalidades que facilitam o desenvolvimento de robôs, incluindo (OPEN\_ROBOTICS, 2021):

- **Descoberta de nós:** Permite que os nós se encontrem automaticamente na rede;
- **Sistema de mensagens:** Facilita a comunicação entre os nós através de mensagens publicadas em tópicos;
- **Sistema de parâmetros:** Permite armazenar e recuperar parâmetros de configuração para os nós;
- **Sistema de tempo:** Sincroniza o tempo entre os nós;
- **Ferramentas de diagnóstico:** Fornecem ferramentas para monitorar e depurar o sistema construído;
- **Bibliotecas de software:** Oferecem bibliotecas para tarefas comuns de robótica, como navegação, manipulação e controle.

O ROS é utilizado em aplicações que incluem robôs de variados tipos, como autônomo, manipulador, colaborativo e de serviço. Desse modo, o *framework* se consolidou como uma ferramenta essencial para facilitar o desenvolvimento de sistemas que envolvem comunicação com elementos robotizados. Sua arquitetura modular, flexibilidade e ampla gama de funcionalidades tornam o ROS o *framework* ideal para a criação de sistemas robustos, escaláveis e autônomos. Ele possui uma comunidade ativa e documentação abrangente, o que facilita o aprendizado e a utilização em diversos tipos de aplicação.

## 3 Desenvolvimento

Este capítulo apresenta como foi o desenvolvimento do supervisor do sistema de simulação que é tema deste trabalho. Serão apresentadas as ferramentas de *hardware* e *software* utilizadas, a arquitetura, o projeto e os requisitos. Assim, este capítulo apresenta de forma detalhada a estrutura do supervisor para o entendimento de suas funcionalidades.

O acompanhamento do desenvolvimento do sistema de simulação foi feito em reuniões periódicas entre as equipes de desenvolvimento de cada frente do projeto. Outras reuniões eram realizadas entre os representantes da Vale e da Unifei. Nessas reuniões eram apresentados os resultados alcançados em cada ciclo de desenvolvimento e eram discutidos os pontos relevantes para ajustar o projeto às necessidades da empresa. As principais fontes de informação sobre o que o sistema deve oferecer foram os documentos disponibilizados pela mineradora, as visitas técnicas em ambas organizações e as reuniões citadas.

### 3.1 Ferramentas Utilizadas

Sobre as ferramentas de *hardware*, é importante lembrar que o sistema supervisor é executado em um computador dedicado, ou seja, em uma máquina separada da que roda o *software* da simulação. O conjunto de *hardware* do supervisor possui as seguintes especificações:

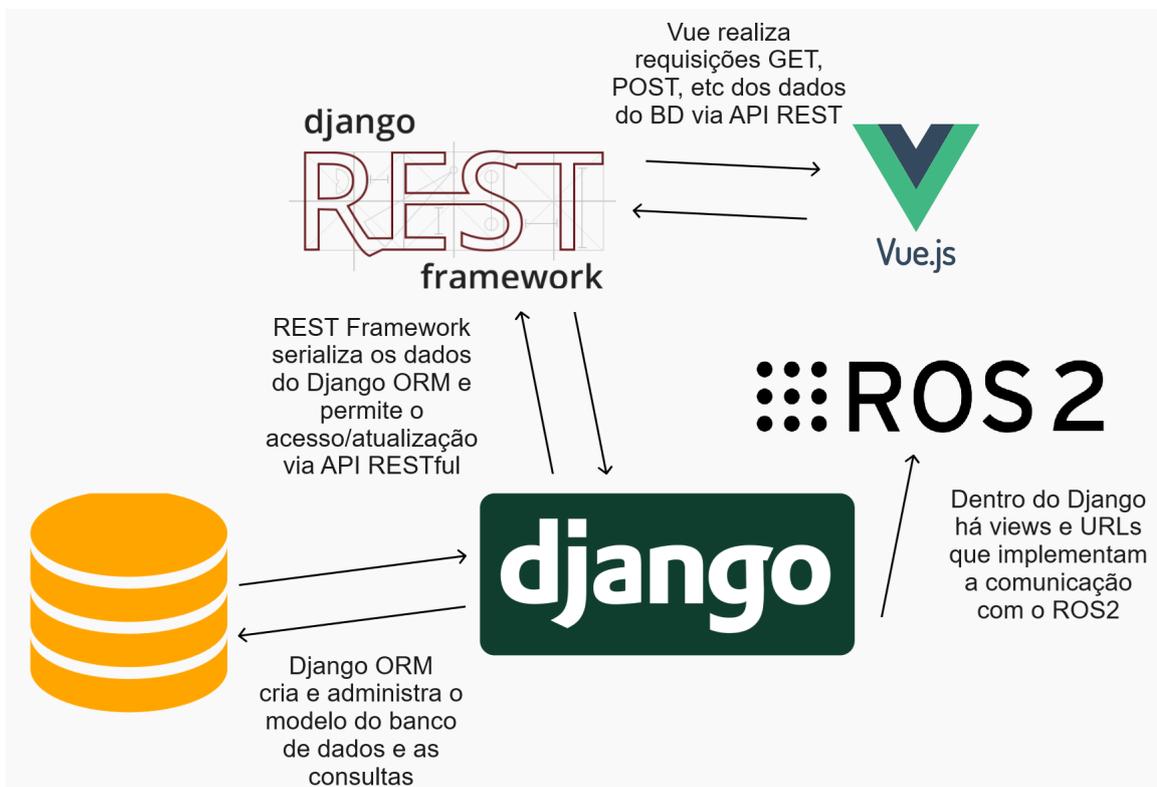
- Processador Intel Core i7-12700F LGA1700 - 2.10GHZ 25MB de Cache
- Placa de Vídeo MSI Nvidia Geforce RTX 3060 Ti Ventus / 8GB / GDDR6
- Placa Mãe MSI Pro B660M-A Wi-Fi DDR4 Intel Soquete LGA 1700
- Armazenamento SSD M.2 2TB Nvme Kingston SNV2S/2000G 3500MB/s
- Memória 64GB (4x16GB) Kingston Fury Beast DDR4 16GB 3000 (KF430C15BB1/16)
- Fonte Corsair RM850 ATX 850W 80PLUS Gold Full Modular CP-9020235
- Gabinete Gamer Aigo DarkFlash DK351, RGB, Mid-Tower, Lateral de Vidro, Com 4 Fans, Preto, DK351-BK
- Monitor Gamer Samsung 27" FHD, 75Hz, HDMI, VGA, Freesync
- Mouse e teclado Kit Logitech MK120 com fio
- Sistema Operacional Ubuntu Linux 20.04 Focal Fossa

Este computador também pode ser usado como estação de trabalho do instrutor, onde este poderá realizar as tarefas relativas à configuração da simulação e avaliação do treinando (operador). As tarefas incluem criação e gerenciamento de cenários, gerenciamento de usuários e currículos de avaliação, acompanhamento de simulação ativa, alteração de configurações de cenário, entre outras.

Sobre as ferramentas de *software*, de início, foi utilizado apenas o *framework* Django e algumas ferramentas secundárias para implementar o sistema numa abordagem de pilha completa (*fullstack*). Dessa forma, todas as camadas do padrão MVC ficariam implementadas pelo *framework* Django. Porém, essa abordagem se mostrou insuficiente para atender à complexidade da interface do usuário necessária para alcançar os objetivos do sistema supervisorio de forma satisfatória.

Assim, a equipe de desenvolvimento decidiu por utilizar o Django Rest Framework em conjunto com o Django para criação de uma API. Além disso, a equipe decidiu por deixar a interface do usuário independente do restante do supervisorio e à cargo de outro *framework* específico para isso, o Vue. Nesta abordagem, é comum dizer que o Vue consome a API para ter acesso aos dados do sistema. A Figura 7 mostra a configuração final dos *frameworks* usados no desenvolvimento e a comunicação entre eles. Nela aparece o ROS2 que é usado para a comunicação do sistema supervisorio com o sistema de simulação.

Figura 7 – *Frameworks* usados no desenvolvimento.



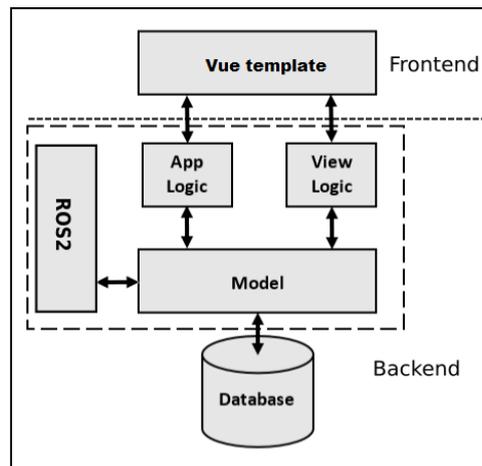
Fonte: elaborado pelo autor.

## 3.2 Arquitetura do Sistema Supervisório

A arquitetura construída segue o padrão MVC (*Model-View-Controller* ou Modelo-Visão-Controlador), já que foi utilizado o Django que é um *framework* baseado no padrão MVC. Além disso, o sistema possui separação entre as camadas de apresentação de dados e interação com o usuário (*front-end*) e acesso ao banco de dados e ao módulo de integração com o sistema de simulação (*back-end*).

A Figura 8 mostra a configuração do supervisório. A interação do *front-end* com o *back-end* se dá através de requisições HTTP (*Hyper Text Transference Protocol*). Uma requisição HTTP, ou *HTTP request*, é uma mensagem enviada por um cliente a um servidor solicitando algum recurso ou ação. Essa mensagem contém informações como o método HTTP, a URL (*Uniform Resource Locator* ou Localizador Uniforme de Recursos), a versão do protocolo HTTP, cabeçalhos com metadados da requisição e o corpo com dados a serem enviados. O *framework* ROS2 também aparece na figura, indicando que é no *back-end* que fica implementada a comunicação com o sistema de simulação.

Figura 8 – Arquitetura do sistema supervisório.

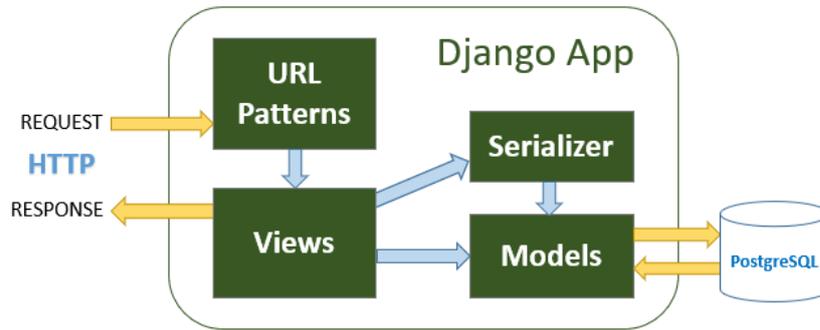


Fonte: Vitor e Diniz (2023).

A Figura 9 foca em como o Django processa uma requisição HTTP. Ao receber uma requisição, ela é filtrada pelo componente *URL Patterns*, que associa os *endpoints* (URLs específicas) com a visão (*view*) correspondente. A visão solicitada interage com os modelos de dados, que requisitam serviços ao banco de dados. Este processa a chamada e devolve o resultado para a visão correspondente, que envia os dados processados para o componente do *front-end* responsável pela apresentação desses dados. Na figura dá pra ver que o SGBD (Sistema Gerenciador de Banco de Dados) usado no projeto foi o PostgreSQL, porém, o tratamento das requisições é independente do SGBD.

A Figura 10 amplia os elementos do sistema e aprofunda o esquema de comunicação do supervisório com o sistema de simulação via *framework* ROS2. Os sistemas se comunicam

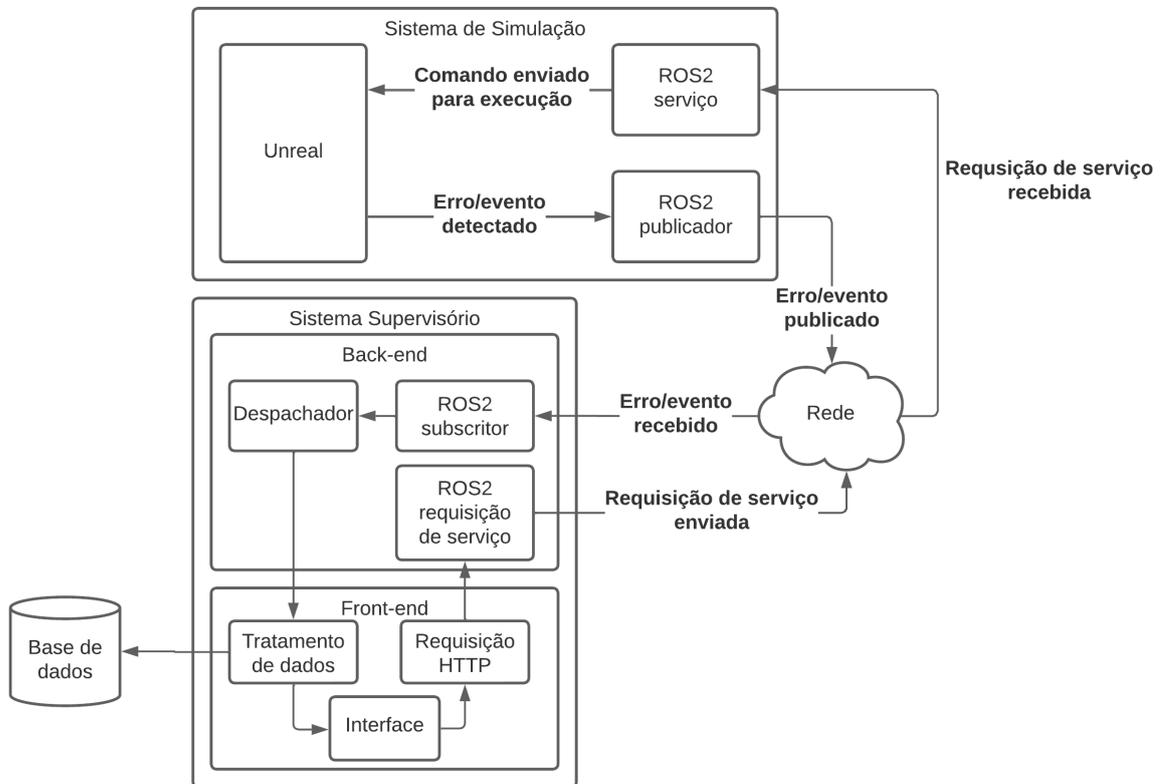
Figura 9 – Arquitetura recomendada para sistemas Django + PostgreSQL



Fonte: <<https://www.bezkoder.com/django-postgresql-crud-rest-framework/>>

através de requisições de serviço e respostas via rede. A comunicação acontece pela troca de mensagens, através de um nó do ROS2 configurado para dar suporte à essa troca de mensagens, descoberta e assinatura de tópicos, envio de requisições de serviços e controle de publicadores e subscritores. Na figura, “Unreal” é o motor (*engine*) gráfico responsável por gerar os elementos gráficos no sistema de simulação.

Figura 10 – Arquitetura do módulo de integração com o sistema de simulação.

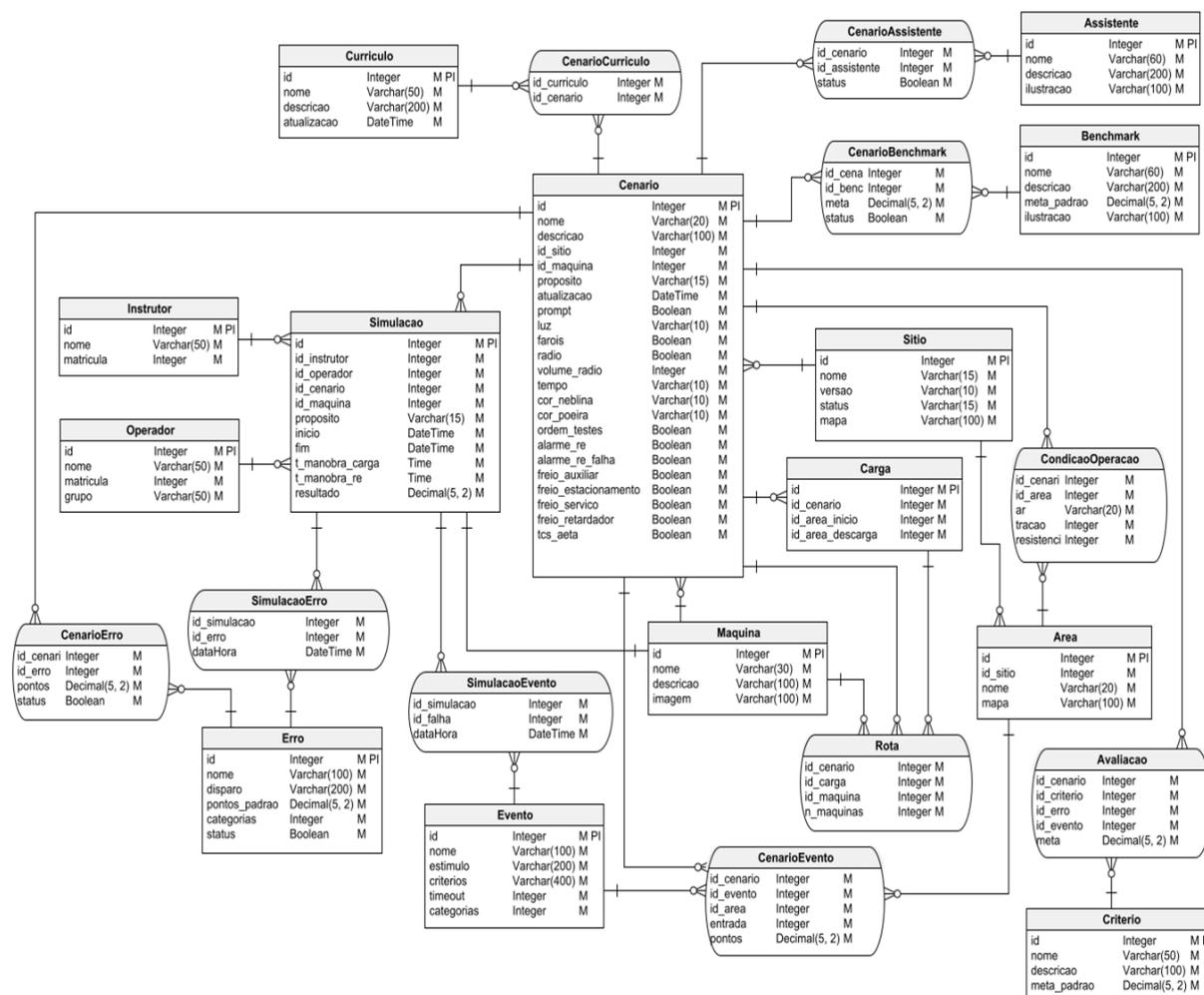


Fonte: Vitor e Diniz (2023).

### 3.3 Projeto do Banco de Dados

Após o entendimento do sistema de simulação de maneira ampla, o desenvolvimento do supervisor seguiu com o objetivo de relacionar todos os dados que são tratados pelo mesmo. Um modelo de banco de dados foi elaborado para esquematizar os conjuntos de dados e o relacionamento entre eles. O projeto conceitual do Banco de Dados (BD) está apresentado na Figura 11 por meio de um Diagrama Entidade-Relacionamento (DER).

Figura 11 – DER - Projeto Conceitual do BD.



Fonte: elaborado pelo autor.

No diagrama, a tabela “Cenario” reúne os dados para a configuração de um cenário de simulação. Esta tabela é de importância central e se relaciona com todas as demais tabelas, direta ou indiretamente. A segunda tabela mais importante é a “Simulacao”. Esta tabela e seus relacionamentos reúnem os dados de uma sessão de simulação. Já a tabela “Currículo” agrupa cenários.

No modelo há tabelas que reúne os dados dos instrutores, operadores, grupos de operadores, erros do operador e eventos de simulação. Também há tabelas que reúnem os

dados para a avaliação do desempenho de um operador durante uma sessão de simulação. O significado de cada uma das tabelas é bastante intuitivo, porém, explicações mais detalhadas dos conceitos relacionados aos dados que elas reúnem serão dadas ao longo deste trabalho.

O *framework* Django fornece ferramentas para criar o banco de dados, criar as tabelas e realizar o povoamento (alimentação) das mesmas. Isso é feito através de classes ORM (*Object Relational Mapping*). O ORM ou Mapeador de Relação de Objetos é uma técnica utilizada no desenvolvimento de *software* orientado a objetos que permite mapear objetos para tabelas em um banco de dados relacional. Com isso, é possível realizar as consultas para criar, retomar, atualizar e excluir registros, ou seja, as funcionalidades do CRUD (Create, Read, Update, Delete), bem como fazer alterações no modelo de dados usando migrações (*migrations*).

A camada de dados passou por vários ciclos de readequações durante todo o desenvolvimento. Isto é comum em trabalhos deste tipo, já que com o andamento do projeto, a equipe vai aprofundando o entendimento dos requisitos de dados. Felizmente, o Django possui ferramentas que facilitam este trabalho, sem que seja necessário escrever código em SQL (*Structure Query Language*) para realizar cada alteração.

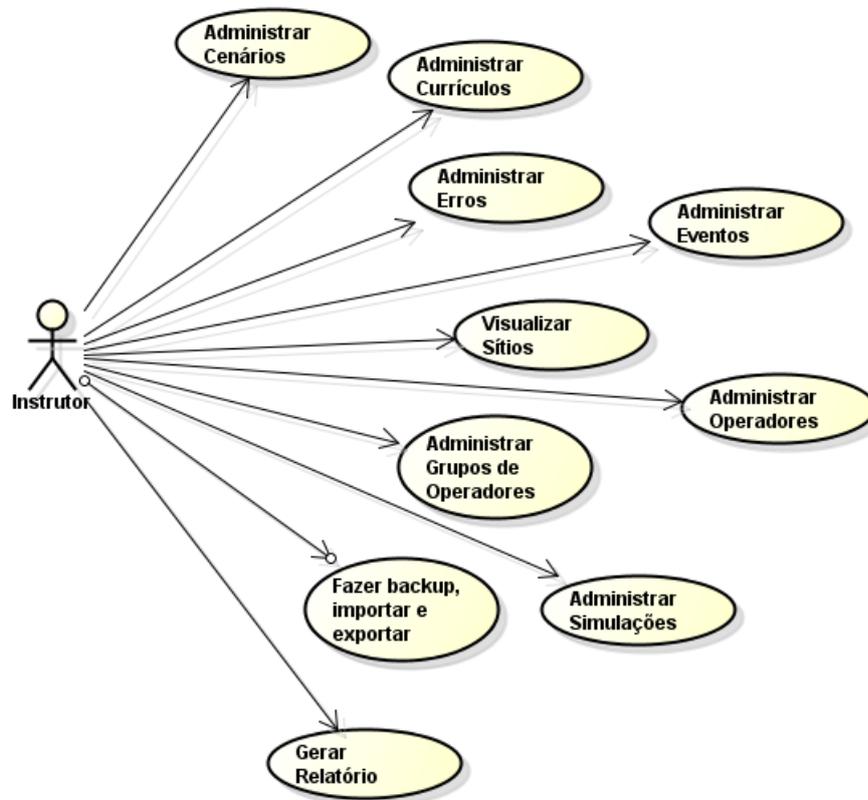
### 3.4 Descrição das Funcionalidades

Os usuários do sistema supervisor são os instrutores de treinamento de operação de máquinas de mineração. Neste caso específico, a máquina se trata do caminhão fora-de-estrada de mineração CAT-793F. Os operadores que serão treinados não utilizam diretamente o supervisor, então não são considerados usuários do mesmo. Eles interagem diretamente com o *software* de simulação e indiretamente com o supervisor, tendo o instrutor como intermediário desta interação.

Os instrutores de treinamento de operação de máquinas de mineração são profissionais da mineradora responsáveis pelo recrutamento, seleção, capacitação, avaliação e reciclagem dos recursos humanos envolvidos na frota especializada da mineração. Na estação do instrutor, ele pode criar novos cenários, trocar cenários, criar novos usuários ou grupos, acompanhar uma simulação com controles que alteram o cenário em tempo de execução, entre outras funcionalidades. O monitor pode girar sobre sua montagem para ser visto de maneira vertical para mostrar a vista correta durante a simulação.

Um caso de uso tem como objetivo relatar como um sistema interage com os usuários ou demais sistemas externos. Ele oferece uma visão geral sobre as funcionalidades do sistema, ressaltando os principais cenários de uso. A Figura 12 apresenta os casos de uso levantados para o supervisor, resumindo graficamente as funcionalidades que o sistema deve oferecer ao usuário, o instrutor.

Figura 12 – Casos de uso.



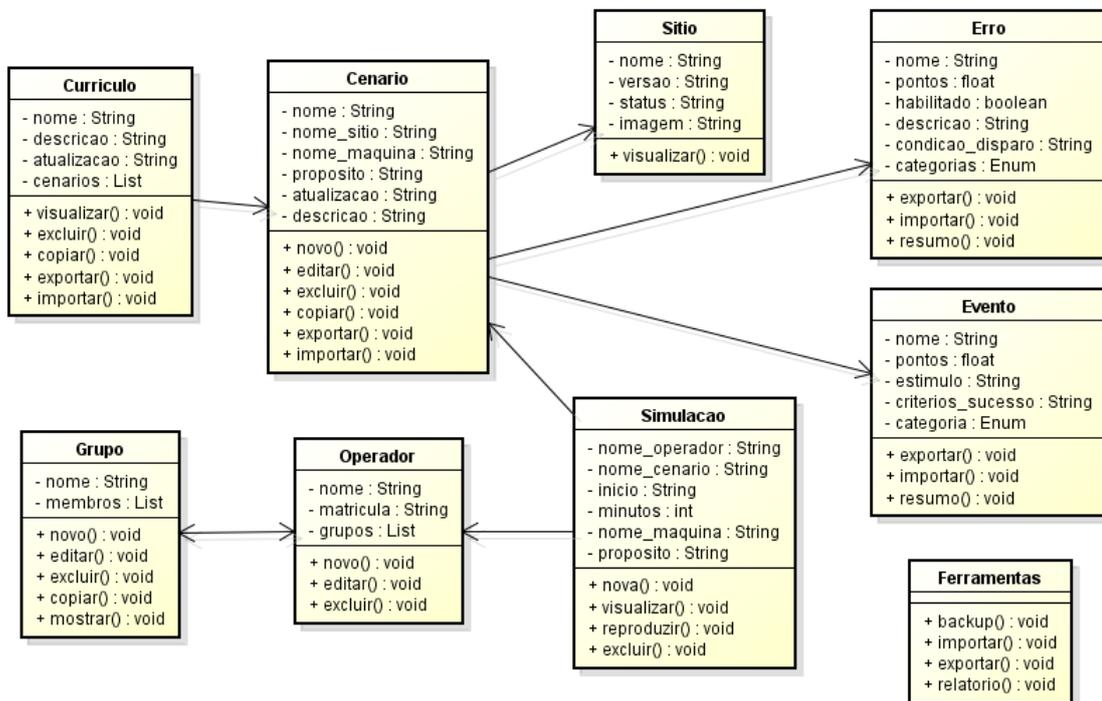
Fonte: elaborado pelo autor.

Neste momento cabe uma breve explicação sobre os currículos e cenários. Um currículo é composto de vários cenários, que, por sua vez, são relacionados a um sítio específico e configuração de erros e eventos. Para o entendimento do que são cenários, segue o exemplo de alguns:

- **Ambientação:** cenário usado para que o operador se ambiente com os comandos da máquina por cerca de dez minutos, sem ser avaliado;
- **Diagnóstico Inicial:** tem o objetivo de produzir um retrato do perfil inicial de trabalho do operador realizado sem treinamento no simulador, somente com os conhecimentos prévios do mesmo;
- **Prática Mina:** cenário para treinamento, utilizando dados da entrevista com o operador e a observação no cenário anterior a fim de aprimorar os conhecimentos/habilidades dele;
- **Prática Britador:** cenário para treinamento em um local específico da mina;
- **Diagnóstico Final:** cenário para avaliar o aprendizado com o treinamento.

A Figura 13 apresenta o diagrama de classes resumido do sistema supervisor, esquematizando as classes do mesmo e como elas se relacionam. A classe “Simulacao” tem importância central, pois ela trata os dados de uma sessão de treinamento. Uma simulação é composta pelos dados de instrutor e um operador e guarda diversos dados do treinamento, como o cenário, os erros cometidos pelo operador, o tratamento que o mesmo deu aos eventos ocorridos, a duração, a gravação, o propósito e a avaliação. No diagrama também estão presentes as ferramentas indispensáveis ao sistema referente aos dados: *backup*, importar, exportar e relatório.

Figura 13 – Diagrama de Classes.



Fonte: elaborado pelo autor.

A partir deste diagrama e dos conceitos aqui apresentados é possível entender o funcionamento básico do supervisor, porém, sua estrutura completa não faz parte do escopo deste trabalho. Na próxima seção serão apresentados os requisitos. Eles ajudam no entendimento das funcionalidades do sistema de uma forma estruturada.

Uma parte extensa e que envolve vários pontos do sistema supervisor é a avaliação do desempenho do operador. A avaliação se inicia na criação de um cenário, em que há várias configurações relacionadas à mesma. Durante uma simulação, são coletados os dados necessários para a avaliação e ao final da simulação, uma tela com estes dados é exibida, bem como o histórico de avaliações do operador que acabou de realizar a sessão de simulação. O instrutor se relaciona constantemente com as funcionalidades do supervisor relacionadas à avaliação, o que faz com que o processo de avaliação conte com aspectos subjetivos do instrutor.

### 3.5 Requisitos

Os requisitos foram levantados a partir de um estudo detalhado das informações levantadas nas interações com a empresa proponente, em documentos, visitas técnicas e reuniões. O resultado foi a elaboração do documento de visão e do documento de especificação dos requisitos, com a análise de diversos fatores que influenciam o funcionamento do sistema de modo a atender às necessidades da empresa.

A Tabela 1 resume os requisitos funcionais, não-funcionais e regras de negócio identificados para o sistema supervisor. Nas subseções a seguir, esse assunto é explorado.

Tabela 1 – Resumo dos requisitos e regras de negócio.

<b>Requisitos funcionais</b>	
RF-1	Administrar cenários
RF-2	Administrar currículos
RF-3	Administrar erros
RF-4	Administrar eventos
RF-5	Visualizar sítios
RF-6	Administrar operadores
RF-7	Administrar grupos de operadores
RF-8	Administrar simulações
RF-9	Fazer backup, importar e exportar
RF-10	Gerar relatório
<b>Requisitos não funcionais</b>	
NF-1	Autenticação de usuário
NF-2	Facilidade do uso e similaridade com sistemas do tipo
NF-3	Telas limpas e responsivas
NF-4	Disponibilidade dos dados
NF-5	Segurança da informação
NF-6	Comunicação satisfatória com o sistema de simulação
<b>Regras de negócio</b>	
RN-1	Uma simulação tem cerca de meia hora e participam dela um instrutor e um operador
RN-2	Durante uma sessão, o instrutor visualiza a simulação dentro do sistema supervisor e pode interferir nos eventos
RN-3	Após a sessão, o instrutor dá <i>feedback</i> ao operador utilizando as informações do sistema

Fonte: elaborado pelo autor.

### 3.5.1 Requisitos funcionais

Requisitos funcionais são todos os problemas e necessidades que devem ser atendidos e resolvidos pelo *software* que se pretende criar por meio de funções e serviços. Tudo o que for relacionado a uma ação a ser feita é considerado uma função. Quanto menos ambíguos e mais objetivos forem os requisitos funcionais, maior será a qualidade do *software* gerado. É natural que estes requisitos sofram alterações à medida que o desenvolvimento do sistema avance. Os requisitos funcionais da tabela 1 são descritos a seguir.

#### RF-1 – Administrar cenários

O instrutor utiliza o sistema supervisorio para criar, editar, excluir, copiar, exportar e importar cenários. A atividade de criação de um cenário é a mais complexa e é composta de várias telas diferentes para definir as opções de configuração dos cenários. Além de exportar digitalmente um cenário, é importante fornecer a opção de gerar um arquivo com todas as informações do cenário criado.

#### RF-2 – Administrar currículos

O instrutor utiliza o sistema supervisorio para visualizar, excluir, copiar, exportar e importar um currículo. Os currículos são compostos por uma lista de cenários, como, por exemplo, para ambientação, diagnóstico inicial, prática na mina, prática no britador e diagnóstico final.

#### RF-3 – Administrar erros

O instrutor utiliza o sistema supervisorio para exportar, importar, ver resumo, descrição, condição de disparo e categorias dos erros que o operador pode cometer durante uma simulação. Na criação de cenário, os erros são personalizados para o propósito específico de cada cenário.

#### RF-4 – Administrar eventos

O instrutor utiliza o sistema supervisorio para exportar, importar, ver resumo, estímulo, critérios de sucesso e categorias dos eventos que podem ocorrer durante uma simulação.

#### RF-5 – Visualizar sítios

O instrutor utiliza o supervisorio para visualizar os sítios instalados no sistema. Os sítios podem ser genéricos ou simularem minas reais. A tela para visualizar um sítio apresenta foto-mapa, locais de carga e descarga e áreas com objetivos específicos.

#### RF-6 – Administrar operadores

O instrutor utiliza o sistema supervisorio para criar, editar e excluir um registro de um operador, ou seja, os usuários que serão treinados ou avaliados com as simulações.

#### RF-7 – Administrar grupos de operadores

O instrutor utiliza o sistema supervisorio para criar, editar, excluir, copiar e mostrar os grupos de operadores. Geralmente estes grupos são nomeados por letras (A, B, C, ...) e funcionam como turmas de um processo de treinamento e/ou avaliação.

#### RF-8 – Administrar simulações

O instrutor utiliza o supervisor para criar, visualizar, reproduzir e excluir uma simulação. A tarefa de criar uma simulação fica na tela inicial do sistema atualmente utilizado. Após o instrutor escolher as opções e dar início a uma simulação, o sistema apresenta uma tela na qual o instrutor interage em tempo de execução com a simulação. As demais tarefas são estáticas e se comunicam com o banco de dados. A tarefa visualizar apresenta os dados de uma simulação, já a tarefa reproduzir, apresenta a gravação de uma simulação.

#### RF-9 – Fazer *backup*, importar e exportar

O instrutor utiliza o sistema supervisor para fazer *backup*, importar e exportar os dados do sistema como um todo. Estas tarefas são críticas para o sistema.

#### RF-10 – Gerar relatório

O instrutor utiliza o sistema supervisor para gerar um relatório com os dados das simulações realizadas. Devem ser fornecidos para serem aplicados aos dados a serem incluídos no relatório, bem como apresentar os dados de forma gráfica para fácil visualização.

### 3.5.2 Requisitos não-funcionais

Os requisitos não funcionais são todos aqueles relacionados à forma como o *software* tornará realidade o que está sendo planejado, mas que não são vistos a princípio como funcionalidades específicas. Costumam ser mais genéricos que os requisitos funcionais, abrangendo vários deles ou todo o sistema. Segue uma lista de requisitos não funcionais facilmente identificados ao se considerar as características do supervisor:

- NF-1 – Autenticação de usuário (instrutor com perfil de administrador)
- NF-2 – Facilidade de uso e similaridade com sistemas do tipo
- NF-3 – Telas limpas e responsivas
- NF-4 – Disponibilidade dos dados
- NF-5 – Segurança da informação
- NF-6 – Comunicação satisfatória com o sistema de simulação

Novos requisitos desse tipo podem ser identificados ao avançar no desenvolvimento do sistema e isto é natural. Os requisitos não funcionais podem ser desdobrados em requisitos funcionais ao se desenvolver o sistema. Ao optar por implementação em sistema *web* publicado na *Internet*, grandes preocupações com a segurança são necessárias.

### 3.5.3 Regras de negócio

Regras de negócio são declarações sobre a forma da empresa fazer negócio. Elas refletem políticas do negócio. As organizações possuem políticas para satisfazer os objetivos do negócio, satisfazer clientes, fazer bom uso dos recursos, e obedecer às leis ou convenções gerais do negócio. A partir da análise dos documentos fornecidos pela empresa, reuniões e visitas técnicas, as seguintes regras de negócio foram identificadas:

- Uma simulação tem duração de cerca de meia hora e participam dela um instrutor e um operador. O operador interage com os controles similares aos do caminhão e o instrutor com o computador do supervisor;
- O instrutor deve criar um ambiente positivo; informar-se sobre o aluno, sobre suas experiências, habilidades e treinamento prévio em simuladores, explicar a finalidade do treinamento e os resultados a serem atingidos, explicar como o treinamento será realizado e a duração de cada cenário;
- Durante uma sessão executada, o instrutor pode ver e interagir com a simulação utilizando as ferramentas disponíveis na tela do monitor;
- Após a sessão de treinamento, o instrutor dá um *feedback* ao aluno, mostrando o resultado da sessão para o mesmo e quais foram os erros cometidos, dando exemplos de como estes erros impactam na segurança e produtividade do equipamento. O instrutor também deve explicar os resultados, o que o aluno fez certo e o que poderia fazer melhor, revisar os objetivos do treinamento, imprimir uma cópia do resumo da sessão e entregar ao aluno, informar sobre o que acontecerá a seguir, comparar com outras sessões, etc;
- O instrutor, como administrador do sistema, precisa acessar funções para criar cenários, administrar erros, eventos, sítios, operadores, grupos de operadores, simulações, fazer *backup* do sistema, importar e exportar todos os dados relacionados ao mesmo e gerar diferentes tipos de relatórios.

### 3.5.4 Erros do operador

Erros são ações do operador que não correspondem ao modo correto de conduzir o veículo fora-de-estrada CAT793F. Segue a descrição dos requisitos relacionados aos erros resumidos na Tabela 2.

RF-11 – Listar informações de erros

O instrutor utiliza o sistema supervisor para visualizar a lista das informações de cada erro que pode ser cometido pelo operador. Essas informações são: código (id), nome, condições de disparo, pontuação, categorias e status.

Tabela 2 – Resumo dos requisitos relacionados aos erros.

<b>Erros</b>	
Supervisório	
RF-11	Listar informações de erros
RF-12	Habilitar ou desabilitar a verificação de um erro
RF-13	Atribuir pontuação a um erro
RF-14	Configurar erros em cenários
RF-15	Apresentar erros cometidos
RF-16	Guardar histórico de erros
Simulador	
RF-17	Detectar ocorrência de erro
RF-18	Apresentar erro na tela
RF-19	Reportar erro ao supervisório

Fonte: elaborado pelo autor.

RF-12 – Habilitar ou desabilitar a verificação de um erro

Na lista do requisito anterior (RF-1), o sistema supervisório permite habilitar ou desabilitar a verificação da ocorrência de determinado erro durante uma simulação genérica.

RF-13 – Atribuir pontuação a um erro

Na lista do (RF-1), o sistema supervisório apresenta uma pontuação padrão ou já alterada e permite ao usuário alterar este valor.

RF-14 – Configurar erros em cenários

Durante a criação de cenário, o sistema supervisório permite ao usuário configurar a verificação de erros do operador, alterando o valor da sua pontuação e habilitando ou desabilitando a verificação de determinado erro. Esta configuração é salva como forma de customizar a forma que determinado cenário trata os erros.

RF-15 – Apresentar erros cometidos

Após uma simulação, o sistema supervisório apresenta ao usuário a lista de erros cometidos pelo operador e a pontuação final do mesmo, juntamente com outras informações que auxilia o instrutor a avaliar o desempenho do operador.

RF-16 – Guardar histórico de erros

O sistema supervisório guarda em banco de dados o histórico de erros do operador em cada uma das simulações realizadas pelo mesmo.

RF-17 – Detectar ocorrência de erro

O sistema simulador analisa os dados gerados durante a simulação procurando detectar a ocorrência de um erro pelo operador.

RF-18 – Apresentar erro na tela

Após detectar um erro, o simulador apresenta em um prompt na tela do operador as

informações do erro cometido pelo mesmo.

#### RF-19 – Reportar erro ao supervisor

Após detectar um erro e apresentá-lo na tela do operador, o simulador reporta ao supervisor as informações do erro cometido, pois é tarefa do supervisor guardar estas informações, persistindo-as em banco de dados, para uso posterior pelo instrutor.

### 3.5.5 Eventos de simulação

Os eventos são situações que ocorrem durante uma simulação, para os quais espera-se um determinado tratamento, ou seja, um conjunto de ações por parte do operador com vistas a passar pelo evento da maneira adequada. Segue a descrição dos requisitos relacionados aos eventos de simulação resumidos na Tabela 3.

Tabela 3 – Resumo dos requisitos relacionados aos eventos.

Eventos	
Supervisor	
RF-20	Listar informações de eventos
RF-21	Atribuir meta ( <i>timeout</i> ) a um evento
RF-22	Configurar eventos em cenários
RF-23	Disparar evento com uma simulação em andamento
RF-24	Apresentar tratamento de eventos
RF-25	Guardar histórico de eventos
Simulador	
RF-26	Disparar ocorrência de evento
RF-27	Verificar tratamento de evento
RF-28	Apresentar tratamento de evento na tela da simulação
RF-29	Reportar evento e tratamento ao supervisor

Fonte: elaborado pelo autor.

#### RF-20 – Listar informações de eventos

O instrutor utiliza o sistema supervisor para visualizar a lista das informações de cada evento que pode ocorrer durante uma simulação. Essas informações são: código (id), nome, estímulo, critérios de sucesso, *timeout* e categorias (eficiência de combustível, manutenção, produtividade, segurança).

#### RF-21 – Atribuir meta (*timeout*) a um evento

Na lista do requisito anterior (RF-10), o sistema supervisor apresenta um *timeout* padrão ou já alterado e permite ao usuário alterar este valor.

#### RF-22 – Configurar eventos em cenários

Durante a criação de cenário, o sistema supervisor permite ao usuário configurar o

disparo de eventos durante uma simulação, alterando o valor do timeout e definindo a área em que o evento irá ocorrer.

#### RF-23 – Disparar evento durante uma simulação

O sistema supervisorio deve permitir ao instrutor disparar um evento com a simulação em andamento, ou seja, interagir com a simulação em tempo de execução.

#### RF-24 – Apresentar tratamento de eventos

Após uma simulação, o sistema supervisorio apresenta ao usuário a lista de eventos ocorridos e se o tratamento que o operador realizou satisfaz ou não aos critérios de sucesso e a pontuação final do mesmo, juntamente com outras informações que auxilia o instrutor a avaliar o desempenho do operador.

#### RF-25 – Guardar histórico de eventos

O sistema supervisorio guarda em banco de dados o histórico dos eventos ocorridos em cada uma das simulações realizadas por determinado operador.

#### RF-26 – Disparar ocorrência de evento

O sistema simulador carrega os dados configurados para o cenário a ser executado e dispara a ocorrência de um evento quando a área configurada para ocorrer o mesmo é atingida pelo caminhão.

#### RF-27 – Verificar tratamento de evento

Ao ocorrer um evento, o sistema simulador verifica se os critérios de sucesso foram atingidos pelo operador dentro do timeout configurado.

#### RF-28 – Apresentar tratamento de evento na tela

Após o tratamento de um evento pelo operador, com sucesso ou falha, o sistema simulador apresenta o resultado em um prompt na tela do operador.

#### RF-29 – Reportar evento e tratamento ao supervisorio

O sistema simulador reporta as informações dos eventos disparados durante a simulação ao supervisorio, para este guardá-las em BD.

### 3.5.6 Integração supervisorio x simulador

O sistema supervisorio deve exibir uma interface em que o instrutor pode acompanhar uma sessão de simulação em tempo de execução. Durante o acompanhamento da sessão, uma série de informações como a telemetria do veículo (velocidade, rotações por minuto (RPM), acionamento de freios) e os erros e eventos detectados pelo sistema de simulação, assim como uma visualização da tela do simulador são exibidos para acompanhamento do instrutor no sistema supervisorio.

A Tabela 4 resume os requisitos funcionais relacionados à integração do supervisorio com o simulador, e depois uma descrição de cada um deles é apresentada.

Tabela 4 – Resumo dos requisitos da integração supervisorio x simulador.

Supervisorio	
RF-30	Publicar dados para iniciar uma simulação
RF-31	Disparar a ocorrência de um evento
RF-32	Receber os dados de uma simulação
RF-33	Persistir os dados recebidos em BD
Simulador	
RF-34	Receber os dados para iniciar uma simulação
RF-35	Gerar eventos durante uma simulação
RF-36	Publicar os dados de uma sessão de simulação

Fonte: elaborado pelo autor.

#### RF-30 – Publicar dados para iniciar uma nova simulação

O instrutor utiliza o sistema supervisorio para iniciar uma nova simulação, preenchendo um formulário simples na página inicial. O supervisorio publica os dados para iniciar uma nova simulação no tópico ou serviço para este fim.

#### RF-31 – Disparar a ocorrência de um evento

Com uma simulação em andamento, o instrutor utiliza a tela de acompanhamento da simulação para disparar a ocorrência de um evento. Os dados referentes a este evento são publicados pelo supervisorio no tópico ou serviço para este fim.

#### RF-32 – Receber os dados de uma simulação

O supervisorio, durante uma simulação, recebe os dados referentes a sessão em andamento, os organiza e os prepara para serem persistidos.

#### RF-33 – Persistir os dados recebidos em Banco de Dados

O supervisorio guarda as informações dos dados de uma simulação no BD.

#### RF-34 – Receber os dados para iniciar uma nova simulação

O sistema simulador recebe todos os dados de configuração necessários para iniciar uma nova simulação, configura o ambiente de simulação e inicia a mesma.

#### RF-35 – Gerar eventos durante uma simulação

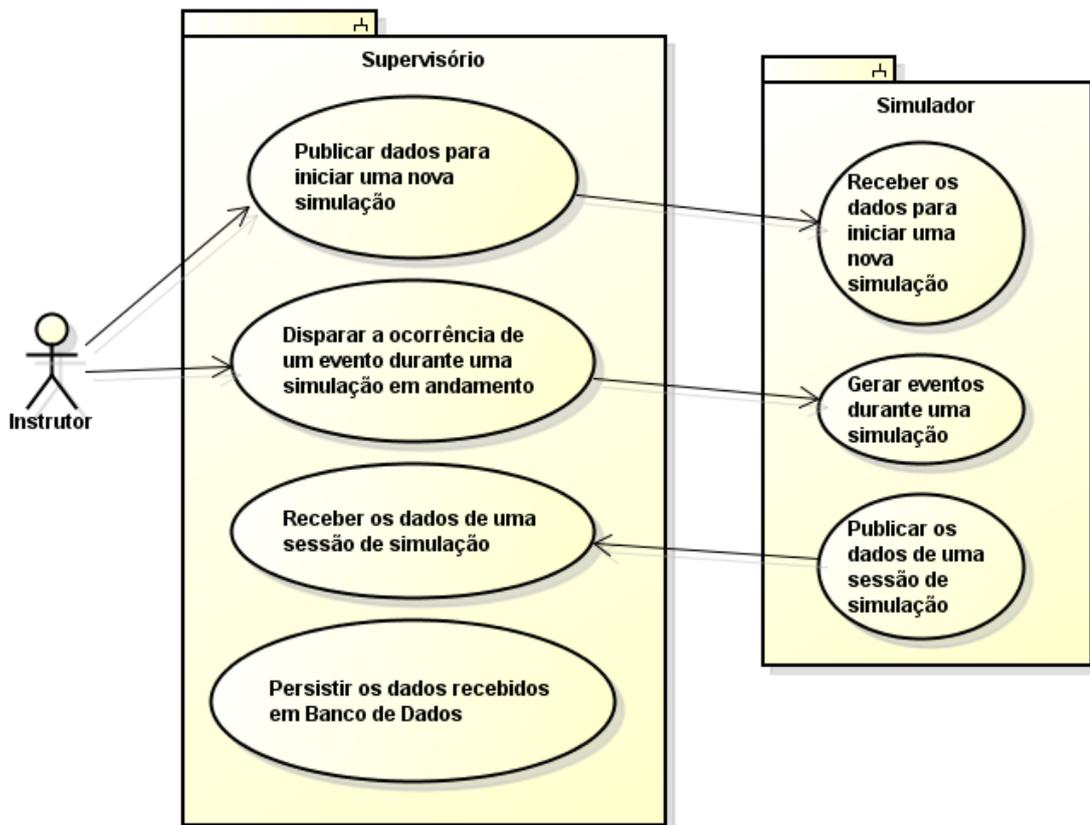
O simulador recebe o gatilho para disparar a ocorrência de um evento durante uma simulação e gera o evento com as configurações correspondentes.

#### RF-36 – Publicar os dados de uma sessão de simulação

Durante uma simulação, o simulador publica no tópico ou serviço destinado para isso os dados relacionados a sessão de simulação em andamento.

Para esquematizar os requisitos apresentados, a Figura 14 traz os casos de uso relacionados às funcionalidades descritas.

Figura 14 – Casos de uso da integração supervisorio x simulador.



Fonte: elaborado pelo autor.

### 3.5.7 Geração das tarefas de simulação

Esta atividade está relacionada com a função do sistema de simulação de utilizar as informações referentes às configurações de uma sessão de simulação para gerar as variadas tarefas que o operador deverá executar em seu treinamento no ambiente de simulação.

Sobre a geração das tarefas de simulação, foram considerados os seguintes pontos na análise dos requisitos:

- O supervisorio deve oferecer uma interface de visualização do mapa da mina onde o instrutor deverá inserir os pontos de início e fim da trajetória;
- Os sistemas em conjunto deverão gerar a trajetória considerando os diferentes caminhos possíveis, de preferência aquele com a rota otimizada;
- O sistema simulador deve associar todos os gatilhos de falhas que estará contido em determinada rota;
- O sistema simulador deve associar a configuração de clima e condições de visibilidade para o ambiente de simulação.

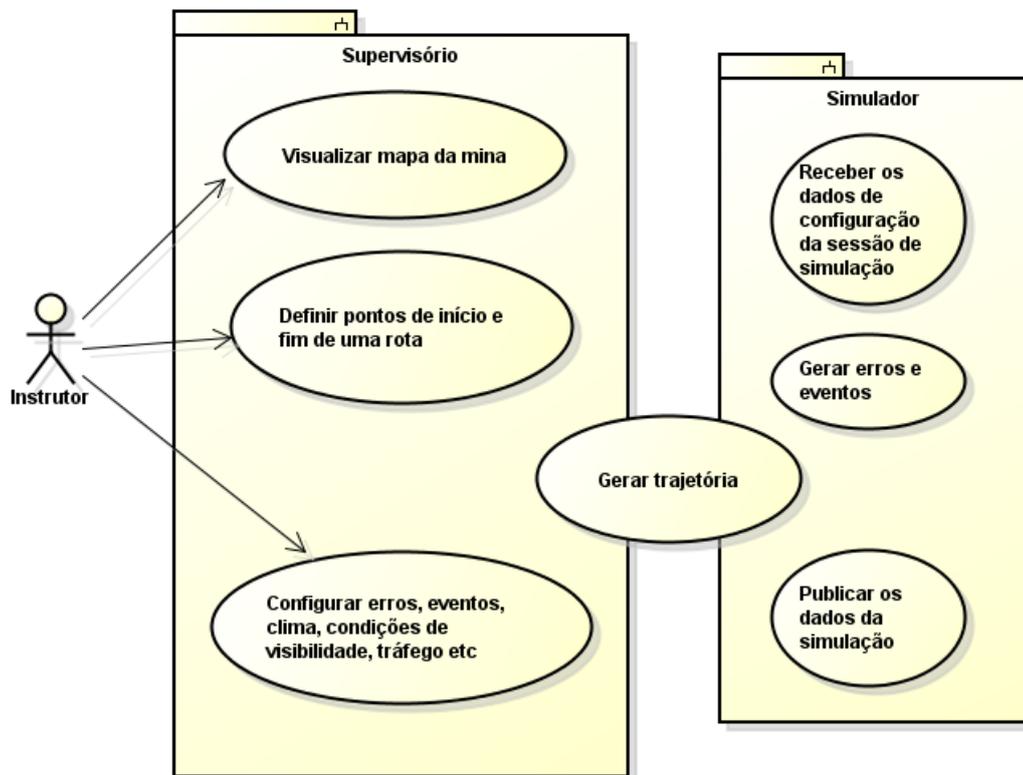
A Tabela 5 resume os requisitos funcionais descritos e a Figura 15 esquematiza os casos de uso referentes aos requisitos da geração das tarefas de simulação.

Tabela 5 – Resumo dos requisitos da geração das tarefas de simulação.

Supervisório	
RF-37	Visualizar mapa da mina
RF-38	Definir pontos de início e fim de uma rota
RF-39	Configurar dados e condições da simulação
RF-40	Gerar trajetória
Simulador	
RF-41	Receber os dados de configuração da simulação
RF-42	Gerar erros e eventos
RF-43	Publicar os dados da simulação

Fonte: elaborado pelo autor.

Figura 15 – Casos de uso da geração das tarefas de simulação.



Fonte: elaborado pelo autor.

## 4 Resultados

Este capítulo apresenta os resultados deste trabalho, buscando demonstrar que o sistema supervisorio desenvolvido cumpriu com os objetivos gerais e específicos estipulados, bem como implementou quase todos os requisitos funcionais, não-funcionais e regras de negócio apresentados no capítulo anterior.

O resultado foi possível devido à dedicação dos envolvidos em aplicar melhores práticas em programação, uso de *software* livre, ferramentas e tecnologias atuais. Durante as reuniões periódicas e as visitas técnicas na Unifei, os resultados foram avaliados pelos profissionais da mineradora de forma positiva, ressaltando que o sistema desenvolvido apresenta uma interface limpa, atual e responsiva.

O esquema de autenticação do usuário e a interface de administrador do sistema supervisorio também foram desenvolvidos, porém, não serão apresentadas neste trabalho. Apesar de fazer parte dos resultados do desenvolvimento, estas partes envolvem questões de segurança que devem ficar resguardadas aos envolvidos no projeto.

### 4.1 Resultados Qualitativos

A qualidade de um sistema de informação está relacionada com alcançar os objetivos gerais e específicos estipulados e implementar as funcionalidades de forma a satisfazer os requisitos do sistema. A partir deste conceito, são apresentados os resultados qualitativos a seguir. Eles foram organizados fazendo uma relação com os requisitos funcionais. Os requisitos não funcionais e as regras de negócio permeiam os requisitos funcionais, sendo sempre considerados durante todo o desenvolvimento.

#### 4.1.1 RF-1 – Administrar cenários

Um cenário é uma lista de condições usadas para configurar as ações/erros/eventos que serão observados durante uma sessão de simulação. A atividade de gerenciamento de cenários é o requisito mais complexo por envolver um grande número de informações e dependências de outros requisitos. Estas informações serão usadas pelo sistema de simulação para apresentar os elementos gráficos e o comportamento da simulação da forma que foi configurada pelo usuário (instrutor).

A apresentação da lista de cenários cadastrados é a primeira tela exibida ao clicar no item *Cenários* localizado no menu lateral esquerdo (menu principal). Esta tela é apresentada na Figura 16, que mostra um cenário cadastrado com o nome “Ambientação”.

Figura 16 – Tela de listagem dos cenários cadastrados.



Fonte: elaborado pelo autor.

A partir desta tela, o usuário do supervisor pode acessar as outras tarefas relativas ao gerenciamento de cenários:

- Criação de um novo cenário: através do botão *Criar novo Cenário*, localizado na região superior esquerda do cartão de cenários, que invoca a tela de criação.
- Modificação de um cenário existente: a partir do primeiro ícone da coluna *Opções*.
- Apagar um cenário: através do segundo ícone da coluna *Opções*.

O ícone de modificação de cenário (o primeiro na coluna opções) também é usado para exibir os detalhes do cenário. Como a configuração do cenário possui muitas informações, estes foram organizados em onze telas acessíveis através de abas. A alteração de um valor é salva automaticamente no banco de dados, portanto, não há um botão para confirmar alterações. A Figura 17 exibe a tela de detalhes do cenário, em que pode-se notar as onze abas no topo do cartão.

#### 4.1.2 RF-2 – Administrar currículos

A administração de currículos compreende a criação, a alteração e a exclusão de um currículo e a adição dos cenários que o compõem. A Figura 18 mostra a tela de Currículos.

Para alterar um currículo, utiliza-se o primeiro ícone da coluna *Opções*, que coloca a linha em modo de edição, onde pode-se alterar o nome e a descrição do currículo. A adição de cenários ao currículo se dá no cartão localizado abaixo da lista de currículos. Escolhe-se o currículo ao qual deseja-se adicionar um cenário e o cenário a ser adicionado e em seguida aciona-se o botão *Adicionar*.

Figura 17 – Tela de acesso aos detalhes do cenário.

**SIRV Supervisório**

**Cenário #2**

[Geral](#)
[Opções](#)
[Requisitos](#)
[Condições de operação](#)
[Rotas](#)
[Cargas](#)
[Eventos](#)
[Assistentes](#)
[Benchmarks](#)
[Erros de operação](#)
[Avaliação](#)

**Informações básicas**

Nome:

Descrição:

Sítio:

Propósito:

**Máquina a ser Operada**

Máquina:

Configuração:

**Opções de Assistência**

Fonte: elaborado pelo autor.

Figura 18 – Tela de administração de currículos.

**SIRV Supervisório**

**Currículos**

[Criar novo currículo](#)

#	Nome	Descrição	Cenários	Criação	Atualização	Opções
1	Básico	Operação básica	Ambientação Carregamento	09/02/2023	01/05/2023	<a href="#">✍</a> <a href="#">🗑</a>

**Adicionar um cenário a um currículo**

Currículo:   
 Cenário:

[Adicionar](#)

Fonte: elaborado pelo autor.

### 4.1.3 RF-3 – Administrar erros

Como dito anteriormente, os erros operacionais são condutas indevidas do treinando, que são detectadas pelo sistema de simulação e notificadas ao sistema supervisório, que então acrescenta o erro ao relatório da sessão de simulação ativa. Ao final da sessão, é possível visualizar a pontuação obtida.

A Figura 19 mostra a tela de administração dos erros do operador, onde pode-se acrescentar um novo erro ou modificar um erro existente.

Figura 19 – Tela de administração dos erros do operador.

#	Erro	Gatilhos	Pontos	Ativo	Categorias
1	Retardador automático desativado durante a condução	O interruptor do retardador automático estava na posição desligado durante a condução.	1.00	<input checked="" type="checkbox"/>	Manutenção, Segurança
2	Motor deu partida dentro de 10 segundos após pressionar a buzina	O operador deu partida no motor após 10 segundos de soar a buzina.	1.00	<input checked="" type="checkbox"/>	Segurança
3	Motor deu partida dentro de 5 segundos após pressionar a buzina	O operador deu partida no motor após 5 segundos de soar a buzina.	1.00	<input type="checkbox"/>	Segurança
4	Marcha muito alta para a inclinação	Selecionou uma marcha mais alta do que o esperado para um determinado grau (inclinação) e estado de carga.	1.00	<input checked="" type="checkbox"/>	Manutenção, Segurança, Eficiência de Combustível
5	Buzina não acionada corretamente antes de partir o motor	Falha ao soar o número correto de toques de buzina de aviso antes de ligar o motor.	1.00	<input type="checkbox"/>	Segurança
6	Buzina não acionada corretamente antes de mover para frente	Falha ao soar o número correto de toques de buzina antes de partir.	1.00	<input type="checkbox"/>	Segurança
7	Buzina não acionada corretamente antes de mover de ré	Falha ao soar o número correto de toques de buzina antes de se mover na direção reversa.	1.00	<input type="checkbox"/>	Segurança
8	Buzina não acionada uma vez antes de mover em frente	Falha ao soar o número correto de toques de buzina antes de partir.	1.00	<input type="checkbox"/>	Segurança
9	Buzina não acionada três vezes antes de reverter	Falha ao soar o número correto de toques de buzina antes de se mover na direção reversa.	1.00	<input type="checkbox"/>	Segurança
10	Buzina não acionada duas vezes antes de seguir em frente	Falha ao soar o número correto de toques de buzina antes de partir.	1.00	<input checked="" type="checkbox"/>	Segurança
11	Buzina não acionada duas vezes antes de reverter	Falha ao soar o número correto de toques de buzina antes de se mover na direção reversa.	1.00	<input type="checkbox"/>	Segurança
12	Temperatura alta do óleo do freio hidráulico	Permitiu que a temperatura do óleo do freio hidráulico ficasse muito alta. Exemplo: o operador faz mau uso do retardador/freio de serviço, especialmente ao descer uma rampa em uma marcha mais alta que a adequada.	1.00	<input checked="" type="checkbox"/>	Segurança, Manutenção

Fonte: elaborado pelo autor.

#### 4.1.4 RF-4 – Administrar eventos

Relembrando, os eventos de simulação são situações inesperadas que são introduzidas em uma sessão de treinamento ou avaliação, para averiguar a capacidade do treinando de responder adequadamente a elas, de acordo com procedimentos predefinidos.

A Figura 20 mostra a tela de administração dos eventos de treinamento, onde pode-se acrescentar um novo evento ou modificar um existente.

#### 4.1.5 RF-5 – Visualizar sítios

Um sítio é uma localização onde a simulação deve acontecer. Os sítios podem ser genéricos ou simularem minas reais. Na tela para visualizar os sítios instalados no sistema pode-se verificar a foto-mapa, locais de carga e descarga e áreas com objetivos específicos, por exemplo.

A Figura 21 mostra a tela de visualização dos sítios instalados. A princípio foi trabalhada a Mina de Conceição, que faz parte do complexo de minas à céu aberto localizadas no município de Itabira/MG. O sistema pretende ser genérico para aceitar a instalação de outros sítios.

Figura 20 – Tela de administração dos eventos de simulação.

Eventos de Simulação					
#	Evento	Estímulo	Critérios	Timeout	Categorias
0	Falha no sistema de carga da bateria	O indicador de falha do sistema de carga da bateria está aceso.	Pare a máquina. Aplique o freio de estacionamento. Desligue o motor. Peça assistência pelo rádio.	120.00	Manutenção
1	Falha de freio	O sistema de mensagens exibe um aviso de pressão de ar do retardador (ou acumulador). Para máquinas com freios hidráulicos/a ar, o freio de serviço e o retardador falham.	Pare a máquina. T139: Pode usar retardador para desacelerar a máquina. Aplique o freio de estacionamento. Aplique o freio de estacionamento. Desligue a máquina. Peça assistência pelo rádio.	60.00	Segurança
2	Alta temperatura do óleo do freio hidráulico	O sistema de mensagens exibe um aviso de alta temperatura do óleo do freio hidráulico. Quando disponível, a luz do painel de controle do teto acende.	Pare a máquina. Selecione "Neutral/Neutro" (não aplicável a alguns caminhões CAT). Aplique o freio de estacionamento e aumente o RPM para entre 1200 e 1500 por 5 segundos (não aplicável a alguns caminhões CAT).	120.00	Manutenção
3	Pára-brisas quebrado	O pára-brisas dianteiro fica gravemente danificado quando atingido por um projétil.	Pare a máquina completamente. Desligue o motor. Aplique o freio de estacionamento. Peça assistência pelo rádio.	60.00	Segurança
4	Nível do líquido de arrefecimento muito baixo	O sistema de mensagens exibe o aviso de nível de líquido de arrefecimento baixo.	Pare a máquina completamente. Selecione "Park/Estacionamento" e a máquina será desligada. Peça assistência pelo rádio.	30.00	Manutenção
5	Temperatura do líquido de arrefecimento muito alta	O sistema de mensagens exibe um aviso de alta temperatura do líquido de arrefecimento. Quando disponível, o medidor de temperatura do líquido de arrefecimento fica na zona vermelha.	Pare a máquina. Desligue a máquina. Aplique o freio de estacionamento.	160.00	Manutenção
6	Rádio comunica incidente de emergência	Um incidente de emergência não visível para o operador é transmitido pelo rádio.	Pare a máquina completamente. Aplique o freio de estacionamento e coloque a máquina na marcha neutra. Mantenha o veículo parado e mantenha o rádio em silêncio até que uma comunicação de rádio libere o tráfego.	15.00	Segurança
7	Falha do motor	O sistema de mensagens exibe um erro de controle do motor. O motor executa o desligamento automático.	Pare a máquina completamente. Aplique o freio de estacionamento. Peça assistência pelo rádio.	30.00	Manutenção
8	Nível de óleo do motor baixo	O sistema de mensagens exibe um aviso de nível baixo de óleo do motor.	Pare a máquina completamente. Aplique o freio de estacionamento. Desligue o motor. Peça assistência pelo rádio.	60.00	Manutenção
9	Filtro de combustível obstruído	O sistema de mensagens exibe um aviso de obstrução do filtro de combustível.	Faça uma chamada de serviço pelo rádio.	120.00	Manutenção
10	Nível de combustível baixo	O sistema de mensagens exibe um aviso de nível baixo de combustível. O medidor de nível de combustível está na zona baixa. Todos os caminhões.	Pare a máquina completamente. Aplique o freio de estacionamento. Peça assistência pelo rádio.	120.00	Produtividade
11	Baixa pressão da alta	O sistema de mensagens exibe um aviso de baixa pressão da direção	Pare a máquina completamente. Aplique o freio de estacionamento. Desligue o motor.	30.00	Manutenção

Fonte: elaborado pelo autor.

Figura 21 – Tela de visualização dos sítios instalados.

The screenshot shows the SIRV Supervisório interface. On the left is a navigation menu with categories: HOME (Nova Simulação), TREINAMENTO (Cenários, Currículos, Erros do Operador, Eventos, Sítios), ADMINISTRAR (Operadores, Grupos, Visualizar simulações), and FERRAMENTAS (Exportar dados, Importar dados, Atualizar com github). The main area is titled 'Sítios Disponíveis' and contains a table with the following data:

#	Nome	Versão	Status	Mapa
1	Mina Conceição	1.0	Em construção	

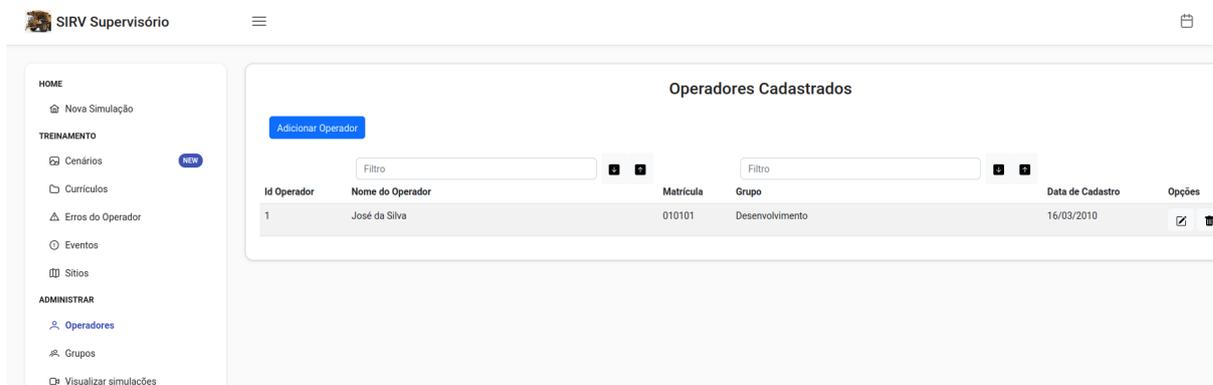
Fonte: elaborado pelo autor.

#### 4.1.6 RF-6 – Administrar operadores

Os dados dos operadores ou treinandos devem ser inseridos no sistema pelo operador antes que estes possam ser treinados ou avaliados. As tarefas de administração incluem incluir, alterar ou excluir um operador.

A Figura 22 mostra a tela de administração de operadores, onde pode-se incluir um novo operador ou modificar/excluir um operador existente.

Figura 22 – Tela de administração dos operadores cadastrados.



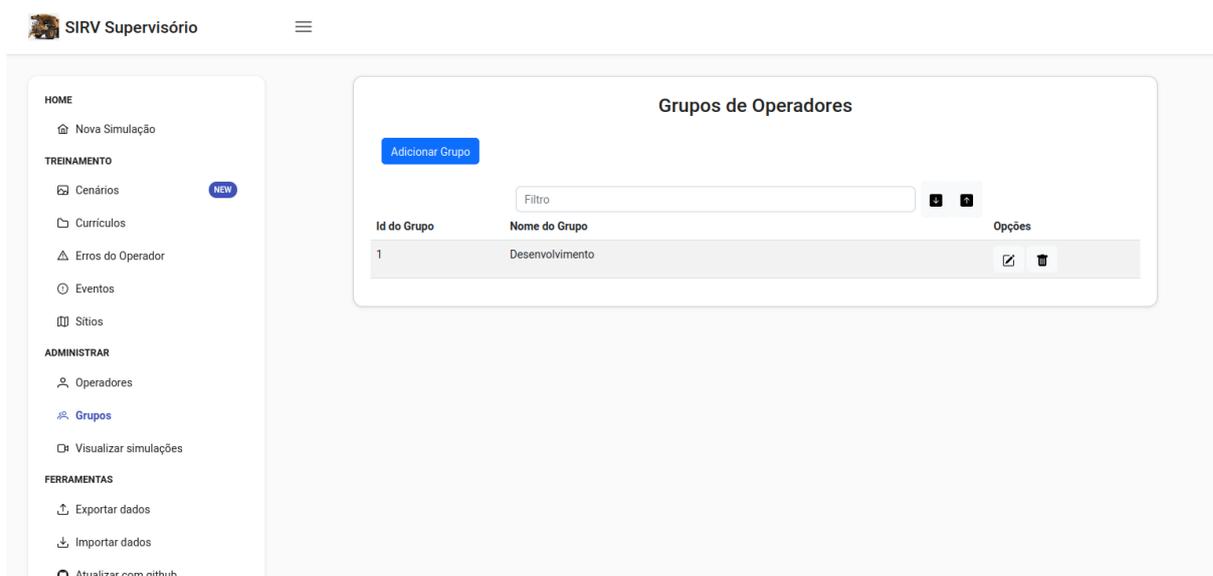
Fonte: elaborado pelo autor.

#### 4.1.7 RF-7 – Administrar grupos de operadores

Os operadores podem ser organizados em grupos, por exemplo, agrupando os operadores de uma mesma turma, o que facilita a pesquisa por um operador em específico para treinamento, avaliação ou revisão de sessão.

A Figura 23 mostra a tela de administração de grupos de operadores, onde pode-se criar um novo grupo ou modificar/excluir um grupo existente.

Figura 23 – Tela de administração dos grupos de operadores.

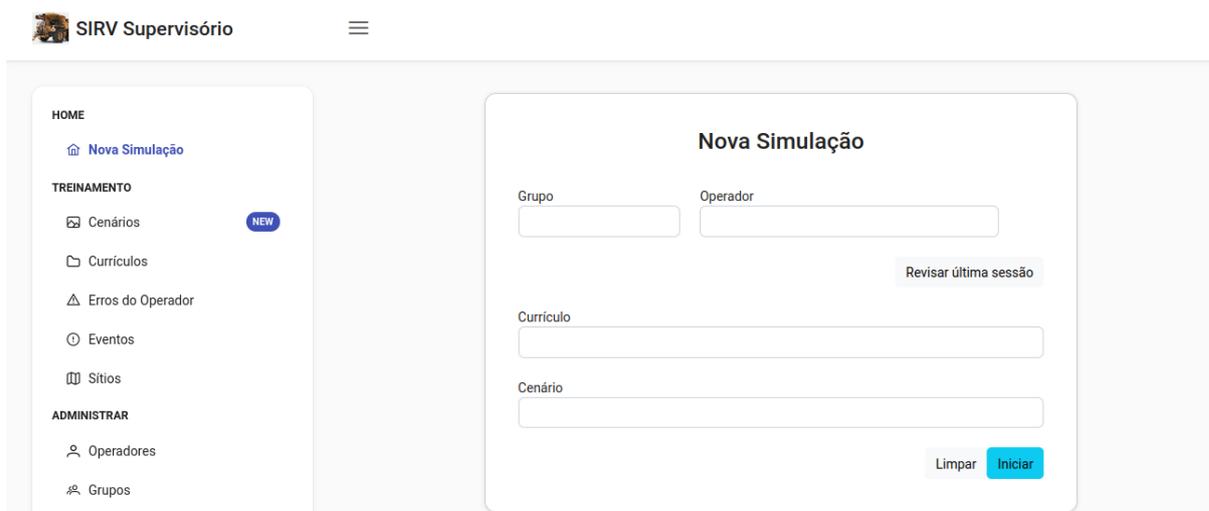


Fonte: elaborado pelo autor.

### 4.1.8 RF-8 – Administrar simulações

A administração das sessões de simulação é dividida em dois menus. O primeiro é o menu *Nova Simulação*, que oferece a funcionalidade para se iniciar uma sessão, informando o grupo, o operador, o currículo e o cenário a ser utilizado. Após informar os dados, o instrutor requisita o início da sessão, o que envia o comando de inicialização ao sistema de simulação. A Figura 24 mostra a tela de inicialização de uma sessão de simulação.

Figura 24 – Tela de inicialização da sessão de simulação.



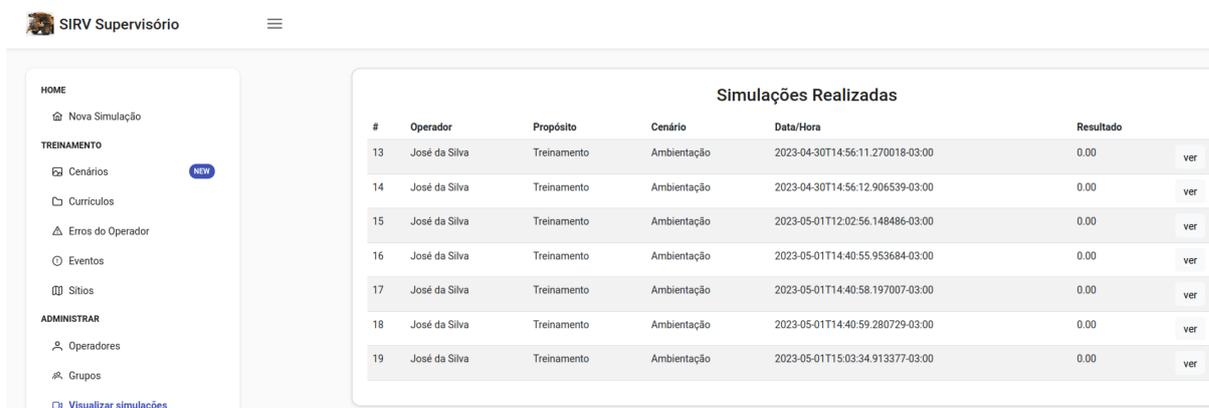
The screenshot shows the 'Nova Simulação' form. On the left is a sidebar menu with categories: HOME (Nova Simulação), TREINAMENTO (Cenários, Currículos, Erros do Operador, Eventos, Sítios), and ADMINISTRAR (Operadores, Grupos). The main form has the following elements:

- Group:
- Operator:
- Revisar última sessão:
- Curriculum:
- Scenario:
- Limpar:
- Iniciar:

Fonte: elaborado pelo autor.

O segundo menu associado às tarefas de simulação é o menu *Visualizar simulações*, que lista o histórico de sessões de simulação já realizadas. São listadas as informações básicas da sessão, como nome do operador, propósito, cenário utilizando data e hora do início da sessão e a pontuação resultante. A Figura 25 mostra a tela correspondente.

Figura 25 – Tela com as simulações realizadas.



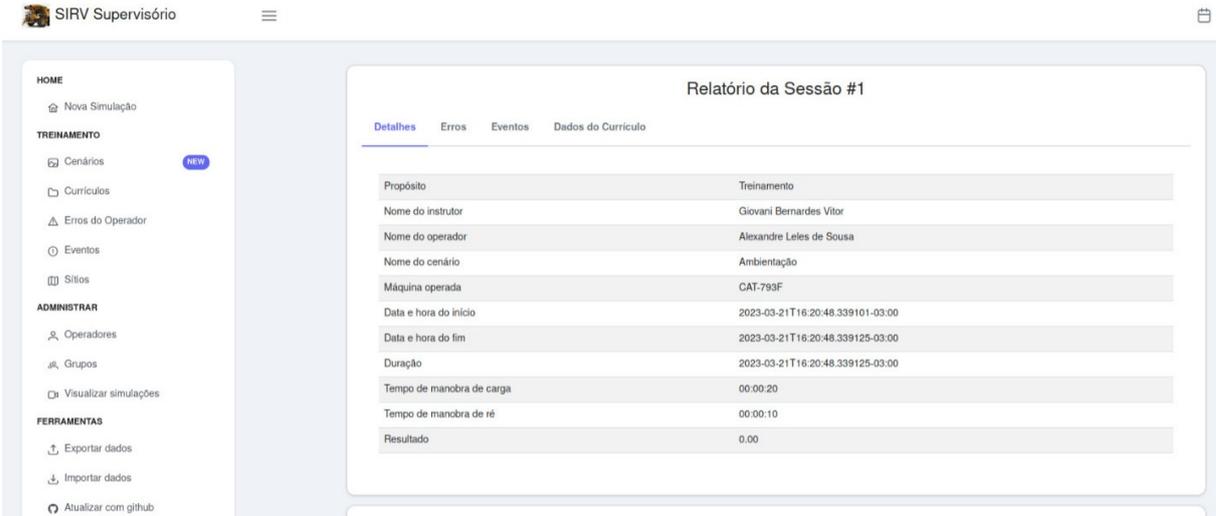
The screenshot shows the 'Visualizar simulações' screen. The sidebar menu is similar to the previous screen, but 'Visualizar simulações' is highlighted. The main content is a table titled 'Simulações Realizadas'.

#	Operador	Propósito	Cenário	Data/Hora	Resultado	
13	José da Silva	Treinamento	Ambientação	2023-04-30T14:56:11.270018-03:00	0.00	ver
14	José da Silva	Treinamento	Ambientação	2023-04-30T14:56:12.906539-03:00	0.00	ver
15	José da Silva	Treinamento	Ambientação	2023-05-01T12:02:56.148486-03:00	0.00	ver
16	José da Silva	Treinamento	Ambientação	2023-05-01T14:40:55.953684-03:00	0.00	ver
17	José da Silva	Treinamento	Ambientação	2023-05-01T14:40:58.197007-03:00	0.00	ver
18	José da Silva	Treinamento	Ambientação	2023-05-01T14:40:59.280729-03:00	0.00	ver
19	José da Silva	Treinamento	Ambientação	2023-05-01T15:03:34.913377-03:00	0.00	ver

Fonte: elaborado pelo autor.

Ao acionar o botão *Ver*, abre-se a tela de resultado da sessão, que mostra os dados referentes a sessão escolhida. Os dados estão divididos em quatro abas, que mostram os resultados obtidos pelo operador naquela sessão específica sendo estas: Detalhes, Erros, Eventos e Dados do Currículo, conforme mostra a Figura 26.

Figura 26 – Tela de resultado da sessão, mostrando detalhes, erros, eventos e dados do currículo em diferentes abas.



The screenshot shows the SIRV Supervisório interface. The main content area displays the 'Relatório da Sessão #1' page, which is divided into four tabs: 'Detalhes', 'Erros', 'Eventos', and 'Dados do Currículo'. The 'Detalhes' tab is active, showing a table with the following data:

Relatório da Sessão #1	
Propósito	Treinamento
Nome do instrutor	Giovani Bernardes Vitor
Nome do operador	Alexandre Leles de Sousa
Nome do cenário	Ambientação
Máquina operada	CAT-793F
Data e hora do início	2023-03-21T16:20:48.339101-03:00
Data e hora do fim	2023-03-21T16:20:48.339125-03:00
Duração	2023-03-21T16:20:48.339125-03:00
Tempo de manobra de carga	00:00:20
Tempo de manobra de ré	00:00:10
Resultado	0.00

Fonte: elaborado pelo autor.

#### 4.1.9 RF-9 – Fazer backup, importar e exportar e RF-10 Gerar relatórios

As rotinas de backup do banco de dados foram implementadas através da biblioteca Django Database Backup. Através desta, é possível configurar uma frequência na qual os backups automáticos são realizados e também o diretório onde estes ficarão salvos.

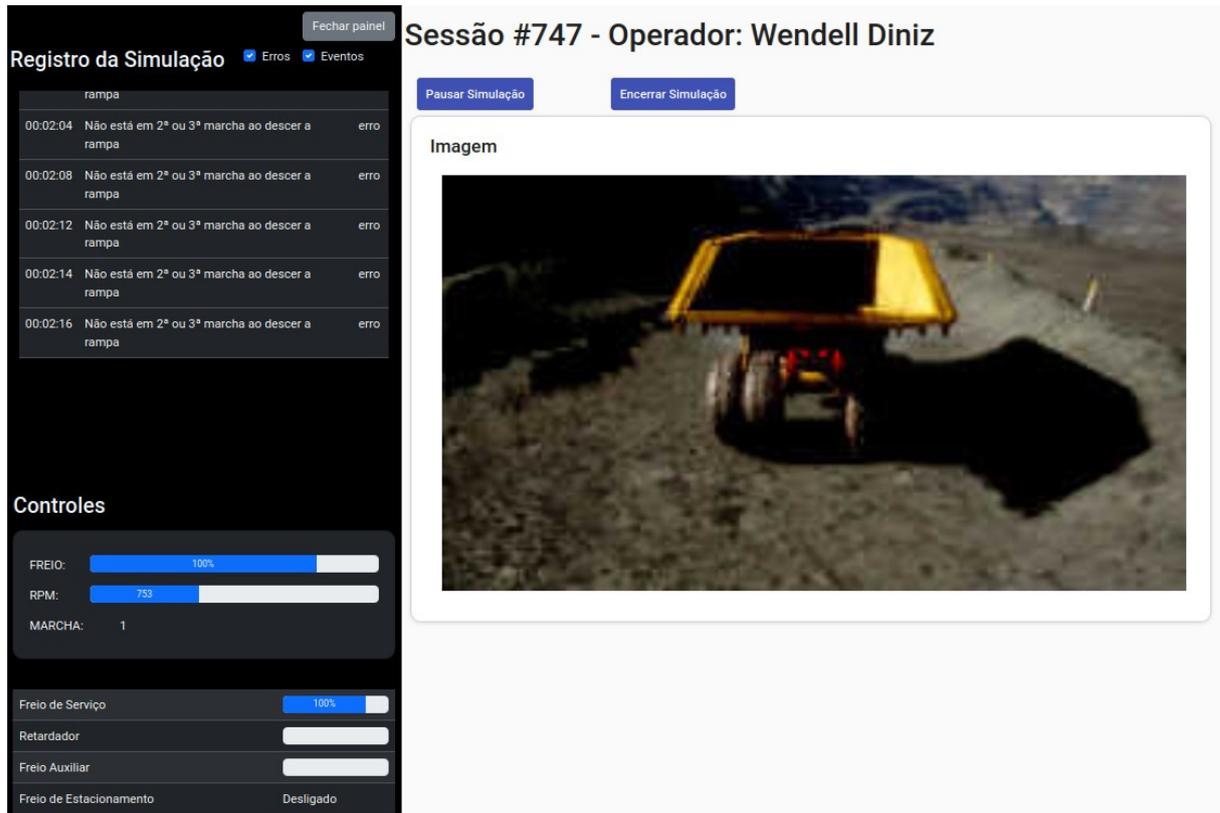
As rotinas de importação e exportação encontram-se pendentes, assim como as rotinas referentes ao RF-10 - Gerar relatórios, porém, os relatórios de sessão individual estão implementados.

#### 4.1.10 Integração supervisorio x simulador e geração das tarefas de simulação

A Figura 27 mostra a tela projetada para acompanhamento do instrutor da sessão de simulação em andamento.

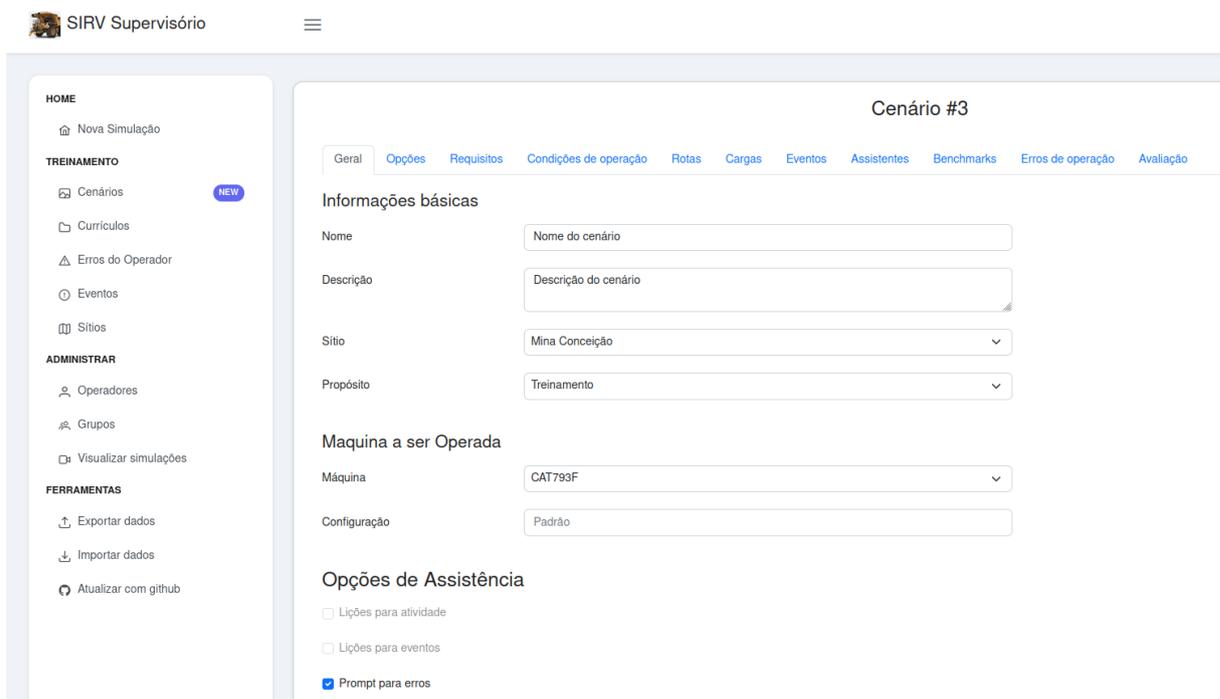
As configurações da sessão são definidas na tela referente ao cenário. Os grupos de configuração foram divididos em abas, para melhor organização da interface. A aba *Geral*, mostrada na Figura 28, apresenta as configurações básicas do cenário, a saber: nome, descrição, sítio onde ocorrerá a simulação, propósito, máquina a ser operada, configuração e opções de assistência.

Figura 27 – Tela de acompanhamento de simulação em tempo de execução.



Fonte: Vitor e Diniz (2023).

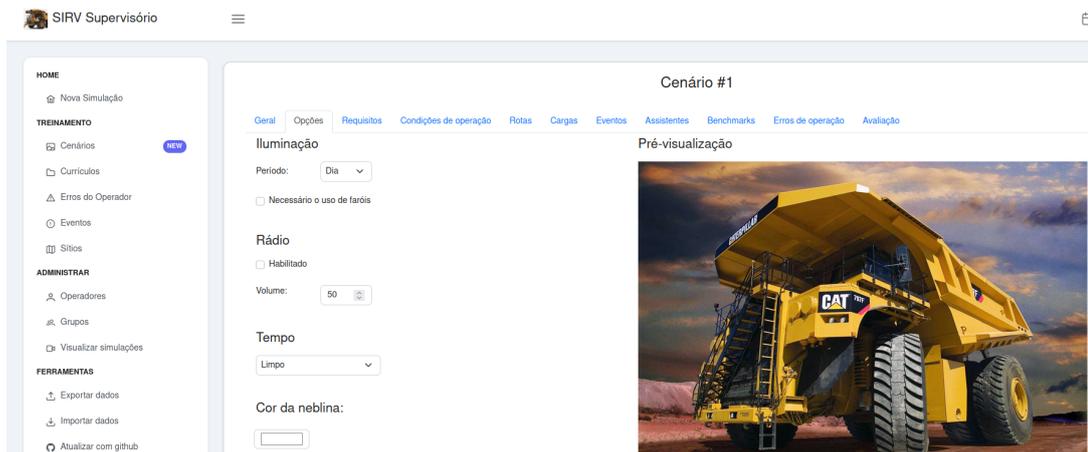
Figura 28 – Aba Geral da tela de detalhamento do cenário.



Fonte: elaborado pelo autor.

A aba *Opções*, mostrada na Figura 29, apresenta opções gerais da simulação referentes à iluminação (dia ou noite) e uso dos faróis, uso do rádio, clima (limpo ou chovendo) e das cores da neblina e da poeira. Uma pré-visualização das condições escolhidas deverá aparecer na posição da imagem de exemplo.

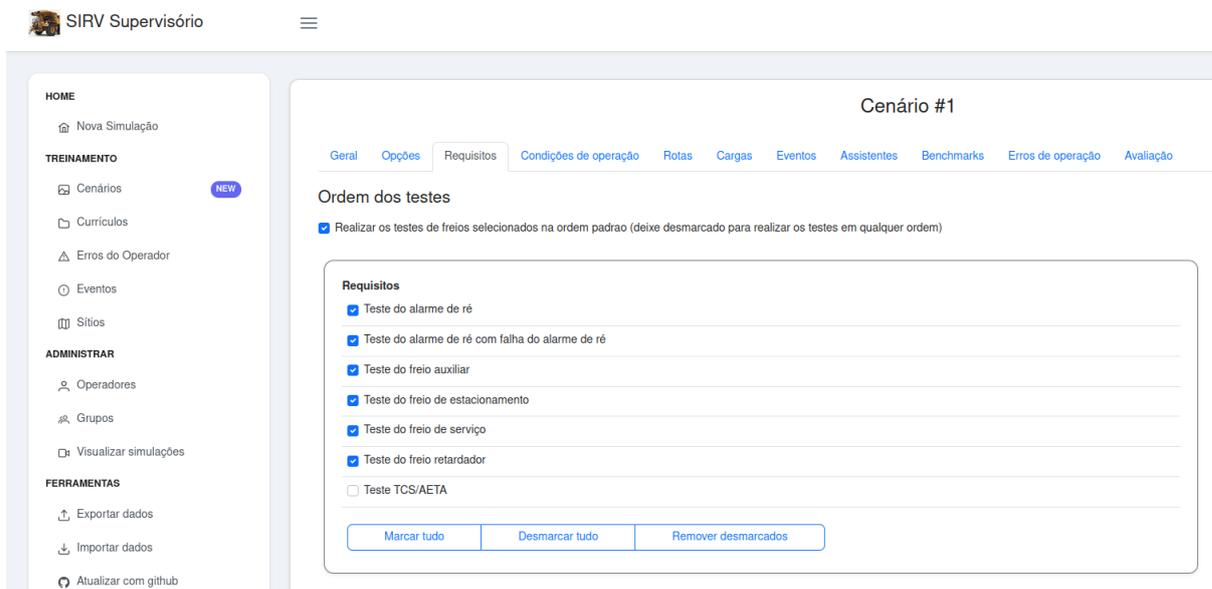
Figura 29 – Aba Opções da tela de detalhamento do cenário.



Fonte: elaborado pelo autor.

A aba *Requisitos*, mostrada na Figura 30, apresenta as opções dos testes de freio e os controles para marcá-los ou desmarcá-los, o que faz com que sejam exigidos ou não como parte dos critérios de avaliação nas sessões de simulação que utilizarem o cenário. Lembrando que tais funcionalidades estão desenvolvidas no supervisório e que ainda aguardam seu perfeito desenvolvimento por parte do sistema de simulação.

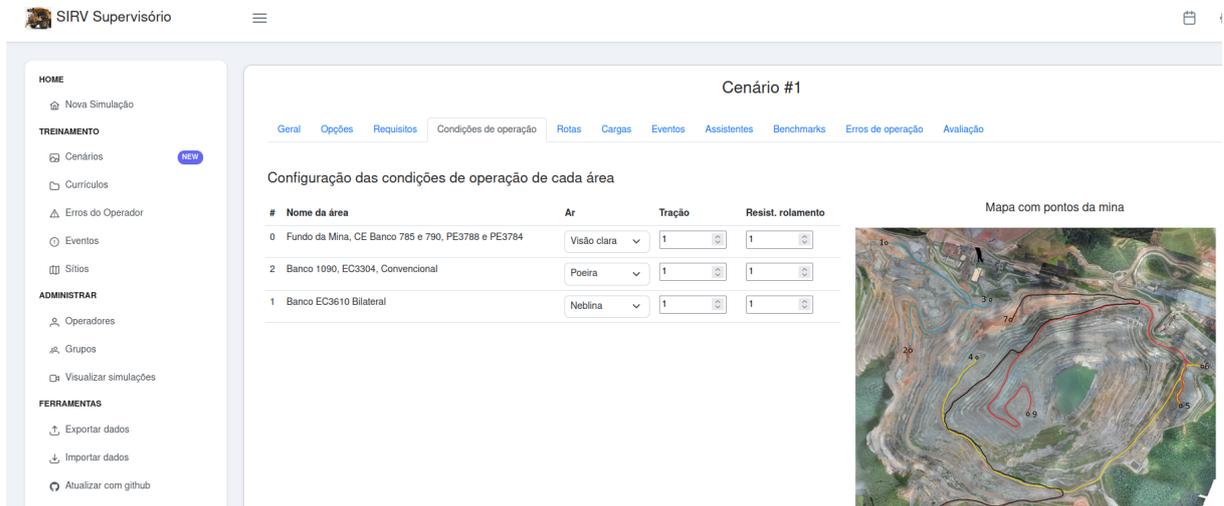
Figura 30 – Aba Requisitos da tela de detalhamento do cenário.



Fonte: elaborado pelo autor.

A aba *Condições de operação*, mostrada na Figura 31, abriga controles para determinar condições de visibilidade em pontos específicos do sítio, podendo escolher entre visão clara, neblina ou poeira.

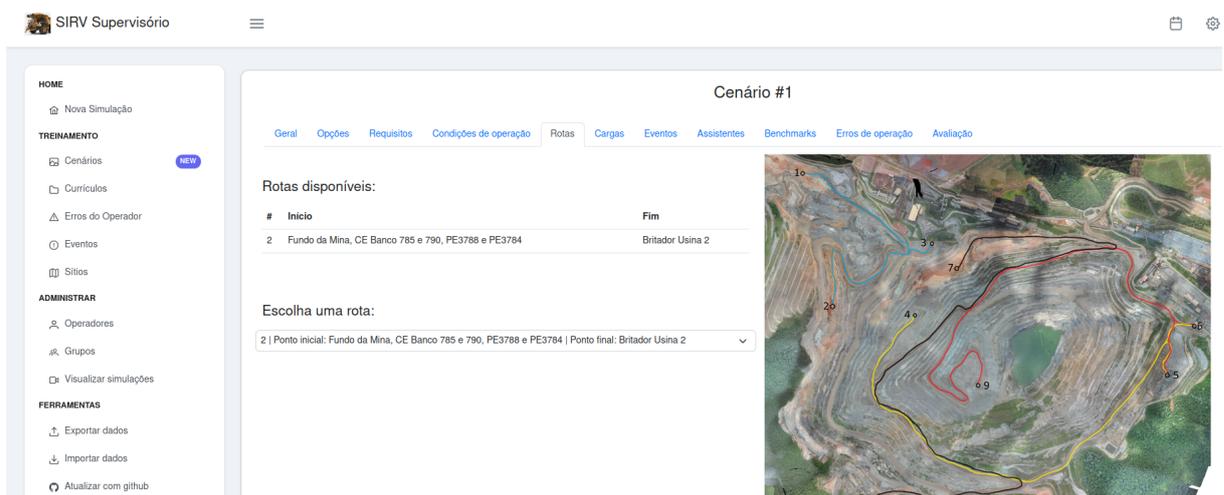
Figura 31 – Aba Condições de Operação da tela de detalhamento do cenário.



Fonte: elaborado pelo autor.

A aba *Rotas*, mostrada na Figura 32, mostra as rotas disponíveis no mapa do sítio definido para o cenário e dá a opção de escolher uma delas para a sessão de simulação.

Figura 32 – Aba Rotas da tela de detalhamento do cenário.



Fonte: elaborado pelo autor.

A aba *Cargas*, mostrada na Figura 33, permite definir se haverá outros veículos trafegando na rota e quais serão eles. Em uma mina real, vários tipos de veículo dividem as rotas com os caminhões, como picapes, microônibus, caminhão-pipa, escavadeiras etc. Ao passar pelos veículos, o operador do caminhão fora-de-estrada deve tomar algumas

ações, como parar o caminhão, comunicar via rádio com a central e aguardar o comando para continuar sua operação.

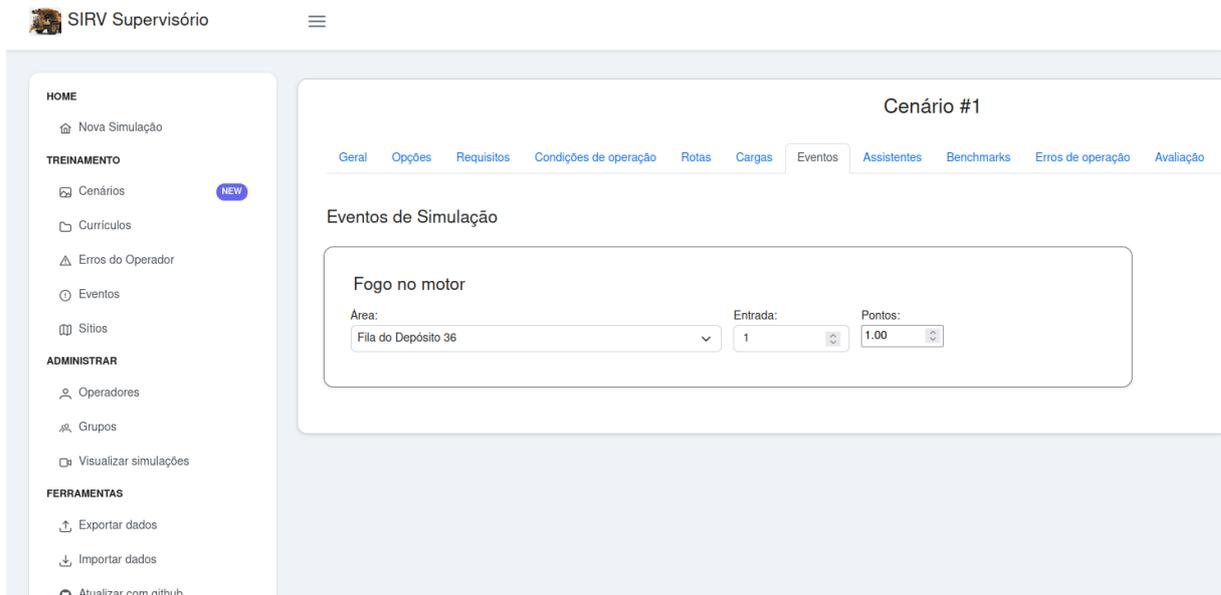
Figura 33 – Aba Cargas da tela de detalhamento do cenário.



Fonte: elaborado pelo autor.

A aba *Eventos* é apresentada na Figura 34. Ela mostra os eventos disponíveis para o cenário escolhido e permite definir a área onde o evento será disparado. A aba também permite definir a entrada e os pontos que devem ser atribuídos caso o operador não execute os procedimentos adequados ao evento ocorrido.

Figura 34 – Aba Eventos da tela de detalhamento do cenário.

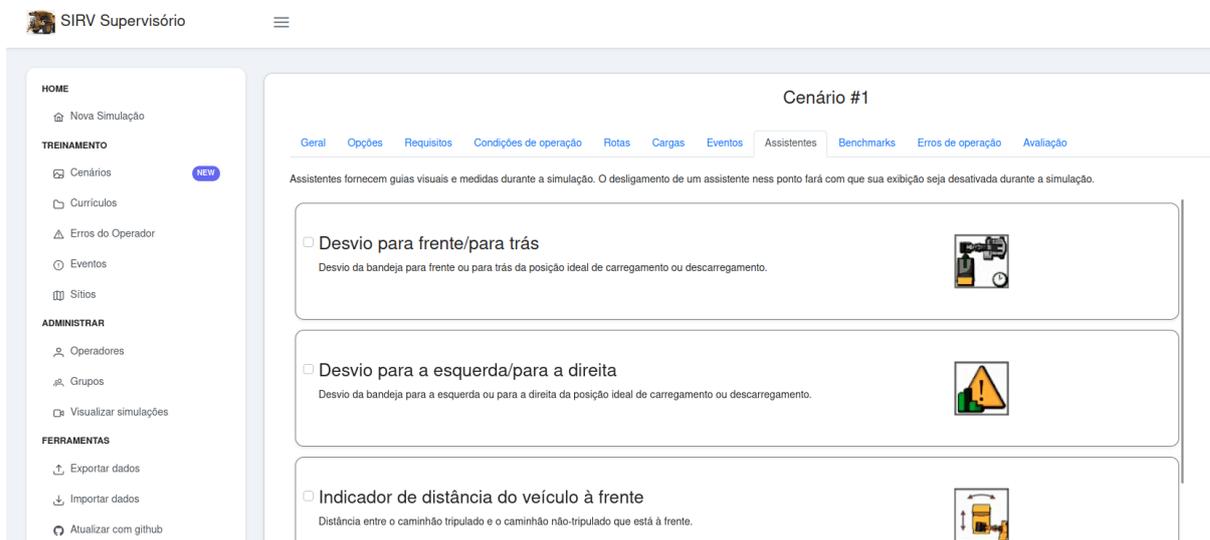


Fonte: elaborado pelo autor.

A aba *Assistentes*, mostrada na Figura 35, permite escolher quais assistentes estarão ativos durante a sessão de simulação. Os assistentes são ferramentas do caminhão fora-de-estrada para auxiliar o operador. Os exemplos mostrados na figura são para exibir

informações do caminhão sobre desvios para frente ou para trás, para a esquerda ou para a direita, e a distância do veículo à frente.

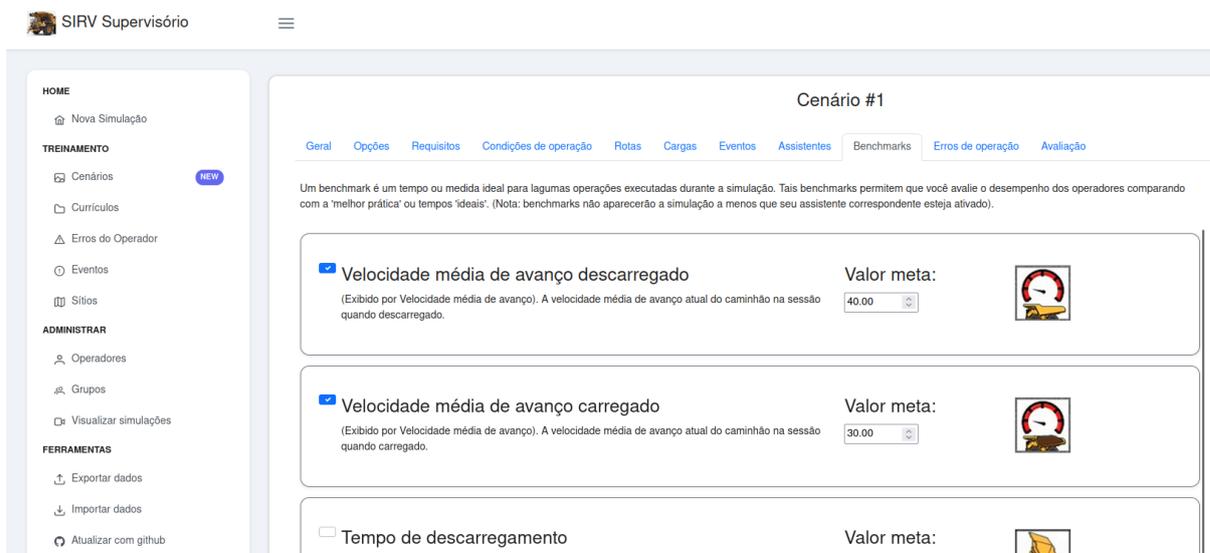
Figura 35 – Aba Assistentes da tela de detalhamento do cenário.



Fonte: elaborado pelo autor.

A aba *Benchmarks*, mostrada na Figura 36, exibe os tempos de referência para tarefas específicas da sessão, permitindo escolher quais deles estarão ativos e também definir o valor do tempo de referência para os mesmos.

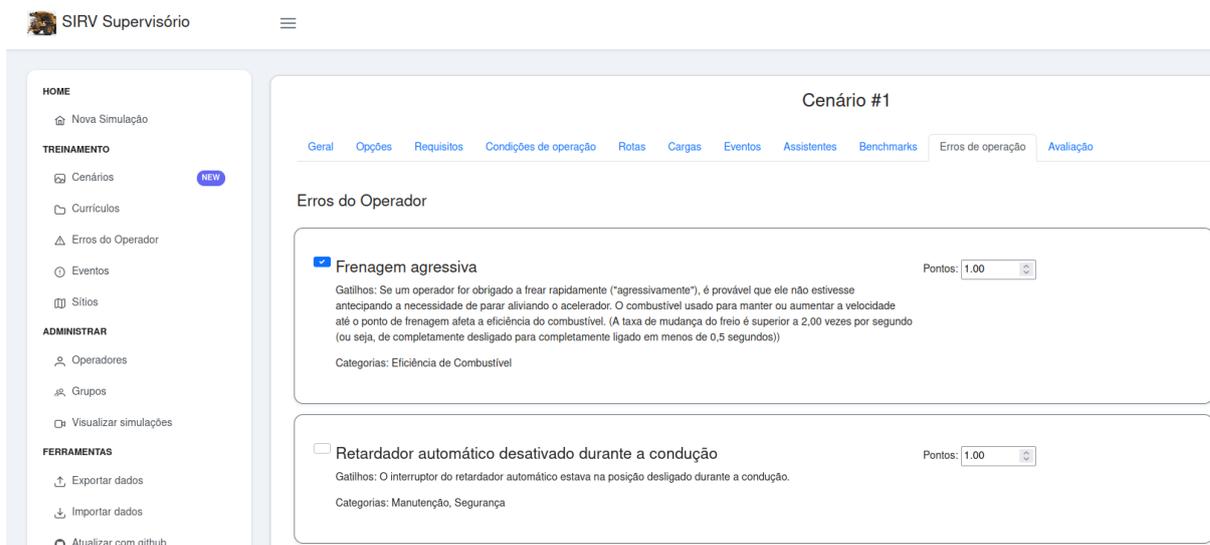
Figura 36 – Aba Benchmarks da tela de detalhamento do cenário.



Fonte: elaborado pelo autor.

A aba *Erros de operação*, mostrada na Figura 37, agrupa os erros definidos no cenário, onde pode-se escolher quais estarão ativos e a pontuação a ser atribuída quando um erro é detectado e reportado pelo sistema de simulação.

Figura 37 – Aba Erros de operação da tela de detalhamento do cenário.



Fonte: elaborado pelo autor.

A aba *Avaliação*, mostrada na Figura 38, exibe as opções de configuração dos critérios de avaliação da sessão: a pontuação máxima de erros/custos, a quantidade admitida de ocorrências de erros específicos, a quantidade admitida de falhas em eventos e o tempo máximo da sessão.

Figura 38 – Aba Avaliação da tela de detalhamento do cenário.



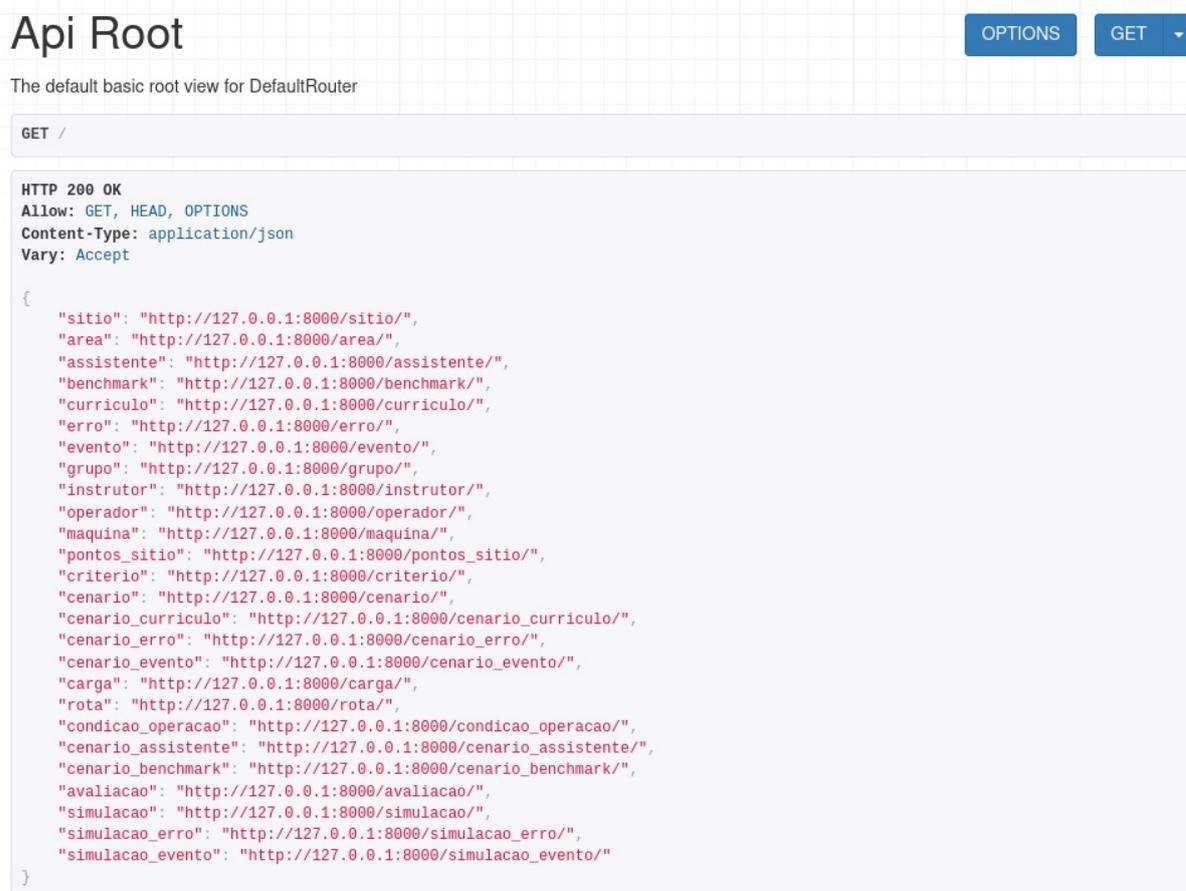
Fonte: elaborado pelo autor.

Os dados de configuração do cenário são transmitidos ao simulador no início de uma sessão de simulação via requisição de serviço do ROS2. O motor gráfico do simulador recebe os dados e procede os ajustes no cenário de acordo com os valores recebidos.

## 4.2 Resultados Quantitativos

A quantidade de artefatos produzidos no desenvolvimento relatado neste trabalho pode ser resumida da seguinte forma: 01 documento de visão, 03 documentos de requisitos, 01 modelo lógico do Banco de Dados, 04 diagramas conceituais, 04 relatórios gerenciais, 01 artigo científico em inglês, 01 API com 26 endpoints (figura 39), 01 interface para administrador de dados e 23 telas funcionais da interface do usuário do sistema supervisorio que foram apresentadas na seção anterior.

Figura 39 – API com os *endpoints* do *back-end*.



The screenshot shows a web interface for an API root endpoint. At the top, there is a title "Api Root" and two buttons: "OPTIONS" and "GET". Below the title, it says "The default basic root view for DefaultRouter". The main content area shows the response for a GET request to the root endpoint. The response is an HTTP 200 OK with headers: Allow: GET, HEAD, OPTIONS; Content-Type: application/json; Vary: Accept. The body of the response is a JSON object containing 26 endpoints, each with a name and a URL.

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
  "sitio": "http://127.0.0.1:8000/sitio/",
  "area": "http://127.0.0.1:8000/area/",
  "assistente": "http://127.0.0.1:8000/assistente/",
  "benchmark": "http://127.0.0.1:8000/benchmark/",
  "curriculo": "http://127.0.0.1:8000/curriculo/",
  "erro": "http://127.0.0.1:8000/erro/",
  "evento": "http://127.0.0.1:8000/evento/",
  "grupo": "http://127.0.0.1:8000/grupo/",
  "instrutor": "http://127.0.0.1:8000/instrutor/",
  "operador": "http://127.0.0.1:8000/operador/",
  "maquina": "http://127.0.0.1:8000/maquina/",
  "pontos_sitio": "http://127.0.0.1:8000/pontos_sitio/",
  "criterio": "http://127.0.0.1:8000/criterio/",
  "cenario": "http://127.0.0.1:8000/cenario/",
  "cenario_curriculo": "http://127.0.0.1:8000/cenario_curriculo/",
  "cenario_erro": "http://127.0.0.1:8000/cenario_erro/",
  "cenario_evento": "http://127.0.0.1:8000/cenario_evento/",
  "carga": "http://127.0.0.1:8000/carga/",
  "rota": "http://127.0.0.1:8000/rota/",
  "condicao_operacao": "http://127.0.0.1:8000/condicao_operacao/",
  "cenario_assistente": "http://127.0.0.1:8000/cenario_assistente/",
  "cenario_benchmark": "http://127.0.0.1:8000/cenario_benchmark/",
  "avaliacao": "http://127.0.0.1:8000/avaliacao/",
  "simulacao": "http://127.0.0.1:8000/simulacao/",
  "simulacao_erro": "http://127.0.0.1:8000/simulacao_erro/",
  "simulacao_evento": "http://127.0.0.1:8000/simulacao_evento/"
}
```

Fonte: elaborado pelo autor.

## 5 Conclusões

A integração das tecnologias da Indústria 4.0 na mineração tem o potencial de revolucionar o setor, melhorando a eficiência operacional, a segurança, a sustentabilidade e o crescimento econômico. À medida que a indústria da mineração continua a adaptar-se à era digital, é essencial que as empresas do setor invistam nas tecnologias transformadoras e adaptem as suas estratégias para se manterem competitivas num cenário econômico e social em rápida evolução.

Este trabalho apresentou como foi realizado o desenvolvimento de um sistema supervisorio para um simulador 3D de caminhão fora-de-estrada de mineração. Este supervisorio é independente do sistema de simulação e é responsável por gerenciá-lo e controlá-lo, auxiliando o processo de capacitação da mão de obra relacionada à operação destas máquinas. O objetivo de desenvolver o sistema e sua documentação foi atingido, bem como foram alcançados todos os objetivos específicos estipulados.

As limitações do sistema supervisorio estão relacionadas à necessidade de aguardar a evolução do sistema de simulação para ser possível realizar testes satisfatórios. Este trabalho não contemplou a análise das fases de entrega e validação do sistema, por que estas ainda não foram realizadas no projeto. Questões de segurança e confidencialidade limitaram o detalhamento completo do supervisorio, porém, o que foi apresentado neste documento é suficiente para entender as principais funcionalidades do sistema desenvolvido.

Como contribuições, este trabalho mostrou como o desenvolvimento de tecnologia para simulação de caminhões fora-de-estrada de mineração pode ser feito, considerando uma solução customizada, porém, que pode ser aplicada em outros contextos. O trabalho também contribuiu para consolidar a apreensão do conhecimento relacionado a tecnologia de simulação e correlacionadas, além de oferecer aos envolvidos no projeto uma rica experiência de trabalho em equipe no desenvolvimento de tecnologia para a indústria.

Como trabalhos futuros, pode-se pensar na melhoria contínua do sistema, de modo a satisfazer todas as necessidades atuais e futuras da organização, bem como a manutenção e evolução do *software*, ampliação do escopo, abrangendo outras máquinas, e integração com outros sistemas, por exemplo, gestão de recursos humanos ou gêmeos idênticos (*digital twin*). Como forma de reduzir a subjetividade na avaliação do desempenho do operador do caminhão, pode-se estudar a aplicação de algoritmos de inteligência artificial para traçar o perfil ótimo de condução destas máquinas, a fim de utilizar o modelo resultante no esquema de avaliação do sistema supervisorio construído.

# Referências

- ALI, M. et al. An e-learning platform based on mvc architecture and django rest framework with vue.js user interface. *International Journal of Advanced Research in Computer Science and Software Engineering*, v. 1, p. 8–14, 2019. Citado na página 21.
- ASSEFAW, D. *Building RESTful APIs with Python: Django REST Framework*. [S.l.]: Packt Publishing, 2015. Citado na página 20.
- BELOGLAZOV, I.; PETROV, P.; BAZHIN, V. The concept of digital twins for tech operator training simulator design for mining and processing industry. *Eurasian Mining*, v. 2, p. 50–54, 12 2020. Citado na página 16.
- BIGLIARDI, B.; FILIPPELLI, S.; TAGLIENTE, L. Industry 4.0 and open innovation: evidence from a case study. *Procedia Computer Science*, v. 200, p. 1796–1805, 2022. ISSN 1877-0509. 3rd International Conference on Industry 4.0 and Smart Manufacturing. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S1877050922003891>>. Citado na página 16.
- BOJAN, M. *Vue.js 2 - Building Responsive Web Apps*. [S.l.]: Manning Publications, 2017. Citado na página 21.
- CHEN, J.; ZHOU, J. Revisiting industry 4.0 with a case study. In: . [S.l.: s.n.], 2018. p. 1928–1932. Citado na página 12.
- CRUZARA, G. et al. O impacto da transformação digital e da indústria 4.0 nos aspectos de valor: Evidências de uma meta-síntese. *Contextus – Revista Contemporânea de Economia e Gestão*, v. 18, p. 92–106, jun. 2020. Disponível em: <<http://www.periodicos.ufc.br/contextus/article/view/43717>>. Citado na página 12.
- FOWLER, M. *Patterns of Enterprise Application Architecture: A Guide to Object-Oriented Design, Integration, and Distributed Systems*. [S.l.]: Ed. Campus, 2003. Citado na página 19.
- GERMANY, D. *A mineração no Brasil*. Rio de Janeiro: Fundo Setorial Mineral, 2002. Citado 3 vezes nas páginas 12, 13 e 14.
- HASAN, K. *What, Why and How of ROS*. 2019. Acessado em 24 de Julho de 2024. Disponível em: <<https://towardsdatascience.com/what-why-and-how-of-ros-b2f5ea8be0f3>>. Citado 2 vezes nas páginas 22 e 23.
- HYNDMAN, M. *Django REST Framework 3.x Cookbook*. Beijing: Packt Publishing, 2016. Citado na página 20.
- JAIN, A. et al. An enhanced and interactive training model for underground coal mines using virtual reality. In: *2023 Second International Conference on Electronics and Renewable Systems (ICEARS)*. [S.l.: s.n.], 2023. p. 361–365. Citado na página 17.
- LARMAN, G. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design*. [S.l.]: Pearson Education, 2004. Citado na página 19.

- LAUDON, K.; LAUDON, J. *Sistema de informação gerenciais: administrando a organização digital*. São Paulo: Prentice Hall, 2011. Citado na página 17.
- LööW, J.; ABRAHAMSSON, L.; JOHANSSON, J. Mining 4.0 - the impact of new technology from a work place perspective. *Mining, Metallurgy and Exploration*, v. 36, p. 701–707, 2019. Citado na página 12.
- MCMAHAN, P. et al. Virtual environments for surface mining powered haulage training. *Extracting the Science: A Century of Mining Research*, p. 520–528, 2010. Citado na página 16.
- NGOC, H.; LASA, G.; IRIARTE, I. Human-centred design in industry 4.0: case study review and opportunities for future research. *Journal of Intelligent Manufacturing*, p. 35–76, 1 2022. Citado na página 16.
- NILAY, D. *A Complete Guide And Comparison Of MVC and MVVM*. 2021. Acessado em 26 de Julho de 2024. Disponível em: <<https://www.intuz.com/blog/guide-on-mvc-vs-mvvm>>. Citado na página 20.
- OPEN\_ROBOTICS. *ROS.ORG*. 2021. Acessado em 08 de Maio de 2024. Disponível em: <<https://www.ros.org/>>. Citado 2 vezes nas páginas 22 e 23.
- PAGE, R. Brief history of flight simulation. *SimTecT 2000 proceedings*, 2000. Citado na página 18.
- RIBEIRO, V. et al. Knowledge management and industry 4.0: a critical analysis and future agenda. *Gestão e Produção*, p. 1–17, 10 2022. Citado na página 12.
- ROMERAL, P. et al. Development of managerial and information technology skills in learning factories in the context of industry 4.0: a case study. *Revista Gestão da Produção Operações e Sistemas*, v. 16, n. 2, p. 195, jun. 2021. Disponível em: <<https://revista.feb.unesp.br/gepros/article/view/2785>>. Citado na página 16.
- SILVA, B. *Operação em simuladores de alta fidelidade*. [S.l.], 2021. Citado 2 vezes nas páginas 10 e 13.
- SKOVGAARD, J. *Architecting Enterprise Solutions: Integrating Legacy Systems, Web Services, and Mobile Applications*. [S.l.]: Ed. Elsevier, 2007. Citado 2 vezes nas páginas 19 e 21.
- SOUZA, M. et al. Industry 4.0: An analysis of the production of scientific papers based on indicators. *Research, Society and Development*, v. 10, n. 11, p. e193101118923, Aug. 2021. Disponível em: <<https://rsdjournal.org/index.php/rsd/article/view/18923>>. Citado na página 16.
- STEVENSON, D. *Developing RESTful APIs with Django 2.0*. [S.l.]: Apress, 2013. Citado 2 vezes nas páginas 20 e 21.
- TOTVS. *Entenda o que é SCADA e o seu papel dentro da indústria*. 2023. Acessado em 08 de Maio de 2024. Disponível em: <<https://www.totvs.com/blog/gestao-industrial/scada/>>. Citado na página 18.
- VITOR, G.; DINIZ, W. Relatórios periódicos: Desenvolvimento do protótipo de hardware e software para simulação do cat 793f. 2023. Citado 4 vezes nas páginas 11, 26, 27 e 50.

- WYK, E. van; VILLIERS, R. de. Virtual reality training applications for the mining industry. *AFRIGRAPH '09: Proceedings of the 6th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, v. 1, p. 53–63, 2 2009. Citado na página 16.
- YANG, F.; GU, S. Industry 4.0, a revolution that requires technology and national strategies. *Complex and Intelligent Systems*, p. 1311–1325, 6 2021. Citado na página 16.
- YOU, E. *VUEJS.ORG*. 2024. Acessado em 08 de Maio de 2024. Disponível em: <<https://vuejs.org/>>. Citado na página 21.
- YU, L. et al. Establishing design computing and extended reality facilities for remote virtual reality training. In: *2023 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. [S.l.: s.n.], 2023. p. 216–220. Citado na página 17.
- YU, Y. *Vue.js 2: Up and Running - Building Responsive Web Applications*. [S.l.]: Packt Publishing, 2016. Citado na página 21.
- YU, Y. A library management system based on mvc architecture and django rest framework. *International Journal of Advanced Computer Science and Applications*, v. 11, p. 1014–1019, 2018. Citado na página 21.