Universidade Federal de Ouro Preto
Escola de Minas
Engenharia de Controle e Automação

Lavínia Camile Alves Teixeira

Desenvolvimento de uma Ferramenta de Voz para Auxílio no Monitoramento de Alarmes Industriais

Ouro Preto
2026

Lavínia Camile Alves Teixeira

Desenvolvimento de uma Ferramenta de Voz para Auxílio no Monitoramento de Alarmes Industriais

Trabalho apresentado ao Colegiado do Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como parte dos requisitos para a obtenção do Grau de Engenheiro(a) de Controle e Automação.

Orientador: Dr[a].Karla Boaventura Pimenta Palmieri

Ouro Preto
2026

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
ESCOLA DE MINAS
DEPARTAMENTO DE ENGENHARIA CONTROLE E AUTOMACAO

FOLHA DE APROVAÇÃO

Lavínia Camile Alves Teixeira

Desenvolvimento de uma Ferramenta de Voz para Auxílio no Monitoramento de Alarmes Industriais

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Engenheira

Aprovada em 30 de junho de 2026

Membros da banca

Dra. Karla Boaventura Pimenta Palmieri - Orientador(a) - Universidade Federal de Ouro Preto
Dra. Luciana Gomes Castanheira - Universidade Federal de Ouro Preto
Dr. Caio Fernando Teixeira Portela - Universidade Federal de Ouro Preto

Karla Boaventura Pimenta Palmieri, orientadora do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 07/07/2026

Documento assinado eletronicamente por **Karla Boaventura Pimenta Palmieri**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 07/07/2026, às 15:43, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **1136333** e o código CRC **815C3CEA**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: 3135591533 - www.ufop.br

Agradecimentos

Aos meus pais, Sidney José Teixeira e Cristiane Maria de Andrade Teixeira, meu agradecimento mais profundo. Obrigada por serem porto seguro nas tempestades, por compreenderem minhas ausências e por segurarem a minha mão nos momentos de cansaço, sem nunca me deixar desistir dos meus sonhos. Um obrigado especial ao meu pai, que não mediu esforços para me apoiar e acreditou na minha força mesmo quando eu mesma duvidava dela. Esta conquista é, acima de tudo, de vocês.

À minha tia Helenice Nascimento, que desde a infância me mostrou, com carinho e gestos simples, o valor real da educação. Obrigada pela paciência nas tarefas de casa, por cada ensinamento e por me fazer enxergar, desde cedo, que o conhecimento abre caminhos que antes pareciam impossíveis. Muito da mulher que me tornei hoje carrega a sua essência.

Aos meus irmãos, Pedro Lucas Alves Teixeira e Ísis de Andrade Teixeira, meus parceiros de vida. Obrigada por trazerem leveza, risadas e o apoio diário que eu precisava para enfrentar os dias mais pesados. Ter vocês por perto deixou essa caminhada muito mais bonita e cheia de afeto.

À Universidade Federal de Ouro Preto (UFOP), por ter sido o cenário de tantas vivências marcantes e o lugar onde cresci não só como engenheira, mas como pessoa.

Aos meus amigos, especialmente à Ana Clara Neves de Almeida (Pepeta) e ao Daniel Rodrigues de Vasconcelos, que estiveram ao meu lado em cada passo dessa caminhada. Obrigada por dividirem comigo os desesperos, as madrugadas de estudo, as inseguranças e, finalmente, as comemorações. Estendo meu carinho a todos os colegas de curso que tornaram os anos de graduação inesquecíveis.

À minha orientadora, Dra. Karla Boaventura Pimenta Palmieri, pela paciência, dedicação e pela condução precisa durante o desenvolvimento deste trabalho. Sua orientação foi fundamental para que este projeto ganhasse vida.

"Quando a mulher negra se movimenta, toda a estrutura da sociedade se movimenta com ela."

— Angela Davis.

Resumo

O aumento da complexidade dos processos industriais supervisionados por sistemas SCADA (*Supervisory Control and Data Acquisition*) pode ampliar a quantidade de informações e alarmes apresentados ao operador. Em situações críticas, a ocorrência simultânea de alertas exige processos de percepção, atenção seletiva, memória de trabalho e tomada de decisão sob pressão temporal. Quando os eventos não são organizados de forma clara, pode haver dificuldade para identificar qual condição demanda atenção prioritária. Neste Trabalho de Conclusão de Curso, foi desenvolvida uma ferramenta de voz para auxílio no monitoramento de alarmes industriais. O projeto foi fundamentado em princípios da Ergonomia Cognitiva e na utilização complementar dos canais visual e auditivo, sem substituir a supervisão visual realizada no sistema. Utilizou-se o software Elipse E3 para a simulação de uma caldeira industrial, o Prosys OPC UA (*Open Platform Communications Unified Architecture*) Simulation Server para a disponibilização das variáveis e uma aplicação em Python com síntese de voz offline, denominada *Text-to-Speech* (TTS). Como resultado, foi implementada uma lógica de leitura, filtragem e priorização de alarmes capaz de evitar repetições desnecessárias, impedir a sobreposição de mensagens e descartar alertas cuja condição tenha sido normalizada antes da reprodução. A validação foi realizada em ambiente simulado, verificando o funcionamento técnico da comunicação OPC UA e da fila de prioridades. Não foram realizados testes com operadores humanos; portanto, não foi mensurada diretamente a redução da carga mental ou do tempo de resposta.

Palavras-chave: Sistemas SCADA; Ergonomia Cognitiva; Alarme por Voz; Protocolo OPC UA; Elipse E3.

Abstract

The increasing complexity of industrial processes monitored by Supervisory Control and Data Acquisition (SCADA) systems can increase the amount of information and alarms presented to operators. In critical situations, simultaneous alerts require perception, selective attention, working memory, and decision-making under time pressure. When alarms are not clearly organized, operators may have difficulty identifying which condition requires priority attention. This Final Course Project developed a voice-based tool to support the monitoring of industrial alarms. The project was based on principles of Cognitive Ergonomics and on the complementary use of visual and auditory channels, without replacing the visual supervision performed through the SCADA system. The Elipse E3 software was used to simulate an industrial boiler, while the Prosys OPC Unified Architecture (OPC UA) Simulation Server provided the process variables. A Python application with offline Text-to-Speech (TTS) functionality was developed to generate voice notifications. As a result, an alarm reading, filtering, and prioritization logic was implemented to prevent unnecessary repetitions, avoid message overlap, and discard alerts whose conditions had returned to normal before playback. Validation was conducted in a simulated environment, confirming the technical operation of OPC UA communication and the priority queue. No tests were performed with human operators; therefore, reductions in mental workload or response time were not directly measured.

Keywords: SCADA Systems; Cognitive Ergonomics; Voice Alarm; OPC UA Protocol; Elipse E3.

Lista de figuras

Lista de tabelas

Lista de abreviaturas e siglas

AET	Análise Ergonômica do Trabalho
AES	*Advanced Encryption Standard* (Padrão de Criptografia Avançado)
ANSI	*American National Standards Institute* (Instituto Nacional Americano de Padrões)
API	*Application Programming Interface* (Interface de Programação de Aplicativos)
CLP	Controlador Lógico Programável
DCOM	*Distributed Component Object Model* (Modelo de Objeto de Componente Distribuído)
ERP	*Enterprise Resource Planning* (Planejamento de Recursos Empresariais)
IA	Inteligência Artificial
IHM	Interface Homem-Máquina
ISA	*International Society of Automation* (Sociedade Internacional de Automação)
MES	*Manufacturing Execution System* (Sistema de Execução de Manufatura)
NR	Norma Regulamentadora
OPC	*Open Platform Communications* (Comunicações de Plataforma Aberta — originalmente *OLE for Process Control*)
OPC DA	*OPC Data Access* (Acesso a Dados OPC)
OPC UA	*OPC Unified Architecture* (Arquitetura Unificada OPC)
RAMI	*Reference Architecture Model Industrie 4.0* (Modelo de Arquitetura de Referência para a Indústria 4.0)
SAPI5	*Speech API 5* (Interface de Programação de Aplicativos de Voz da Microsoft 5)
SCADA	*Supervisory Control and Data Acquisition* (Controle Supervisório e Aquisição de Dados)

SESMT	Serviço Especializado em Engenharia de Segurança e em Medicina do Trabalho
SOA	*Service-Oriented Architecture* (Arquitetura Orientada a Serviços)
TTS	*Text-to-Speech* (Texto para Fala / Síntese de Voz)
URI	*Uniform Resource Identifier* (Identificador Uniforme de Recurso)

Sumário

1 Introdução

A automação industrial é peça-chave na dinâmica produtiva moderna. Enquanto a manufatura clássica dependia predominantemente de esforços manuais e repetitivos, a introdução de tecnologias automatizadas otimizou os processos fabris e elevou a eficiência das plantas industriais(SILVEIRA; LIMA, 2014).

Na definição de Silveira e Lima (2014), a automação constitui um conjunto de técnicas para executar tarefas operacionais de forma automática, substituindo o trabalho humano — tanto o esforço muscular quanto o mental — por elementos eletromecânicos computáveis. Essa transição tecnológica traz vantagens diretas à indústria, como o aumento da eficiência, a redução de custos operacionais e a melhoria dos índices de segurança física.

Entre os componentes fundamentais para a operação de uma planta industrial, destacam-se os Controladores Lógicos Programáveis (CLPs), as Interfaces Homem-Máquina (IHMs) e os sistemas *Supervisory Control and Data Acquisition* (SCADA). Enquanto os CLPs executam o controle lógico e sequencial das malhas, as IHMs estabelecem a interface visual entre o operador e a planta. Por fim, os sistemas SCADA integram os dispositivos de campo, viabilizando a aquisição de dados e a supervisão em tempo real.

O ambiente SCADA integra *hardware*, *software* e infraestrutura de rede para centralizar o monitoramento e oferecer suporte analítico às decisões operacionais. Contudo, concentrar um grande volume de dados em telas operacionais gera desafios ergonômicos complexos. Nesse cenário, a Ergonomia Cognitiva avalia os processos mentais exigidos do operador — como percepção, atenção e memória de curto prazo —, buscando mitigar o risco de falhas humanas e acidentes.

No Brasil, a Norma Regulamentadora NR-17 estabelece as diretrizes para adaptar as condições de trabalho às características psicofisiológicas dos trabalhadores, utilizando a Análise Ergonômica do Trabalho (AET) como ferramenta de diagnóstico com o suporte do Serviço Especializado em Engenharia de Segurança e em Medicina do Trabalho (SESMT). Assim, desenvolver sistemas de notificação de alarmes por voz — que atuam como redundância sensorial aos alarmes visuais convencionais — é uma alternativa eficaz para reduzir o tempo de resposta e aliviar a carga cognitiva nas salas de controle.

1.1 Justificativas e Relevância

A complexidade dos processos industriais modernos exige que os operadores analisem grandes volumes de dados textuais e numéricos sob forte pressão temporal. A sobrecarga de estímulos visuais somada a alertas sonoros genéricos, como sirenes e *bips* repetitivos,

provoca a chamada fadiga de alarmes. Esse fenômeno aumenta o risco de erros operacionais e de negligência diante de falhas críticas.

Este trabalho justifica-se pela proposta de desenvolver uma ferramenta de voz para auxílio no monitoramento de alarmes industriais. A solução foi fundamentada em princípios da Ergonomia Cognitiva e nas diretrizes da NR-17, buscando organizar e priorizar as informações de alarme como recurso complementar ao monitoramento visual realizado no supervisório.

A relevância do estudo está na integração de diferentes plataformas alinhadas à Indústria 4.0. Ao conectar o *software* comercial Elipse E3 a um algoritmo em *Python* com síntese de voz (*Text-to-Speech* - TTS) *offline*, a pesquisa comprova a viabilidade de construir interfaces industriais intuitivas, robustas e seguras. Embora o estudo de caso foque no controle de nível de uma caldeira industrial — devido ao alto risco de acidentes e às exigências da NR-13 —, o sistema monitora de forma integrada todas as variáveis essenciais (temperatura, nível, vazão e status do atuador),permitindo a simulação integrada de condições de alarme associadas às principais variáveis do processo.

A escolha detalhada desse equipamento como objeto de estudo justifica-se pelo seu alto risco operacional em sistemas de utilidades térmicas. Por trabalhar sob pressões e temperaturas elevadas, qualquer desvio ou atraso no diagnóstico pode causar acidentes graves ou danos estruturais severos à planta. Esse ambiente de alta criticidade torna a caldeira o cenário perfeito para avaliar a aplicação prática das diretrizes da NR-17, provando que o uso combinado de avisos visuais e auditivos funciona como uma medida preventiva eficiente para poupar a atenção do operador e evitar falhas humanas catastróficas.

1.1.1 Objetivos Específicos

Para alcançar o escopo geral proposto, esta pesquisa cumpriu as seguintes etapas:

- Revisar e fundamentar os conceitos de Ergonomia Cognitiva, fadiga de alarmes e as diretrizes normativas para o design de IHMs em sistemas de supervisão;
- Projetar uma arquitetura de comunicação industrial robusta, utilizando o protocolo OPC UA, para realizar a captura das tags de processo e eventos em tempo real;
- Desenvolver o módulo de assistência verbal em Python por meio de ferramentas de síntese de voz *offline*, garantindo inteligibilidade e baixa latência computacional;
- Validar o sistema integrado através de ensaios práticos de falhas concorrentes na planta térmica, avaliando o comportamento da fila de prioridades síncronas.

1.2 Metodologia

Para atingir os objetivos propostos, este trabalho adotou uma abordagem de pesquisa aplicada, dividida em três fases lógicas e sequenciais.

Como apoio à elaboração do documento, utilizou-se uma ferramenta de Inteligência Artificial Generativa exclusivamente para auxiliar na estruturação de códigos em *LaTeX*, incluindo a inserção, o posicionamento e os ajustes de imagens e tabelas nas páginas. A ferramenta também auxiliou na organização da formatação textual. Não foi utilizada para gerar dados experimentais, resultados, análises técnicas ou conclusões, permanecendo a seleção dos conteúdos, a revisão e a validação final sob responsabilidade da autora.

1.2.1 Fase 1: Referencial Teórico e Normativo

Esta etapa compreendeu o levantamento bibliográfico e documental sobre sistemas SCADA, protocolo OPC UA, Ergonomia Cognitiva, fadiga de alarmes e gestão de alarmes industriais. O foco consistiu em identificar fundamentos teóricos e normativos para orientar a organização, a priorização e a apresentação das informações de alarme no ambiente supervisório.

1.2.2 Fase 2: Desenvolvimento do Ambiente Virtual

Esta etapa compreendeu a modelagem da interface gráfica e da lógica de simulação da planta de utilidades no *software* Elipse E3. O ambiente foi configurado para representar, de forma simplificada, as variações de nível, temperatura, vazão e estado da bomba em uma caldeira industrial, permitindo a geração de diferentes condições de alarme no ambiente simulado.

1.2.3 Fase 3: Integração Multiplataforma e Validação Técnica

Esta etapa contemplou o desenvolvimento do algoritmo de assistência por voz em *Python* e a configuração da comunicação via protocolo OPC UA. A validação foi realizada por meio de testes no ambiente simulado, verificando a comunicação entre os sistemas, a leitura das variáveis, a identificação das condições de alarme, o funcionamento da fila de prioridades e a emissão das mensagens de voz.

1.3 Organização e Estrutura do Trabalho

Para apresentar a divisão cronológica e lógica das etapas desta pesquisa, o presente relatório de engenharia encontra-se estruturado em cinco capítulos principais:

- **Capítulo 1 – Introdução:** Apresenta a contextualização do tema, as justificativas de engenharia, os objetivos geral e específicos, além da abordagem metodológica adotada;
- **Capítulo 2 – Revisão de Literatura:** Fundamenta a base teórica essencial do projeto, abrangendo a evolução histórica e a arquitetura funcional dos sistemas SCADA, a infraestrutura de comunicação do protocolo OPC UA e os conceitos de Ergonomia Cognitiva voltados à redução da fadiga de alarmes;
- **Capítulo 3 – Desenvolvimento:** Detalha o processo de engenharia aplicada na construção do simulador da caldeira no Elipse E3, o mapeamento prático de tags e *NodeId*, a modelagem do algoritmo de assistência verbal em Python e a implementação da camada de segurança operacional;
- **Capítulo 4 – Resultados e Discussões:** Apresenta os resultados obtidos por meio dos testes de estresse e das simulações de falhas na planta térmica, validando a hierarquia de alarmes fundamentada na norma ISA 18.2 e a capacidade do sistema de manter uma comunicação eficiente e com tempo de resposta adequado;
- **Capítulo 5 – Considerações Finais:** Sintetiza as principais conclusões obtidas ao longo do estudo, discute as limitações técnicas superadas por meio da aplicação de técnicas de engenharia de *software* e propõe diretrizes para futuras investigações e aprimoramentos da solução desenvolvida.

2 Revisão de Literatura

A base teórica deste trabalho divide-se em três eixos principais: a evolução dos sistemas de supervisão e aquisição de dados (SCADA); a infraestrutura de comunicação industrial baseada no protocolo OPC; e as diretrizes de ergonomia cognitiva aplicadas à segurança do operador.

2.1 Sistemas Supervisórios: do Monitoramento à Estratégia

A automação industrial passou por profundas transformações operacionais nas últimas décadas. No passado, o objetivo principal das indústrias era substituir o trabalho físico repetitivo por mecanismos automáticos. Atualmente, o foco estratégico está voltado ao gerenciamento analítico da informação e à inteligência aplicada à operação. Como aponta Pérez-López (2015), a automação atual tornou-se um requisito essencial para a sustentabilidade mercadológica das empresas, pois padroniza a qualidade, reduz custos de manufatura e, principalmente, eleva os índices de segurança ao afastar os trabalhadores de áreas de risco físico.

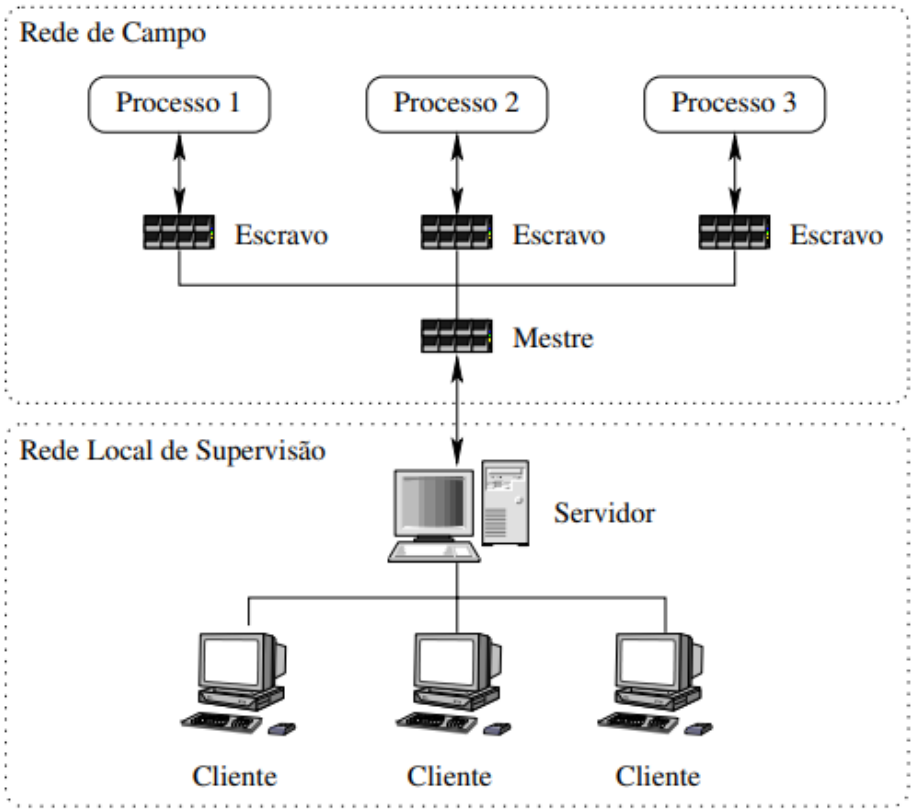
Souza (2005) define a arquitetura SCADA como uma estrutura integrada por dispositivos de campo, controladores lógicos programáveis (CLPs) e redes de comunicação dedicadas. Essa estrutura permite a aquisição contínua de variáveis de processo, o que viabiliza aos operadores monitorar e atuar em malhas de controle geograficamente dispersas em tempo real, superando as antigas limitações dos painéis sinópticos analógicos. Os sistemas SCADA (*Supervisory Control and Data Acquisition*) funcionam como o núcleo central de convergência de dados da planta.

Por outro lado, a concentração de uma grande quantidade de dados em uma única tela trouxe novos desafios para a engenharia de fatores humanos. Souza (2005) ressalta que o volume de informações trafegadas nas redes de supervisão pode crescer de forma exponencial. Diante disso, a Interface Homem-Máquina (IHM) torna-se o elemento mais crítico do sistema de controle. Segundo Pérez-López (2015), uma interface gráfica mal projetada, que apenas exibe dados brutos sem categorização ou prioridade, prejudica a atenção seletiva do operador. Sob essas condições, o profissional deixa de exercer uma supervisão analítica e passa a agir de forma puramente reativa — silenciando alarmes de emergência de maneira mecânica devido à saturação de estímulos visuais e sonoros, sem compreender a causa raiz da anomalia.

2.1.1 Arquitetura Funcional e o Fluxo da Informação

Como aponta Souza (2005), a eficiência de um sistema de supervisão depende de elementos estruturais básicos que garantam a integridade e o determinismo temporal dos dados transmitidos à tela, conforme ilustrado na Figura 1.

Figura 1 – Estrutura física e de funcionamento de um sistema supervisório



Fonte: Adaptado de Souza (2005).

O *software* de supervisão coleta, trata e contextualiza as variáveis operacionais. No escopo deste trabalho, essa infraestrutura fundamenta o desenvolvimento do módulo de assistência verbal, projetado como uma camada complementar de redundância sensorial. O algoritmo intercepta as variáveis (*tags*) de alarme ativas no servidor e as converte em mensagens sintetizadas por voz, garantindo a inteligibilidade necessária para acelerar o diagnóstico do operador.

2.1.2 Hierarquia de Automação e o Posicionamento do SCADA

Para compreender o papel dos sistemas supervisórios, é necessário analisar sua posição dentro da Pirâmide de Automação Industrial (Figura 2). Essa estrutura organiza

os diferentes níveis de controle e decisão de uma planta, mapeando o fluxo de dados desde o chão de fábrica até o planejamento estratégico corporativo (SOUZA, 2005).

Figura 2 – Níveis da Pirâmide de Automação Industrial

Fonte: Adaptado de Souza (2005), com seta indicativa acrescentada pela autora.

A pirâmide é segmentada em cinco níveis operacionais que delimitam a subida vertical das informações dentro do ecossistema fabril:

- **Nível 1 (Chão de Fábrica):** Composto por dispositivos físicos, atuadores e sensores de campo. No escopo experimental deste projeto, este nível abriga os instrumentos que medem continuamente a temperatura do vaso de pressão, o nível hidráulico do tanque e a vazão de fluido nas tubulações da caldeira;
- **Nível 2 (Controle):** Constituído pelos Controladores Lógicos Programáveis (CLPs), que coletam os sinais elétricos dos sensores do Nível 1 e executam as rotinas de controle e intertravamento em tempo real;
- **Nível 3 (Supervisão):** Camada onde se localiza o sistema SCADA, foco central desta pesquisa. Este nível realiza a aquisição dos dados gerenciados pelo controle (Nível 2) e os exibe graficamente, funcionando como uma ponte de comunicação com as camadas superiores;
- **Nível 4 (Gerenciamento):** Compreende os *softwares* dedicados ao planejamento e execução da produção, controle de estoque e balanço de massa, como os sistemas MES (*Manufacturing Execution System*);

- **Nível 5 (Corporativo):** Topo da estrutura hierárquica, onde os sistemas de gestão integrada ERP (*Enterprise Resource Planning*) centralizam as decisões estratégicas e de negócios com base no histórico produtivo da planta.

2.2 Interoperabilidade e Integração via Protocolo OPC

Para aumentar a eficiência operacional das interfaces convencionais, é necessário conectar o sistema SCADA a aplicações externas especializadas, como o *script* em Python desenvolvido neste projeto para a síntese de voz de eventos críticos. No passado, a integração de plataformas de diferentes fabricantes era um desafio complexo de engenharia de redes.

Fumagalli, Branco e Dias (2020) explicam que, no início da automação industrial, a comunicação de dados dependia de *drivers* proprietários e fechados de cada fabricante de *hardware*. Como cada arquitetura de CLP utilizava protocolos distintos e incompatíveis, as indústrias lidavam com sistemas isolados. Esse cenário obrigava os desenvolvedores a programar códigos customizados para cada equipamento de campo, o que elevava os custos de implantação e dificultava a expansão das plantas.

Para superar essa barreira de conectividade, desenvolveu-se o padrão OPC (*Open Platform Communications*). Segundo Leitner e Mahnke (2006), o OPC funciona como uma camada de abstração de *software* universal: o sistema supervisório (cliente) não precisa conhecer detalhadamente os registradores internos do CLP (servidor). O cliente apenas requisita as informações em um formato padronizado fornecido pela interface OPC.

2.2.1 A Evolução: do OPC DA ao OPC UA

A evolução desse protocolo industrial divide-se em duas fases principais:

- **OPC DA (*Data Access*):** Especificação clássica desenvolvida para a leitura e escrita de variáveis de processo em tempo real. Esta versão serviu de base para a captura inicial das variáveis operacionais (*tags*) monitoradas neste trabalho (LEITNER; MAHNKE, 2006);

- **OPC UA (*Unified Architecture*):** Versão moderna do protocolo e um dos pilares de conectividade da Indústria 4.0 (FUMAGALLI; BRANCO; DIAS, 2020). O padrão opera de maneira independente do sistema operacional, permitindo a execução nativa em ambientes Windows, Linux ou sistemas embarcados. Além disso, incorpora recursos robustos de segurança e criptografia. Leitner e Mahnke (2006) ressaltam que essa arquitetura eliminou as complexidades de configuração de rede associadas à antiga tecnologia *Distributed Component Object Mode*(DCOM) da Microsoft, reduzindo as falhas de comunicação entre os níveis de controle.

O uso do protocolo OPC neste trabalho garante o caráter universal do assistente em Python. Assim, o módulo de voz pode ser conectado de forma agnóstica a qualquer *software* supervisório comercial que adote essa tecnologia, como o Elipse E3, *ScadaBR* ou *WinCC*.

2.2.1.1 Fundamentação e Características Estruturais do OPC UA

O padrão OPC UA foi projetado com características funcionais que superam as limitações operacionais do modelo clássico DA (HIRSCH; HOHER; HUBER, 2023). Desenvolvido sob o conceito de *Service-Oriented Architecture* (SOA, Arquitetura Orientada a Serviços), o protocolo oferece uma estrutura de comunicação escalável e independente de plataforma (OPC FOUNDATION, 2022). Isso elimina a dependência de sistemas operacionais específicos — comum no OPC DA devido ao uso da tecnologia DCOM da Microsoft —, permitindo a troca nativa de dados entre plataformas heterogêneas, como Windows, Linux e sistemas embarcados em controladores de campo (FUMAGALLI; BRANCO; DIAS, 2020).

O OPC UA também traz uma arquitetura nativa de segurança da informação, essencial para mitigar riscos na convergência entre as redes de Tecnologia da Informação (TI) e Tecnologia Operacional (TO) (HIRSCH; HOHER; HUBER, 2023). O protocolo incorpora mecanismos de segurança, incluindo criptografia baseada no padrão *Advanced Encryption Standard* (AES, Padrão de Criptografia Avançado), autenticação de usuários e certificados digitais (OPC FOUNDATION, 2022). Esses mecanismos garantem a confidencialidade, autenticidade e integridade dos dados contra ataques cibernéticos, protegendo as redes industriais sem interromper o fluxo de dados (FUMAGALLI; BRANCO; DIAS, 2020).

Devido a essas vantagens, o OPC UA tornou-se um padrão recomendado pelo Modelo de Arquitetura de Referência da Indústria 4.0, denominado *Reference Architecture Model Industrie 4.0* (RAMI 4.0). O protocolo desacopla os geradores de dados no chão de fábrica, como CLPs e sensores, dos seus consumidores, como interfaces SCADA, bancos de dados e o assistente de voz (HIRSCH; HOHER; HUBER, 2023). Esse arranjo melhora o uso da largura de banda da rede e reduz a carga de processamento dos equipamentos de campo, pois a coleta de dados passa a ser centralizada (SOUZA, 2005).

2.2.1.2 O Espaço de Endereçamento *Address Space* e a Organização de Nós

O diferencial técnico do OPC UA está na modelagem de informações por meio do *Address Space* (Espaço de Endereçamento). Enquanto o OPC DA gerenciava apenas registros planos e isolados de memória, a arquitetura atual organiza os dados em um modelo hierárquico (grafo), no qual as variáveis de processo são contextualizadas em tempo real com seus metadados (HIRSCH; HOHER; HUBER, 2023).

Segundo as diretrizes da OPC Foundation (2022), o *Address Space* é constituído por *Nodes* (Nós), que representam objetos físicos ou lógicos, como sensores, atuadores ou blocos

de controle. A modelagem de cada nó baseia-se em duas propriedades complementares (HIRSCH; HOHER; HUBER, 2023):

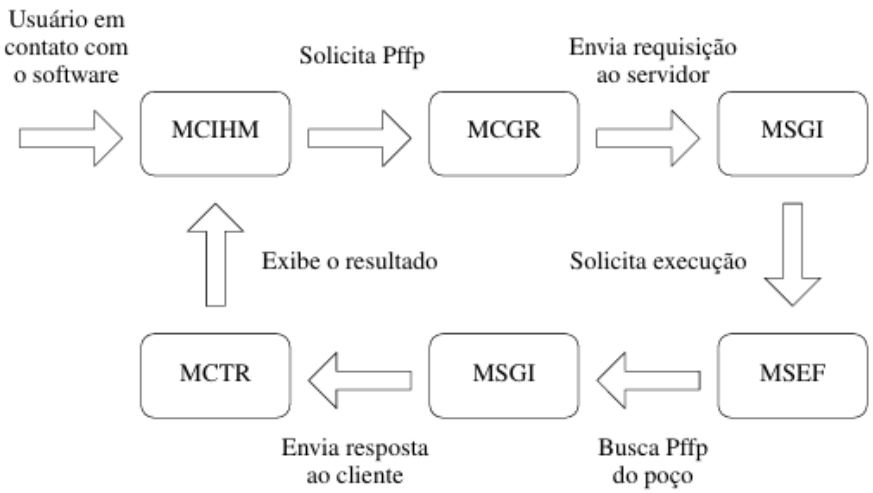
- ***Attributes* (Atributos):** Metadados fixos que descrevem as características estáticas do nó, como o *Value* (valor dinâmico medido), *DataType* (tipo de dado, como Float ou Int32) e *AccessLevel* (permissões de leitura e escrita);
- ***References* (Referências):** Vínculos que estabelecem as relações hierárquicas e lógicas entre os nós. Elas permitem que o sistema compreenda o contexto físico da variável, associando, por exemplo, qual indicador de temperatura ou status de bomba pertence a determinado subsistema da caldeira (SOUZA, 2005).

Para organizar as diferentes *tags* e evitar conflitos, utiliza-se o conceito de *Namespace* (Espaço de Nomes). O *Namespace* funciona como um repositório indexado por um número único (`ns`), vinculado à *Uniform Resource Identifier* (URI) do fabricante ou da simulação. Isso impede a colisão de nomes quando múltiplos dispositivos compartilham a mesma rede. Assim, a localização de qualquer variável exige um identificador estruturado chamado *NodeId*, composto pelo índice do *Namespace* (`ns`), o tipo do nó — numérico (`i`) ou textual (`s`) — e seu valor exclusivo. Essa padronização permite realizar consultas diretas, reduzindo a complexidade de varredura e fornecendo a base lógica para o *script* em Python ler os estados da planta.

2.2.2 O Papel do OPC na Integração de Equipamentos

No passado, a incompatibilidade entre *softwares* e *hardwares* gerava custos extras e retrabalho nos projetos de automação (SOUZA, 2005). Como mostra a Figura 3, o padrão OPC permite interligar plataformas diferentes dividindo o tráfego de dados em blocos específicos de software. Para compreender o funcionamento desse fluxo, faz-se necessário detalhar o significado de cada módulo presente na estrutura:

Figura 3 – Integração de vários dispositivos e clientes usando o padrão OPC



Fonte: Adaptado de Souza (2005).

- **MCIHM (Módulo Cliente de Interface Homem-Máquina):** camada visual que realiza a interação direta com o usuário por meio da tela do computador;
- **MCGR (Módulo Cliente de Gerenciamento de Rede):** bloco responsável por organizar o tráfego de dados e as requisições do sistema;
- **MSGI (Módulo Servidor de Gerenciamento de Informações):** servidor responsável por processar as mensagens e buscar as informações do processo;
- **MCTR (Módulo de Controle em Tempo Real):** unidade responsável pela execução das lógicas de controle e pela disponibilização dos resultados do processo;
- **MSEF (Módulo Servidor de Equipamentos Físicos):** módulo responsável pela conexão entre o sistema supervisório e os equipamentos físicos de campo, como sensores e atuadores.

Essa flexibilidade viabiliza o funcionamento do módulo de assistência por voz desenvolvido nesta pesquisa. O servidor OPC gerencia o tráfego na rede de controle e distribui as variáveis simultaneamente para a interface do SCADA e para o interpretador Python, garantindo a integridade e o sincronismo dos dados (SOUZA, 2005).

2.3 Ergonomia Cognitiva e Fatores Humanos

Enquanto a engenharia de redes soluciona a comunicação entre os ativos de *hardware*, a Ergonomia Cognitiva concentra-se na interface entre o sistema digital e o operador

humano. Com o aumento da complexidade e da automação nas plantas industriais modernas, a demanda física sobre o operador diminuiu, mas o esforço cognitivo exigido para supervisionar os processos cresceu de forma expressiva. Como consequência, a origem dos riscos operacionais e acidentes migrou das falhas estritamente mecânicas para os erros humanos induzidos pela sobrecarga de informações nas salas de controle.

Na análise de Araújo e Fátima Ferraz da Silva Tacconi (2023), a organização do trabalho atual impõe altas demandas aos processos mentais, como percepção, atenção seletiva, memória de curto prazo, raciocínio lógico e tomada de decisão sob pressão de tempo. Os autores definem a "Carga Mental de Trabalho" como a relação entre as demandas cognitivas da tarefa e a capacidade do indivíduo para processá-las. Quando as exigências do ambiente superam os limites biológicos e psicológicos do operador, surgem falhas operacionais, lapsos de memória e fadiga mental.

A Ergonomia Cognitiva estuda justamente esses mecanismos de processamento de informações no ambiente de trabalho. Seu principal objetivo é desenvolver ferramentas e interfaces gráficas que otimizem o fluxo de dados, reduzindo a necessidade de memorizar comandos complexos ou interpretar dados ambíguos em momentos de crise (ARAÚJO; FÁTIMA FERRAZ DA SILVA TACCONI, 2023). Dessa forma, busca-se projetar um sistema de supervisão compatível com as características do pensamento humano, garantindo a integridade do trabalhador e a segurança da planta.

2.3.1 Ergonomia Cognitiva em Ambientes de Alta Complexidade

A aplicação prática dos conceitos ergonômicos é indispensável em ambientes industriais de alta complexidade e criticidade, como o setor mineral, indústrias de processo contínuo e plantas térmicas. Conforme apontado por Arruda *et al.* (2006), embora a automação nessas infraestruturas tenha reduzido os acidentes por esforço físico, ela concentrou os riscos na atividade mental de supervisão. Em salas de controle de mineração ou de cogeração de vapor, o operador monitora telas saturadas de variáveis interconectadas, sob o risco de que uma falha em um único ativo provoque um efeito em cascata em todo o processo.

Para avaliar o impacto desses fatores na rotina de trabalho, Silva e Girão (2015) conduziram um estudo de caso sobre a ergonomia de um posto em uma sala de controle operacional. Os autores evidenciaram que a falta de planejamento ergonômico nas tarefas e no arranjo espacial das ferramentas resulta em quadros crônicos de fadiga, distúrbios osteomusculares e estresse. Esses problemas afetam diretamente operadores submetidos a longas jornadas de monitoramento de telas de vídeo sem sinalizações visuais ou sonoras bem estruturadas. A pesquisa confirma que falhas no leiaute e a ausência de mecanismos que ajudem a filtrar os dados provocam a sobrecarga cognitiva do operador, prejudicando tanto o seu desempenho técnico quanto a sua saúde a longo prazo.

No monitoramento de uma caldeira industrial, o operador lida constantemente com esses elevados riscos físicos e mentais. Sob essa perspectiva, Arruda *et al.* (2006) ressaltam que o erro humano em sistemas complexos raramente deve ser atribuído apenas à conduta do operador. Na verdade, ele costuma ser o resultado direto de interfaces mal projetadas que ignoram os limites humanos de processamento de dados.

O estudo de Silva e Girão (2015) reforça a importância de adaptar os postos de supervisão às características psicofisiológicas dos trabalhadores, mostrando que melhorias na sinalização e na organização visual dos dados reduzem o estresse ocupacional. Portanto, o uso de sistemas assistivos que convertem dados críticos em mensagens de voz sintetizada é uma alternativa de engenharia eficiente para atenuar a sobrecarga provocada pela atenção visual contínua, oferecendo ao operador o suporte necessário para tomadas de decisão rápidas e para a prevenção de acidentes graves.

2.3.2 O Fenômeno da Fadiga de Alarmes e a Norma ISA 18.2

Uma das manifestações mais graves da sobrecarga mental em ambientes de supervisão centralizada é a "Fadiga de Alarmes". Essa condição ocorre quando o operador é bombardeado por um volume excessivo de alertas visuais e sonoros simultâneos, dos quais uma parte significativa não exige nenhuma intervenção operacional ou representa apenas a repetição de uma mesma falha do processo.

De acordo com Araújo e Fátima Ferraz da Silva Tacconi (2023), a exposição contínua a estímulos sonoros invariáveis, como *bips* e sirenes repetitivas, ativa um mecanismo neuropsicológico de defesa chamado dessensibilização. Para conseguir manter o foco em uma atividade primária, a estrutura cognitiva do operador passa a tratar as sinalizações de advertência como ruído de fundo, ignorando sua ocorrência. O perigo dessa resposta comportamental é que o operador pode deixar de perceber um alarme de emergência real e grave, o que pode levar a falhas catastróficas, como explosões em vasos de pressão, paradas inesperadas na produção ou desastres ambientais.

Para reduzir esse problema, a norma internacional **ANSI/ISA 18.2** (*Management of Alarm Systems for the Process Industries*), desenvolvida pelo *American National Standards Institute* (ANSI) e pela *International Society of Automation* (ISA), estabelece diretrizes para o ciclo de vida e gerenciamento de alarmes (INTERNATIONAL SOCIETY OF AUTOMATION, 2016). Essa regulamentação determina que um alarme legítimo deve ser gerado apenas se indicar um desvio real do processo e se exigir uma ação corretiva imediata do operador. Os princípios de priorização, supressão e filtragem definidos pela norma ajudam a organizar a interface de monitoramento, garantindo que a camada assistiva de voz funcione estritamente como um elemento de redundância informativa, sem se tornar mais um fator de distração sensorial.

2.3.3 Teoria dos Recursos Múltiplos e a Eficácia da Voz

A justificativa científica para o uso de avisos por voz na redução da fadiga mental baseia-se na Teoria dos Recursos Múltiplos, formulada por Wickens (2008). Essa abordagem demonstra que a estrutura cognitiva humana não possui um único canal atencional centralizado; pelo contrário, ela dispõe de *pools* independentes de processamento para diferentes modalidades sensoriais, separando o canal visual-espacial do canal auditivo-verbal.

Wickens (2008) aponta que executar duas tarefas ao mesmo tempo usando o mesmo canal atencional — como monitorar visualmente o nível hidráulico e ler simultaneamente um texto descritivo de alarme na tela — gera interferência e prejudica significativamente a eficiência do usuário. Na arquitetura desenvolvida neste projeto, as sinalizações que antes ficavam restritas ao formato textual foram convertidas em alertas falados por meio de síntese de voz.

Dessa forma, o sistema direciona a informação crítica para a modalidade auditivo-verbal, deixando o canal visual-espacial livre para que o operador mantenha a atenção focada na dinâmica física das variáveis da caldeira. Essa distribuição balanceada dos estímulos respeita os limites da arquitetura cognitiva humana, reduz a carga mental de trabalho e evita a perda de foco durante cenários de emergência.

2.4 Normas Regulamentadoras e Diretrizes do Projeto

O desenvolvimento de interfaces industriais e sistemas supervisórios deve estar alinhado às exigências legais e às normas regulamentadoras de segurança ocupacional vigentes no país.

2.4.1 A Norma Regulamentadora NR-13 (Caldeiras e Vasos de Pressão)

A Norma Regulamentadora NR-13 é compulsória para plantas industriais que operam caldeiras, visto que esses equipamentos trabalham sob altas pressões e temperaturas, apresentando risco latente de explosão (PAIOLA; ROCHA; RODRIGUES, 2019). De acordo com a norma, a ausência ou a indisponibilidade de dispositivos de controle do nível de água na caldeira, bem como a sua operação sem os sistemas instrumentados de segurança regulamentares, configuram condição de grave e iminente risco (BRASIL, 2023a). O texto legal determina que todos os instrumentos e malhas de controle devem ser mantidos em perfeitas condições de uso, calibrados e inspecionados por profissionais habilitados.

Além disso, a NR-13 exige que o estabelecimento mantenha um "Registro de Segurança" atualizado, em suporte físico ou sistema informatizado seguro, para a anotação de ocorrências relevantes que possam impactar a integridade do vaso de pressão

(BRASIL, 2023a). No sistema desenvolvido nesta pesquisa, o atendimento a esse requisito legal foi garantido pela implementação de um banco de dados em Microsoft Access. O armazenamento histórico das variações de temperatura, nível, vazão, disparos de alarmes e comandos do atuador atende às exigências de rastreabilidade e auditoria da legislação federal.

2.4.2 A Norma Regulamentadora NR-17 (Ergonomia Ocupacional)

Pelo lado dos fatores humanos e da engenharia de usabilidade, a Norma Regulamentadora NR-17 define os parâmetros para adaptar as condições de trabalho às características psicofisiológicas dos operadores, buscando garantir conforto, segurança e eficiência na planta (BRASIL, 2023b). Quanto à organização do trabalho, a norma determina que as ferramentas tecnológicas e as tarefas respeitem os limites mentais do trabalhador, evitando exigências exageradas que causem desgaste. Nesse cenário, a sobrecarga de atenção e o cansaço sensorial são classificados como riscos ergonômicos, exigindo soluções práticas de engenharia.

No que diz respeito ao uso de telas e monitores, a NR-17 exige que os alarmes, ícones e comandos facilitem uma interação clara e direta, diminuindo a chance de falhas na interpretação dos dados operacionais (BRASIL, 2023b). O assistente de voz proposto neste projeto atende exatamente a essa diretriz. Ao dividir os alertas entre os canais visual e auditivo, o sistema evita que o operador precise ler textos pequenos de alarmes na tela do supervisório no meio de uma rotina pesada. Isso reduz o desgaste mental e protege a sala de controle contra erros causados pelo excesso de estímulos visuais (ARAÚJO; FÁTIMA FERRAZ DA SILVA TACCONI, 2023).

3 Desenvolvimento

Este capítulo detalha os procedimentos técnicos aplicados na construção do sistema de supervisão de uma caldeira industrial e na implementação do módulo de assistência por voz. Ambos os sistemas baseiam-se na integração entre o *software* comercial Elipse E3 e a linguagem Python utilizando o protocolo industrial OPC UA.

3.1 Caracterização da Caldeira Industrial

A caldeira industrial foi utilizada como cenário simulado para o desenvolvimento e a validação técnica da ferramenta proposta. Por operar com variáveis como nível, temperatura, vazão e estado da bomba, esse processo permite a criação de diferentes condições de alarme e a avaliação da lógica de priorização implementada.

A escolha desse cenário também se relaciona à criticidade operacional de sistemas térmicos. Em uma planta industrial real, desvios não identificados ou não tratados adequadamente podem comprometer a segurança e a integridade dos equipamentos. Nesse sentido, a Norma Regulamentadora NR-13 é utilizada como referência para contextualizar os riscos associados à operação de caldeiras e vasos de pressão.

Entretanto, o foco principal deste trabalho está nos princípios da Ergonomia Cognitiva e nas diretrizes da Norma Regulamentadora NR-17. A ferramenta de voz foi concebida como um recurso complementar ao monitoramento visual do supervisório, apresentando mensagens curtas e priorizadas para auxiliar a identificação de condições de alarme. Dessa forma, busca-se contribuir para uma apresentação mais organizada das informações em situações críticas, sem substituir a supervisão visual realizada pelo operador.

Ressalta-se que a validação foi realizada exclusivamente em ambiente simulado e não envolveu testes com operadores humanos. Portanto, o trabalho não mensura diretamente a redução da carga mental, da fadiga de alarmes ou do tempo de resposta dos usuários.

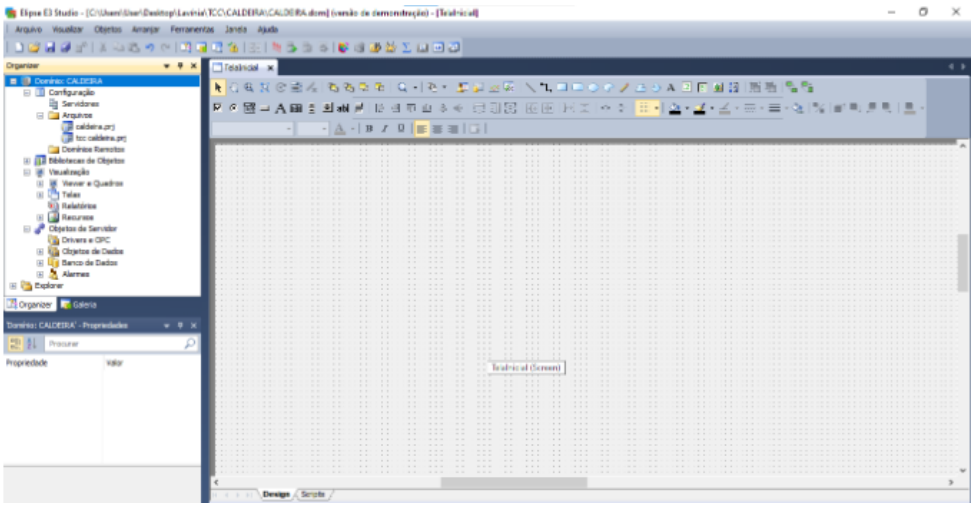
3.2 Desenvolvimento do Supervisório no Elipse E3

O desenvolvimento do supervisório baseou-se na metodologia de Fraga (2020), iniciando-se pela criação das interfaces gráficas e pela organização lógica dos componentes do sistema.

3.2.1 Tela Inicial e Navegação

A primeira etapa do projeto consistiu na estruturação da interface de acesso, denominada “Tela Inicial” (Figura 4). Essa tela funciona como ponto de entrada da aplicação, permitindo que o usuário acesse a tela operacional da caldeira, na qual são apresentadas as variáveis monitoradas, os alarmes e os comandos disponíveis no supervisório.

Figura 4 – Tela inicial no ambiente Elipse E3



Fonte: Produção própria (2026).

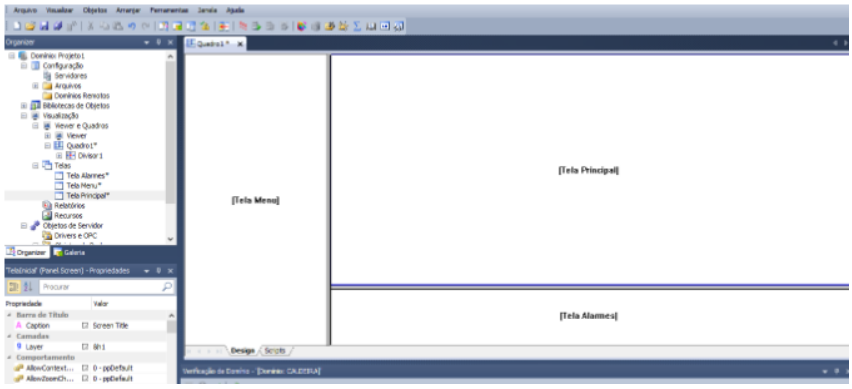
3.2.2 Divisão Estrutural das Telas

O sistema foi organizado de forma modular, dividindo as telas de simulação e controle dentro do *software*. Essa separação é necessária para organizar o fluxo de navegação na sala de controle e garantir a distinção clara entre as funções operacionais, analíticas e de segurança do supervisório. As telas principais foram divididas em:

- “Tela Principal”: voltada ao monitoramento em tempo real e à análise das variáveis de processo da caldeira (nível de água, temperatura do vaso, vazão de fluido e status da bomba);
- “Tela de Alarmes”: onde são centralizados, registrados no banco de dados e reconhecidos os eventos críticos e alertas de emergência;
- “Tela de Menu”: interface que permite a alternância rápida entre os subsistemas operacionais da planta.

A Figura 5 ilustra a arquitetura adotada para a divisão das telas, demonstrando o funcionamento integrado do sistema de supervisão.

Figura 5 – Divisão das telas do sistema supervisório



Fonte: Produção própria baseada em Fraga (2020).

3.3 Configuração da Comunicação via OPC UA

Para integrar o sistema, foi utilizado o *software Prosys OPC UA Simulation Server*. O protocolo OPC UA permite que o Elipse E3 e o script em Python acessem as variáveis do servidor ao mesmo tempo e de forma segura.

3.3.1 Mapeamento de *Tags* e *NodeIds*

No servidor OPC UA, cada variável possui um identificador único denominado *NodeId*, composto por um espaço de nomes (*Namespace*, `ns`) e por um identificador numérico (*Numeric*) ou textual (*String*) (OPC FOUNDATION, 2022). Para representar, de forma simplificada, as variações das variáveis da caldeira no ambiente simulado, foram configuradas as seguintes *tags* de processo:

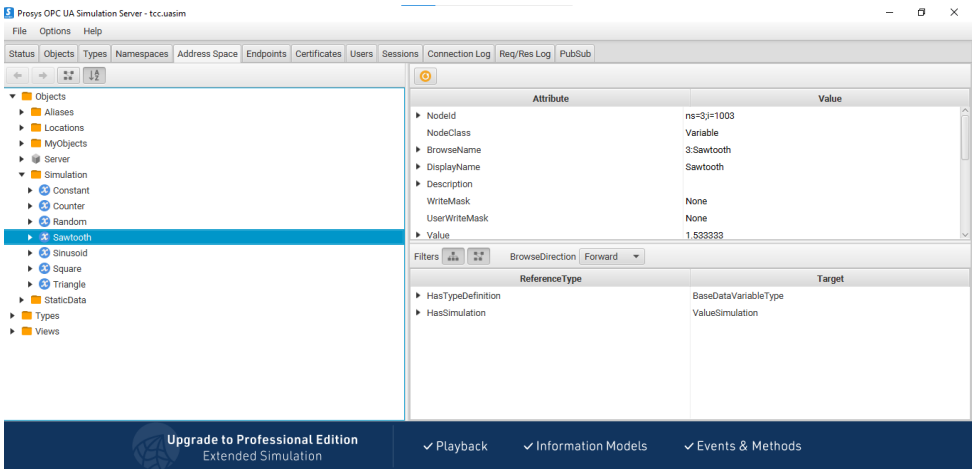
- **Temperatura** ($ns = 3; i = 1003$): sinal analógico em dente de serra (*Sawtooth*), utilizado para gerar uma variação cíclica com elevação gradual da temperatura ao longo da simulação;
- **Nível** ($ns = 3; i = 1004$): variável analógica utilizada como base para a lógica de alarmes, devido à sua relevância para a operação segura de caldeiras;
- **Vazão** ($ns = 3; i = 1002$): sinal analógico senoidal (*Sinusoid*), utilizado para representar uma variação periódica da vazão, alternando entre valores baixos e altos no ambiente simulado;
- **Bomba** ($ns = 5; s =$ Int32DataItem): *tag* de memória estática utilizada para representar os estados discretos de ligada e desligada da bomba.

Os identificadores foram estruturados conforme os espaços de nomes do servidor. As variáveis de processo — temperatura, nível e vazão — foram centralizadas no *namespace* `ns=3`, utilizando identificadores numéricos no formato `i`, padrão do simulador. Já o acionamento da bomba foi alocado no *namespace* `ns=5`, com identificador textual `s="Int32DataItem"`, utilizado como um bloco de memória para representar os comandos de ligar e desligar.

No script em Python, a varredura acessa o atributo *Value* de cada nó para obter o valor instantâneo das variáveis. A aplicação também considera o tipo de dado associado a cada variável: ponto flutuante (*Float*) para as medições analógicas e inteiro (*Int32*) para o estado da bomba. Essa distinção permite que a lógica de alarmes interprete adequadamente cada informação recebida.

A Figura 6 apresenta a configuração do espaço de endereçamento no servidor *Prosys* OPC UA, incluindo os *namespaces*, os identificadores e os tipos de dados das variáveis utilizadas na simulação.

Figura 6 – Configuração do espaço de endereçamento no Servidor *Prosys* OPC UA



Fonte: Produção própria (2026).

Durante a configuração das variáveis no servidor *Prosys*, verificou-se que o simulador disponibiliza os sinais analógicos em uma faixa normalizada entre -2 e 2. Para que esses valores brutos fossem interpretados corretamente pelo Elipse E3 e pela aplicação em Python, foi necessário associar essa faixa à escala de engenharia definida para cada variável.

Assim, o valor -2 do simulador foi associado ao limite inferior da escala configurada e o valor 2 ao limite superior. Por exemplo, na variável de temperatura, o valor -2 corresponde a 25 °C, servindo como base para o cálculo do escalonamento linear.

3.4 Normalização de Sinais e Escalonamento Linear

A etapa seguinte foi tratar os dados brutos (*raw data*) vindos do servidor industrial. Como o simulador fornece as variáveis na faixa entre -2 e 2, esses valores precisam ser convertidos. Para que o operador e o script de voz trabalhem com unidades de engenharia (°C, %, m^3/h), aplicou-se a técnica de **Escalonamento Linear**.

Essa conversão é necessária porque os sensores simulados têm resposta linear. Na prática industrial, os transmissores convertem uma grandeza física (como pressão ou nível) em um sinal elétrico proporcional (como a faixa de 4 a 20 mA). A equação de escalonamento funciona como a função de transferência que transforma o sinal bruto no valor real da grandeza correspondente.

3.4.1 Fundamentação da Equação de Escalonamento

A conversão matemática baseia-se em uma relação linear entre a faixa de entrada disponibilizada pelo simulador e a escala de engenharia utilizada para cada variável. Assim, o valor bruto lido no servidor OPC UA é convertido para uma unidade física compreensível no supervisório e no módulo de voz.

A equação de escalonamento adotada é dada por:

$$Valor_{final} = \left(\frac{Sinal_{bruto} - Início_{sinal}}{Fim_{sinal} - Início_{sinal}} \right) \times (Fim_{escala} - Início_{escala}) + Início_{escala} \quad (3.1)$$

Onde:

- **Normalização:** a primeira fração da equação subtrai o ponto inicial do servidor e divide pela amplitude total do sinal de entrada, tratando o dado bruto de forma proporcional;
- **Ganho:** o fator $(Fim_{escala} - Início_{escala})$ define a faixa real da variável. Na malha de temperatura, por exemplo, a escala de 25 °C a 250 °C possui amplitude de 225 °C;
- **Offset de Engenharia:** o parâmetro $+Início_{escala}$ soma o valor mínimo do sensor, como 25 °C, para que a escala não comece em zero.

A equação calcula a posição proporcional do sinal bruto dentro da faixa disponibilizada pelo servidor e associa esse resultado à escala de engenharia da variável monitorada.

Por exemplo, na variável temperatura, a faixa de entrada do simulador varia de -2 a 2, enquanto a escala de engenharia foi definida entre 25 °C e 250 °C. Dessa forma, o escalonamento permite associar corretamente os valores brutos às grandezas físicas exibidas no Elipse E3 e utilizadas pela lógica de alarmes no Python.

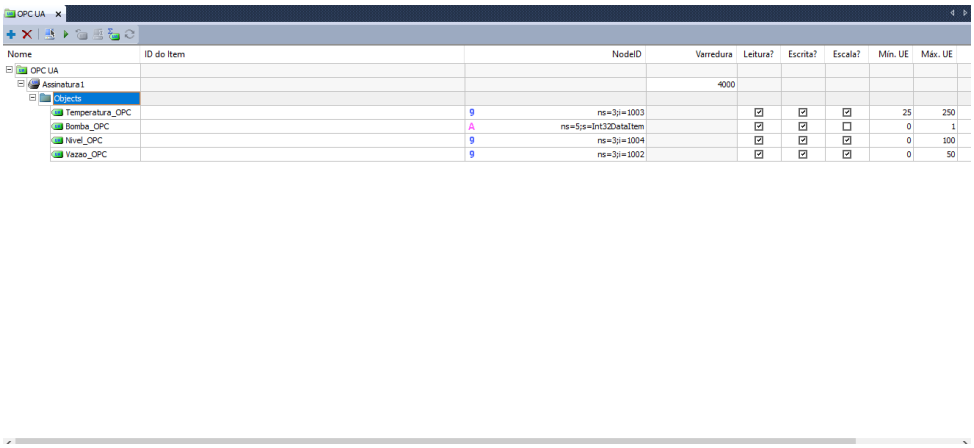
A Figura 7 apresenta a conexão estabelecida entre o Elipse E3 e o servidor OPC UA. Essa conexão permitiu acessar as variáveis disponibilizadas pelo servidor e importá-las para o supervisório, conforme mostrado na Figura 8.

Sem a aplicação da Equação 3.1, os alarmes e as mensagens de voz seriam baseados apenas nos valores brutos do simulador, que não correspondem diretamente às unidades de engenharia utilizadas na operação da caldeira.

Figura 7 – Conexão do Elipse E3 ao servidor OPC UA

Fonte: Produção própria (2026).

Figura 8 – Importação das *tags* mapeadas

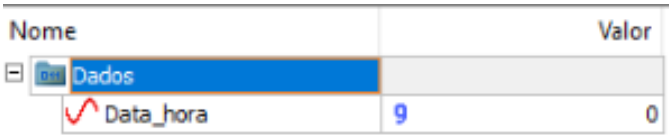


Fonte: Produção própria (2026).

3.4.2 Implementação da *Tag* de Rastreabilidade (Data e Hora)

Além das variáveis de processo, configurou-se uma *tag* interna de **Data e Hora** no Elipse E3. Como mostra a Figura 9, esse dado é processado localmente pelo supervisório para gerar o carimbo de tempo (*timestamping*). Isso permite registrar o horário exato de todas as oscilações de nível, variações de temperatura e disparos de alarmes, atendendo às exigências de fiscalização e auditoria.

Figura 9 – Configuração da *tag* interna de Data e Hora no *Organizer*



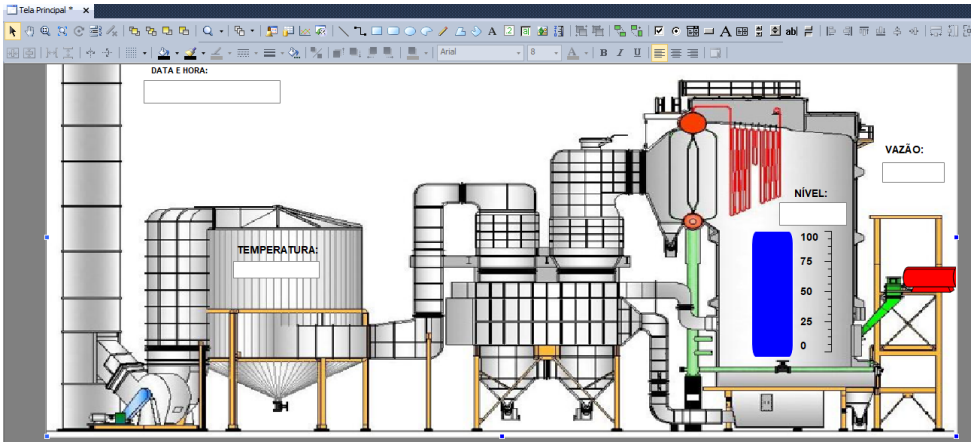
Fonte: Produção própria (2026).

3.4.3 Associações de Dados e Lógica de Comando

Após a criação das telas e a configuração das *tags*, realizou-se a etapa de associações, na qual as propriedades visuais dos componentes gráficos da "Tela Principal" (Figura 10) foram vinculadas às suas respectivas *tags*. Esse procedimento garante que qualquer mudança de estado no servidor OPC UA atualize instantaneamente os valores e indicadores da tela.

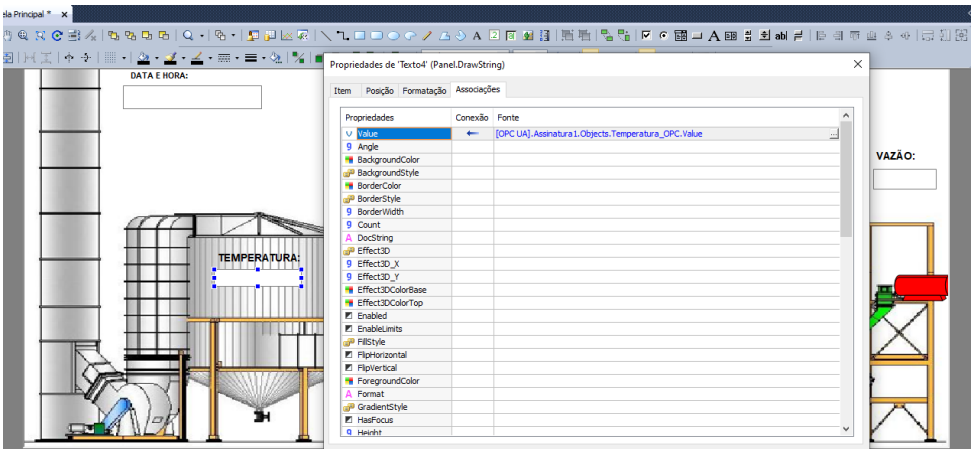
Como mostrado na Figura 11, cada indicador de campo (Nível, Temperatura e Vazão) recebeu uma associação simples, ligando o atributo *Value* do objeto na tela à sua *tag* calibrada em unidades de engenharia.

Figura 10 – Tela Principal de monitoramento da caldeira



Fonte: Produção própria (2026).

Figura 11 – Exemplo de associações das *tags* aos objetos da Tela Principal



Fonte: Produção própria (2026).

3.4.4 Acionamento da Bomba via *Script* de Inversão

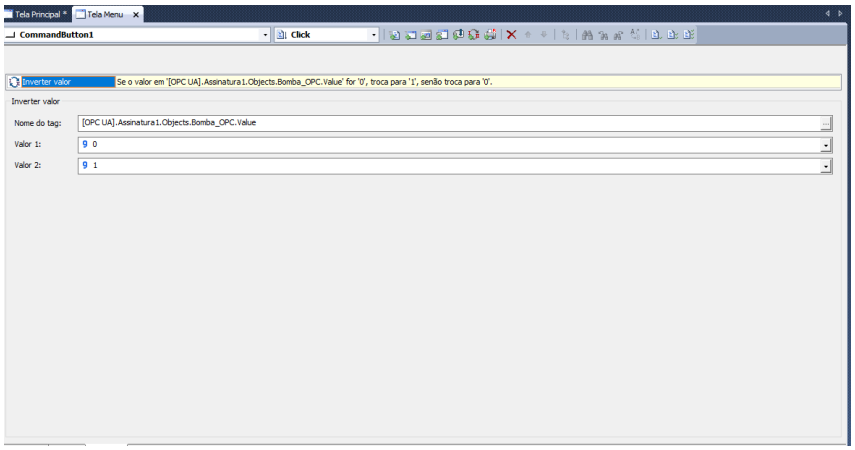
Para ligar ou desligar a bomba, implementou-se uma lógica de comando na "Tela de Menu". Diferentemente das variáveis de leitura, o acionamento do motor exige uma ação do operador que mude o estado do registrador no servidor OPC.

Para isso, desenvolveu-se um *script* no evento *Click* de um botão de comando, usando a lógica de inversão de estado (Figura 12), conforme os seguintes critérios:

- Se a *tag* for igual a 0 (bomba desligada), o *script* escreve o valor 1 (bomba ligada);
- Se o valor atual for igual a 1, o código muda o registrador para 0.

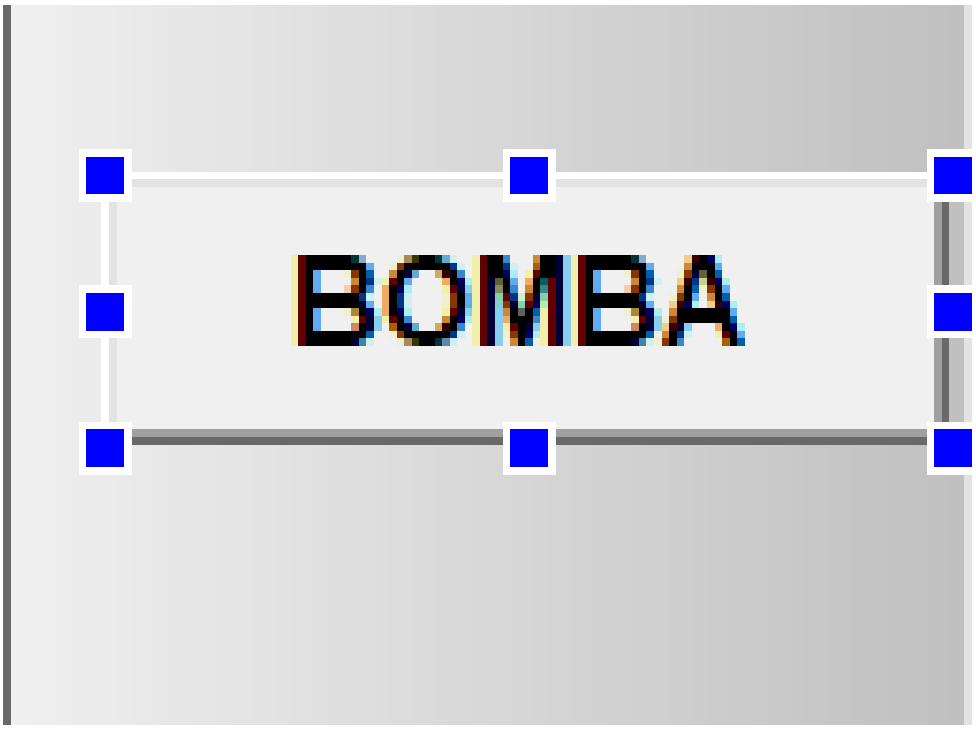
Assim, o mesmo botão funciona como um interruptor liga/desliga, simplificando a operação. O comando binário vai via OPC UA para o servidor, onde o script em Python lê a mudança de estado e dispara o aviso de voz correspondente.

Figura 12 – *Script* de comando para inversão de valor da *tag* Bomba



Fonte: Produção própria (2026).

Figura 13 – Visualização da Tela de Controle e *Status* da Bomba



Fonte: Produção própria (2026).

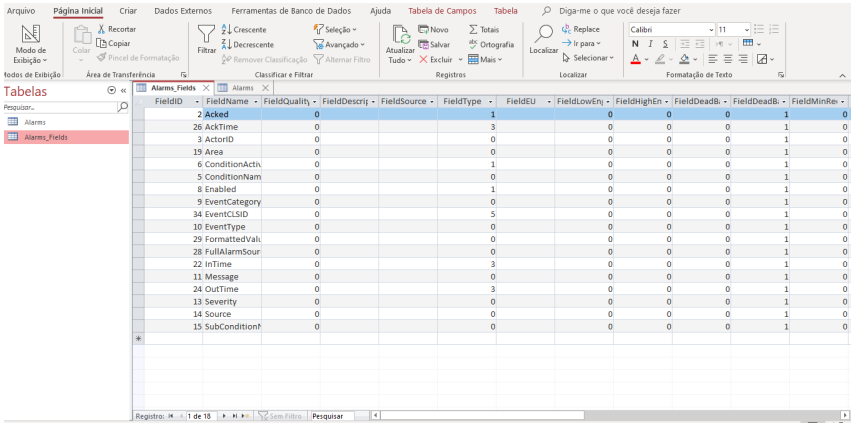
3.5 Integração de Dados e Gerenciamento de Alarmes

Esta seção detalha o desenvolvimento da camada de armazenamento de dados e a modelagem da lógica de monitoramento de alarmes, essenciais para garantir a segurança e a rastreabilidade do processo térmico.

3.5.1 Criação e Conexão com o Banco de Dados

Para garantir o armazenamento histórico das variáveis de processo e dos eventos da planta, foi desenvolvido um banco de dados no *Microsoft Access*, conforme mostrado na Figura 14. A organização das tabelas no arquivo `.mdb` permite que o sistema registre não apenas os parâmetros instantâneos, mas todo o comportamento dinâmico da caldeira durante a operação.

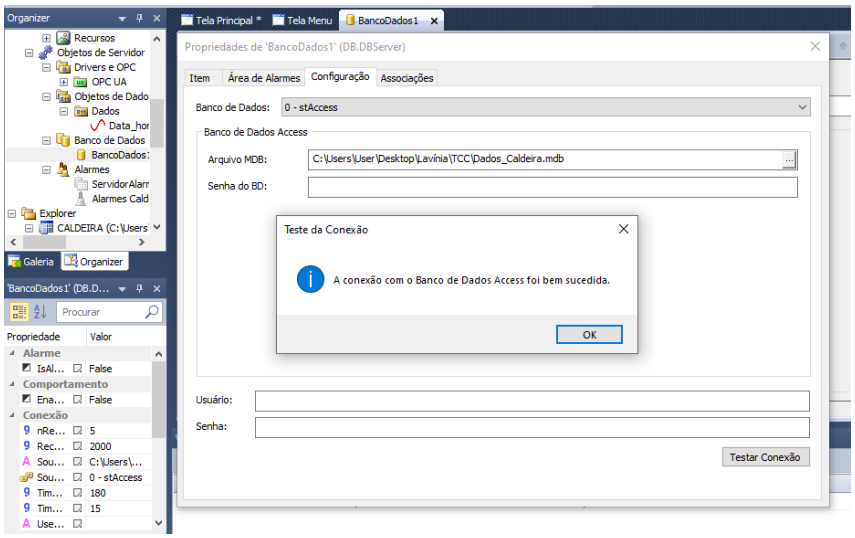
Figura 14 – Criação do banco de dados no *Access*



Fonte: Produção própria (2026).

A conexão foi estabelecida configurando-se um objeto de banco de dados no ambiente *Organizer* do supervisório. Na Figura 15, exibe-se o teste de comunicação realizado no Elipse E3, que confirma o sucesso na leitura e escrita de dados no arquivo externo. Essa etapa é necessária para garantir que as oscilações de temperatura, nível, vazão e os comandos da bomba sejam armazenados em tempo real.

Figura 15 – Validação da conexão bem-sucedida entre o Elipse E3 e o Banco de Dados *Access*

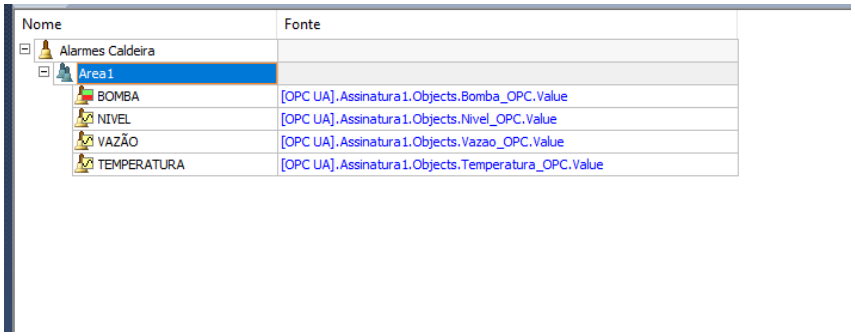


Fonte: Produção própria (2026).

3.5.2 Parametrização e Configuração de Alarmes

Após estruturar o banco de dados, realizou-se a configuração das variáveis no servidor de alarmes nativo do Elipse E3, vinculando as *tags* importadas via barramento OPC UA (Figura 16). Esse mapeamento define os limites lógicos que disparam os alertas visuais na tela e acionam o módulo de voz em Python.

Figura 16 – Criação e conexão de alarmes com as *tags*



Fonte: Produção própria (2026).

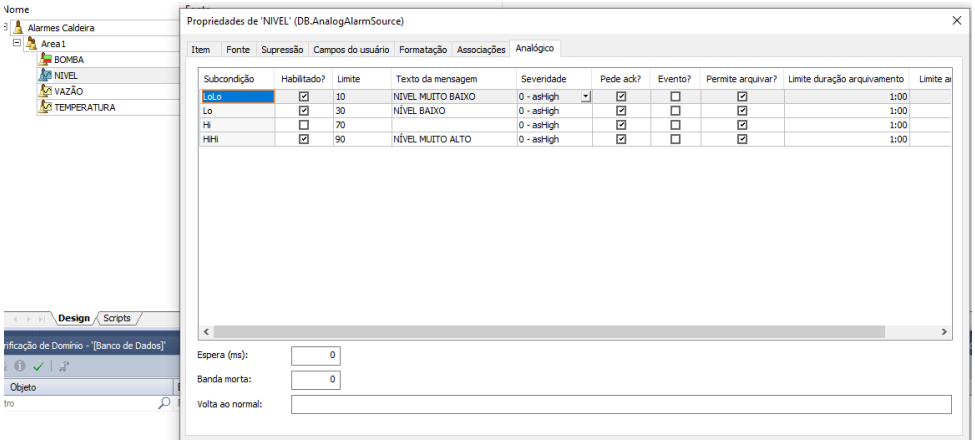
A configuração dos limites de alarme, apresentada na Figura 17, foi organizada com base em princípios de gestão de alarmes da norma ISA 18.2. Os valores adotados foram definidos como parâmetros de teste para representar diferentes condições de operação no ambiente simulado. Para as três variáveis principais do processo, foram configuradas mensagens específicas enviadas ao *script* em *Python*.

A NR-13 foi utilizada como referência para contextualizar a criticidade associada à operação de caldeiras e vasos de pressão. Contudo, os limites numéricos definidos neste trabalho não representam valores universais para todas as caldeiras. Em uma aplicação industrial real, esses parâmetros devem ser estabelecidos conforme os dados de projeto, os procedimentos operacionais, as especificações dos equipamentos e as análises de risco.

- **Nível Hidráulico do Tanque:**
  - Muito Baixo (*Low-Low*): ativado em 15% (mensagem "Nível muito baixo"), representando uma condição de nível reduzido no ambiente simulado;
  - Muito Alto (*High-High*): ativado em 90% (mensagem "Nível muito alto"), representando uma condição de nível elevado no ambiente simulado.
- **Temperatura do Vaso de Pressão:**
  - Baixa (*Low*): configurada em 50 °C (mensagem "Temperatura baixa"), representando uma condição de temperatura reduzida no ambiente simulado;
  - Alta (*High*): configurada em 220 °C (mensagem "Temperatura alta"), valor adotado para representar uma condição de temperatura elevada no ambiente simulado.
- **Vazão de Fluido nas Tubulações:**
  - Baixa (*Low*): configurada em 10 m^3/h (mensagem "Vazão baixa"), representando uma condição de fluxo reduzido no ambiente simulado;
  - Alta (*High*): configurada em 45 m^3/h (mensagem "Vazão alta"), representando uma condição de fluxo elevado no ambiente simulado.

Essa organização permite gerar alertas distintos para cada variável monitorada e verificar, no ambiente simulado, a identificação das condições de alarme e o funcionamento da lógica de priorização das mensagens de voz.

Figura 17 – Parametrização técnica dos níveis de alarme e mensagens de notificação

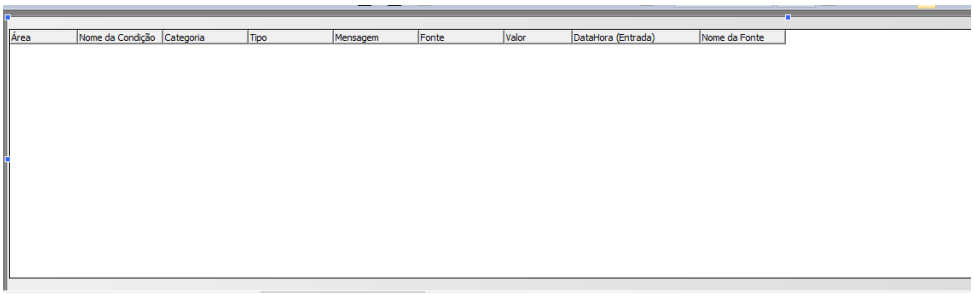


Fonte: Produção própria (2026).

Por fim, consolidou-se a ‘Tela de Alarmes”, apresentada na Figura 18. Essa tela é utilizada para o monitoramento das ocorrências durante a execução do supervisório (*runtime*) e exibe, de forma organizada, os campos fundamentais de cada evento, como Área, Nome da Condição, Mensagem, Valor Medido e ‘Data/Hora de Entrada”.

Um aspecto importante nesta implementação é que a tela de alarmes trabalha conectada ao banco de dados em *Microsoft Access* configurado previamente. Essa integração garante que cada registro de falha mostrado na tabela seja salvo de forma permanente, permitindo a rastreabilidade completa do histórico da caldeira, sem o risco de perda de informações após o reconhecimento do alarme pelo operador.

Figura 18 – Interface final da “Tela de Alarmes” integrada ao banco de dados



Fonte: Produção própria (2026).

3.5.3 Arquitetura do *Software* de Assistência por Voz e Filtros Cognitivos

O módulo de voz desenvolvido em Python foi estruturado em uma arquitetura *multithread* desacoplada para garantir a velocidade das leituras de rede e a estabilidade das mensagens de voz. O algoritmo foi dividido em duas frentes de execução paralelas:

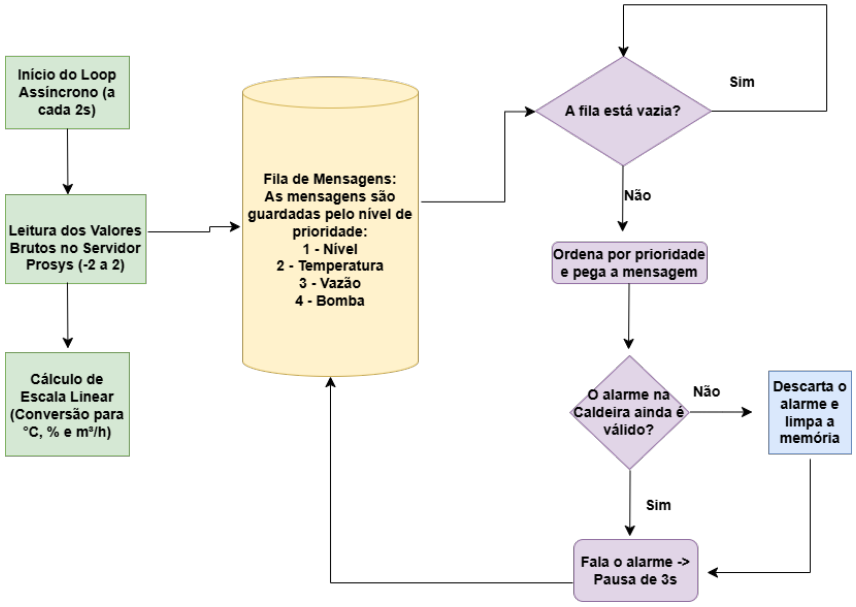
1. ***Subthread* Assíncrona de *Background*:** atua de forma contínua e realiza a coleta de dados via protocolo OPC UA no servidor *Prosys*, executando a varredura e o escalonamento matemático a cada 2 segundos, sem bloquear a aplicação;

2. ***Thread* Principal Síncrona (*Main Thread*):** gerencia e inicializa o motor de síntese de voz *Text-to-Speech* (`pyttsx3`), consumindo de forma sequencial as mensagens enviadas através de uma fila protegida contra concorrência do tipo `queue.Queue`.

Para fornecer mensagens padronizadas, curtas e objetivas, o algoritmo foi configurado para emitir alertas associados aos limites de alarme definidos no ambiente simulado. Adicionalmente, foi implementada uma rotina de *Validação de Estado Ativo*. Essa função verifica a condição da variável imediatamente antes da reprodução sonora, confirmando se a situação de alarme ainda permanece ativa.

Caso a variável tenha retornado à faixa normal durante o tempo de espera na fila, a mensagem é descartada. Dessa forma, evita-se a reprodução de alertas desatualizados e de mensagens que já não correspondem ao estado atual da variável monitorada. O fluxo de processamento de dados e o tratamento de concorrência entre as linhas de execução paralelas estão detalhados no diagrama de blocos da Figura 19.

Figura 19 – Diagrama de blocos funcionais da arquitetura de *software multithread*



Fonte: Produção própria (2026).

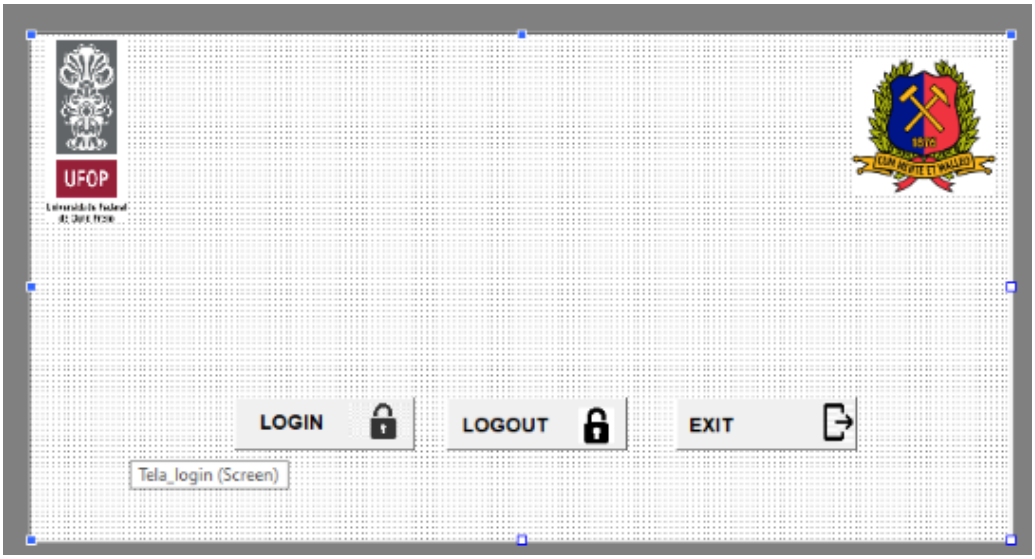
A lógica computacional completa do programa, contendo as rotinas de leitura assíncrona, o gerenciamento da fila e o tratamento do driver de áudio, está disponível no **Apêndice A**.

3.6 Segurança Operacional: Tela de Bloqueio e Controle de Acesso

A etapa final do desenvolvimento do sistema consistiu na criação de uma camada dedicada à segurança operacional e à integridade dos dados, utilizando uma interface de controle de acesso no Elipse E3. Alinhada aos requisitos de ergonomia e redução de falhas humanas da norma NR-17, essa estrutura restringe a operação da caldeira a profissionais qualificados e identificados.

A interface de bloqueio, chamada "Tela_login", foi projetada para ser simples e direta, incluindo os brasões da Universidade Federal de Ouro Preto (UFOP) e da Escola de Minas na parte superior. Três botões de comando foram dispostos de forma centralizada para gerenciar o acesso ao sistema, conforme a Figura 20.

Figura 20 – Interface gráfica final da tela de login desenvolvida



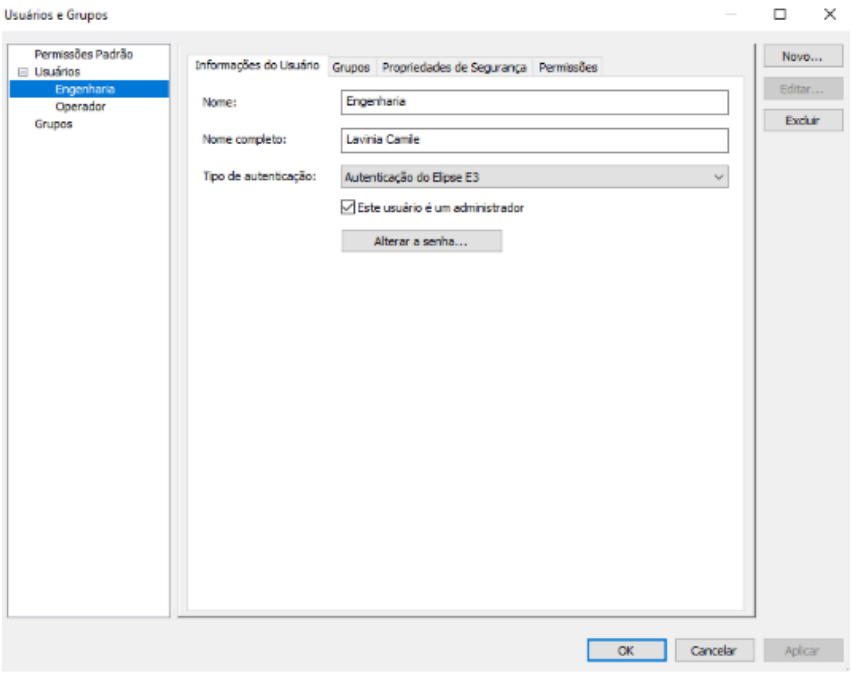
Fonte: Produção própria (2026).

Para gerenciar as responsabilidades na planta, utilizou-se a ferramenta de segurança do Elipse E3 para cadastrar dois perfis de usuários com privilégios distintos:

- **Engenharia:** perfil administrador com direitos totais para modificação, supervisão, configuração de limites técnicos e alterações no sistema;
- **Operador:** perfil restrito voltado apenas ao monitoramento das variáveis de processo (nível, temperatura, vazão e status da bomba), controle de alarmes e acionamentos rotineiros, sem permissão para mudanças estruturais.

A configuração desses usuários no banco de segurança está detalhada na Figura 21.

Figura 21 – Gerenciamento de usuários e atribuição de perfis de acesso



Fonte: Produção própria (2026).

3.6.1 *Scripts* de Execução e Ciclo de Sessão em *VBScript*

As ações de segurança foram automatizadas por *scripts* em *VBScript* associados aos botões da interface e às rotinas internas do aplicativo. Para gerenciar o acesso, a troca de turnos e o encerramento do sistema, foram desenvolvidas três rotinas vinculadas aos botões de comando:

1. *"CommandButton1" ("LOGIN")*: localizado na "Tela_login", abre a caixa de diálogo padrão para a inserção das credenciais do usuário;
2. *"CommandButton2" ("LOGOUT")*: posicionado no menu superior da "Tela Principal", desconecta o perfil ativo e retorna o supervisório para a tela de bloqueio;
3. *"CommandButton3" ("EXIT")*: inserido na "Tela_login", encerra a execução geral do supervisório e fecha o aplicativo.

Os *scripts* em *VBScript* desenvolvidos para o controle de acesso e gerenciamento das sessões são apresentados a seguir:

```
' 1. SCRIPT DE AUTENTICAÇÃO (Botão LOGIN na Tela_login)
Sub CommandButton1_Click()
Application.Login
End Sub
```

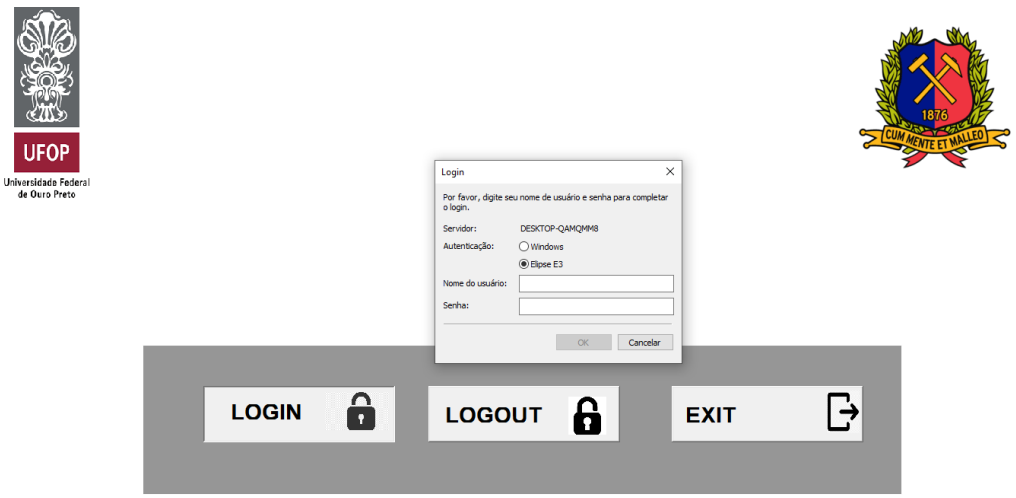
```
' 2. SCRIPT DE DESCONEXÃO (Botão LOGOUT na Tela Principal)
Sub CommandButton2_Click()
Application.Logout([Mode])
End Sub

' 3. SCRIPT DE FECHAMENTO (Botão EXIT na Tela_login)
Sub CommandButton3_Click()
Application.Exit()
End Sub
```

Para automatizar a transição de telas após a validação do usuário, configurou-se o evento *OnLogin* no *Viewer*. Esse evento é executado após a autenticação bem-sucedida e redireciona a navegação para a tela operacional da planta ("Quadro1"), conforme apresentado a seguir:

```
' SCRIPT DE DIRECIONAMENTO APÓS AUTENTICAÇÃO BEM-SUCEDIDA
Sub Viewer_OnLogin()
Application.GetFrame("").OpenScreen("Quadro1"), 0
End Sub
```

Durante a operação em tempo real (*runtime*), quando o botão de acesso é acionado, o sistema abre uma caixa de diálogo para a inserção de senha.(Figura 22).

Figura 22 – Caixa de diálogo flutuante de autenticação em modo de execução



Fonte: Produção própria (2026).

Após o login, o operador é direcionado à tela de monitoramento, onde um mostrador de texto exibe continuamente o nome do usuário ativo (por exemplo, “Engenharia”), garantindo a rastreabilidade e o controle de acesso exigidos pela operação.

4 Resultados e Discussões

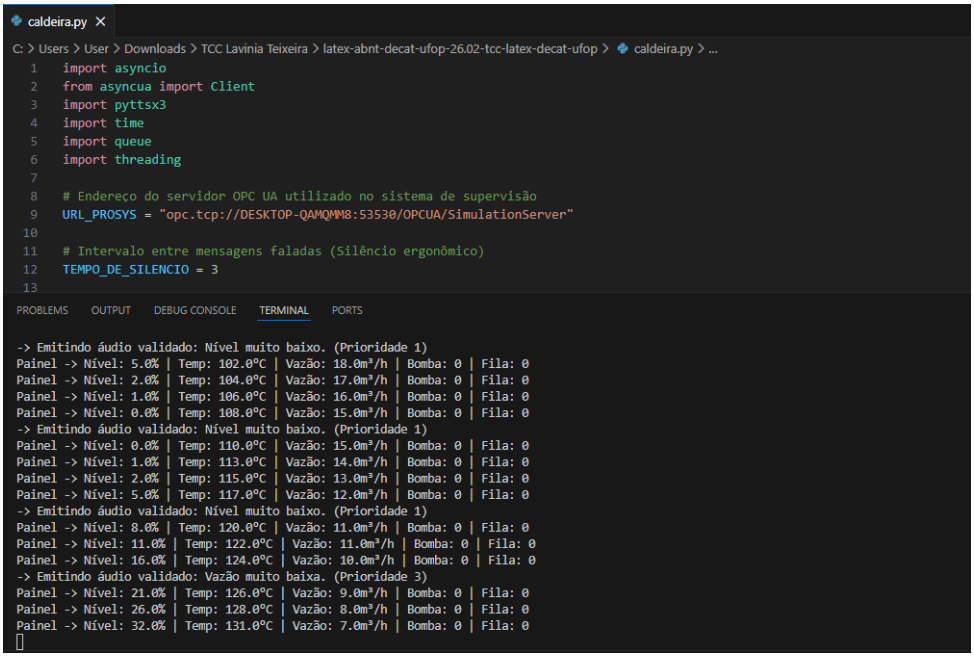
Neste capítulo são apresentados e analisados os resultados dos testes práticos realizados com o sistema integrado da caldeira industrial e o módulo de voz.

4.1 Funcionamento do Script em Python no Visual Studio Code

O algoritmo desenvolvido em Python foi validado no ambiente *Visual Studio Code* (*VS Code*), demonstrando estabilidade. A separação entre o loop de varredura de dados e a rotina de reprodução de áudio eliminou as interrupções de execução comuns em arquiteturas síncronas de thread única.

Como mostra a Figura 23, o console do terminal no *VS Code* registra o monitoramento em tempo real, exibindo as variáveis já convertidas de nível, temperatura e vazão. Os dados confirmam o sincronismo com o servidor OPC UA e demonstram a eficácia do filtro ao exibir o descarte de alarmes transientes. Esse comportamento prova que a variável do processo retornou à faixa de segurança antes da reprodução do áudio, confirmando o correto funcionamento das equações de escalonamento linear.

Figura 23 – Monitoramento de variáveis e execução do script no terminal do (*VS Code*)



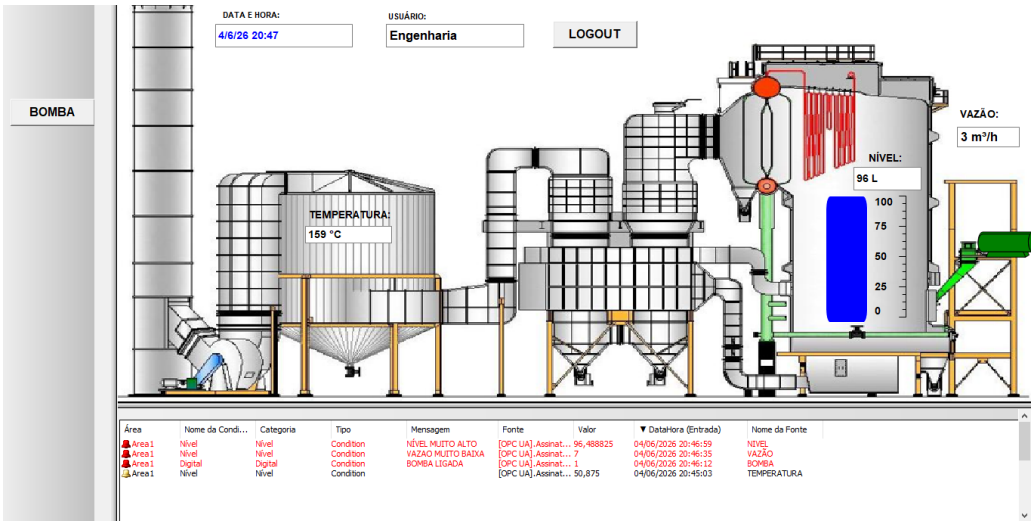
Fonte: Produção própria (2026).

4.1.1 Testes de Alarmes e Simulação Prática

Para validar o comportamento dinâmico do sistema integrado, realizou-se um teste de estresse simulando falhas simultâneas na caldeira. O objetivo foi avaliar a capacidade do algoritmo em Python para gerenciar a fila de prioridades em tempo real e verificar o funcionamento do controle de acesso no supervisório.

A Figura 24 apresenta o estado do supervisório durante a simulação das falhas concorrentes. No painel superior da tela, o sistema exibe o usuário ativo no campo "Usuário: Engenharia", confirmando a eficácia da tela de login e do controle de acesso. O carimbo de tempo indica o horário exato da ocorrência, sincronizado com a *tag* interna de data e hora.

Figura 24 – Simulação prática das telas do Elipse E3 com controle de acesso e disparos de alarmes



Fonte: Produção própria (2026).

No momento exibido na Figura 24, as variáveis atingiram patamares críticos. O nível do reservatório subiu até **96 L (96%)**, ultrapassando o limite superior de segurança ($\geq$90%). Ao mesmo tempo, a temperatura indicou **159 °C** e a vazão caiu para **3 m^3/h**, cruzando o limite inferior crítico ($\leq$10 m^3/h). Em resposta a essas falhas rápidas e ao acionamento manual da bomba (`Bomba = 1`), a tabela de eventos no rodapé exibiu as sinalizações de emergência em vermelho.

4.1.1.1 Validação da Hierarquia baseada na Norma ISA 18.2

Com o disparo simultâneo dessas falhas, o algoritmo em Python processou os alarmes respeitando a severidade de cada risco. Mesmo com a ativação da bomba, o aquecimento da caldeira e a queda na vazão ocorrendo juntos, o sistema priorizou a emergência de nível alto, inserindo-a no topo da fila e emitindo imediatamente o alerta de voz "Nível muito alto." (Prioridade 1) antes de processar as demais mensagens.

Essa ordenação valida a lógica de gerenciamento da norma ANSI/ISA 18.2 (HIRSCH; HOHER; HUBER, 2023).A lógica de fila e de validação de estado ativo organizou as mensagens, permitindo que o alarme de maior criticidade fosse reproduzido isoladamente, sem sobreposição de áudio. No ambiente simulado, esse comportamento evitou a emissão simultânea de mensagens e o envio de alertas cuja condição já havia sido normalizada. . A Tabela 1 resume os cenários operacionais configurados e as respostas automáticas geradas pelo módulo de voz.

Tabela 1 – Cenários de testes simulados e comportamento do assistente de voz

Variável Testada	**Valor Atingido**	**Alerta Emitido por Voz**	**Resultado Técnico**
Nível do Tanque	$\leq 15\%$	"Nível muito baixo."	(Prioridade 1)
Nível do Tanque	$\geq 90\%$	"Nível muito alto."	(Prioridade 1)
Temperatura	≤ 50 °C	"Temperatura muito baixa."	(Prioridade 2)
Temperatura	≥ 220 °C	"Temperatura muito alta."	(Prioridade 2)
Vazão de Fluido	≤ 10 m^3/h	"Vazão muito baixa."	(Prioridade 3)
Vazão de Fluido	≥ 45 m^3/h	"Vazão muito alta."	(Prioridade 3)
Status da Bomba	Alternância 0/1	"Bomba ligada" / "Bomba desligada"	(Prioridade 4)

Fonte: Produção própria (2026).

O tempo de varredura das variáveis via OPC UA foi de 2 segundos. Para a segurança de um vaso de pressão, essa taxa de atualização atende aos padrões de automação industrial, permitindo que o *script* em Python identifique variações rápidas do processo e avise o operador a tempo de evitar danos aos equipamentos.

O intervalo de 3 segundos (`TEMPO_DE_SILENCIO = 3`) entre as mensagens de voz garantiu que o operador assimilasse o aviso antes do início do próximo, evitando sobreposição sonora. A rotina de descarte de alarmes obsoletos (`INTERVALO_REPETICAO = 8`) também funcionou conforme esperado: se uma variável sofresse uma variação rápida e retornasse à faixa normal, o algoritmo identificava a normalização e limpava o alarme da fila, evitando avisos defasados e diminuindo a sobrecarga mental do operador.

5 Considerações Finais

Este Trabalho de Conclusão de Curso atingiu seu objetivo geral ao projetar, desenvolver e validar tecnicamente um sistema automatizado de notificação de alarmes por voz em tempo real, integrado ao supervisório Elipse E3. A simulação da caldeira industrial serviu como um cenário prático para verificar o funcionamento da integração entre os alertas visuais e auditivos, da comunicação OPC UA e da lógica de gerenciamento de falhas do processo.

Os objetivos específicos propostos foram cumpridos:

- Revisou-se a base teórica e normativa sobre Ergonomia Cognitiva e o fenômeno da fadiga de alarmes, fornecendo diretrizes para o projeto;
- Estruturou-se a arquitetura de comunicação baseada no protocolo OPC UA, permitindo a leitura das variáveis com tempo de varredura de dois segundos;
- Desenvolveu-se o módulo de voz em *Python*, utilizando a biblioteca `pyttsx3`, que funcionou de forma *offline* e com baixa latência; e
- Validou-se o sistema por meio de simulações de falhas simultâneas, confirmando a capacidade do algoritmo de gerenciar a fila conforme as prioridades de severidade.

Durante os testes de estresse do programa, identificou-se uma limitação técnica: o *driver* de voz nativo do Windows (*Speech Application Programming Interface 5* — SAPI5) apresentava falhas quando várias sub-rotinas tentavam reproduzir áudios ao mesmo tempo. Essa limitação ocorria porque o motor de áudio não gerenciava adequadamente requisições concorrentes assíncronas. O problema foi tratado no código com a implementação de uma fila síncrona protegida (`queue.Queue`) e a reinicialização dinâmica do motor de voz após cada mensagem reproduzida, garantindo maior estabilidade para a operação no ambiente simulado.

É importante destacar as diferenças entre o ambiente simulado deste projeto e uma planta industrial real. O uso do *Prosys OPC UA Simulation Server* foi eficiente para validar as lógicas, mapear os *NodeId* e testar a comunicação. Contudo, em uma planta real com um CLP físico, surgem fatores como ruídos eletromagnéticos na rede, oscilações nos sensores de campo e falhas pontuais na transmissão de dados, que podem aumentar a latência e exigir a implementação de filtros digitais adicionais.

Ressalta-se que a validação realizada neste trabalho ocorreu exclusivamente em ambiente simulado e não envolveu operadores humanos, seja em planta industrial, seja

em cenário experimental controlado. Portanto, não foram mensurados indicadores como carga mental, fadiga de alarmes, tempo de resposta, taxa de acerto ou redução de erros operacionais. Os princípios da Ergonomia Cognitiva e as diretrizes da NR-17 foram utilizados como fundamentação para o desenvolvimento da ferramenta, especialmente para a organização de mensagens curtas, priorizadas e complementares ao monitoramento visual do supervisório.

Dessa forma, o sistema desenvolvido deve ser compreendido como um produto de *software* funcional e um protótipo tecnicamente validado quanto à comunicação OPC UA, à identificação das condições de alarme, à fila de prioridades e à emissão de mensagens por voz. A arquitetura desenvolvida apresenta potencial para adaptação futura em uma planta industrial real. Entretanto, uma implantação em campo exigiria validação adicional, avaliação com operadores, definição dos limites reais de alarme e adequações relacionadas à segurança operacional e cibernética.

O armazenamento do histórico no banco de dados em *Microsoft Access* permitiu registrar os eventos gerados durante a simulação e apoiar a rastreabilidade das ocorrências. A NR-13 foi utilizada como referência para contextualizar a importância do registro de eventos relacionados à operação de caldeiras e vasos de pressão. Para uma futura aplicação em maior escala, é possível migrar o armazenamento para um Sistema Gerenciador de Banco de Dados Relacional, como *PostgreSQL*, *Microsoft SQL Server* ou *MySQL*.

Essa estrutura relacional poderia organizar tabelas de alarmes, variáveis monitoradas, usuários, perfis de acesso, reconhecimentos de eventos e histórico de comandos. Dessa forma, seria possível ampliar a rastreabilidade, preservar a integridade das informações e permitir acessos simultâneos por diferentes usuários ou estações de supervisão.

Sob o ponto de vista da Ergonomia Cognitiva, da NR-17 e das diretrizes da norma ISA 18.2, os alertas de voz curtos e diretos foram adotados como recurso complementar ao monitoramento visual do supervisório. Entretanto, a avaliação de sua influência sobre a atenção, a fadiga visual e o tempo de resposta dos operadores requer testes específicos com usuários humanos.

Como sugestão para trabalhos futuros, recomenda-se a expansão do sistema por meio do uso de modelos locais de Inteligência Artificial Generativa operando em dispositivos de borda (*edge devices*), permitindo que o programa sugira instruções detalhadas e planos de contingência com base no estado da caldeira. Além disso, sugere-se implementar a infraestrutura de banco de dados relacional mencionada e incluir o reconhecimento de voz para que o operador possa executar comandos diretos na planta — como falar “Desligar Bomba” — como canal complementar de controle, desde que sejam adotados mecanismos de autenticação, permissões de acesso, intertravamentos e análises de risco adequadas.

Referências

ARAÚJO, Eduardo Wagner Santos de; FÁTIMA FERRAZ DA SILVA TACCONI, Marli de. A Ergonomia Cognitiva na Vida do Trabalhador. *Revista Estudo & Debate*, Lajeado, v. 30, n. 4, p. 180–203, 2023. Acesso em: 17 maio 2026. ISSN 1983-036X. DOI: http://dx.doi.org/10.22410/issn.1983-036X.v30i4a2023.3511. Disponível em: http://dx.doi.org/10.22410/issn.1983-036X.v30i4a2023.3511. Citado 4 vezes nas páginas 24, 25, 27.

ARRUDA, Agnaldo Fernando Vieira de *et al.* Contribuições da Ergonomia Cognitiva para a Segurança do Trabalho no Setor Mineral. *Anais do Simpósio Brasileiro de Engenharia de Produção*, 2006. Acesso em: 17 maio 2026. Disponível em: http://www.deps.ufsc.br. Citado 2 vezes nas páginas 24, 25.

BRASIL. *Norma Regulamentadora n. 13 (NR-13): Caldeiras, Vasos de Pressão, Tubulações e Tanques Metálicos de Armazenamento*. 2023. Disponível em: https://www.gov.br/trabalho-e-emprego/.... Acesso em: 17 maio 2026. Citado 2 vezes nas páginas 26, 27.

BRASIL. *Norma Regulamentadora n. 17 (NR-17): Ergonomia*. 2023. Disponível em: https://www.gov.br/trabalho-e-emprego/.... Acesso em: 17 maio 2026. Citado 2 vezes na página 27.

FRAGA, Filipe. *Supervisórios Elipse E3*. 2020. Disponível em: https://www.youtube.com/playlist?list=PLpveIalqkpCY0j2_Gb1CHM4Zhcy8UD7YJ. Citado 1 vez nas páginas 28, 30.

FUMAGALLI, Marco Antônio; BRANCO, Augusto Afonso Castelo; DIAS, Wellington Santana. Implementação de protocolo OPC UA para controle de uma célula de manufatura, utilizando framework Web voltado para Indústria 4.0. *FTT Journal of Engineering and Business*, São Bernardo do Campo, SP, p. 107–117, 2020. ISSN 2525-8729. Citado 4 vezes nas páginas 20, 21.

HIRSCH, Eduard; HOHER, Simon; HUBER, Stefan. An OPC UA-based industrial Big Data architecture. *arXiv preprint arXiv:2306.01418*, 2023. Disponível em: https://arxiv.org/abs/2306.01418. Citado 6 vezes nas páginas 21, 22, 48.

INTERNATIONAL SOCIETY OF AUTOMATION. *ANSI/ISA-18.2-2016: Management of Alarm Systems for the Process Industries*. Research Triangle Park, NC, 2016. Citado 1 vez na página 25.

LEITNER, Stefan-Helmut; MAHNKE, Wolfgang. OPC UA – Service-oriented Architecture for Industrial Applications. *In:* SOFTWARE Engineering 2006. GI, 2006. p. 61–72. Acesso em: 17 maio 2026. Disponível em: https://dl.gi.de/handle/20.500.12116/41561. Citado 3 vezes na página 20.

OPC FOUNDATION. *OPC 10000-1: Unified Architecture – Part 1: Overview and Concepts*. Version 1.05.03. Scottsdale: OPC Foundation, 2022. Disponível em: https://reference.opcfoundation.org/specs/OPC-10000-1/1. Acesso em: 17 maio 2026. Citado 4 vezes nas páginas 21, 30.

PAIOLA, C. E. G.; ROCHA, E. H.; RODRIGUES, A. C. A Importância das Normas de Automação para a Indústria 4.0. *The Journal of Engineering and Exact Sciences*, Universidade Federal de Viçosa, v. 5, n. 5, p. 0415–0423, 2019. Citado 1 vez na página 26.

PÉREZ-LÓPEZ, Esteban. Los sistemas SCADA en la automatización industrial. *Tecnología en Marcha*, Instituto Tecnológico de Costa Rica, v. 28, n. 4, p. 3–14, 2015. Citado 2 vezes na página 17.

SILVA, Ricardo Luís Alves da; GIRÃO, Tulio Aoki. Avaliações Ergonômicas em Salas de Controle Operacional: Um Estudo de Caso. *In:* ANAIS do III Simpósio de Engenharia de Produção (SIMEP). Joinville, SC, 2015. p. 1–12. Acesso em: 17 maio 2026. Disponível em: http://minas.simep.com.br. Citado 2 vezes nas páginas 24, 25.

SILVEIRA, Leonardo; LIMA, Weldson Q. Um breve histórico conceitual da automação industrial e redes para automação industrial. *Universidade Federal do Rio Grande do Norte*, 2014. Acesso em: 25 jan. 2026. Disponível em: https://d1wqtxts1xzle7.cloudfront.net/43839581/Redes_Industriais_e_Automacao-libre.pdf. Citado 2 vezes na página 13.

SOUZA, Rodrigo Barbosa de. *Uma arquitetura para Sistemas Supervisórios Industriais e sua Aplicação em Processos de Elevação Artificial de Petróleo*. 2005. Dissertação de Mestrado – Universidade Federal do Rio Grande do Norte, Natal, RN. Citado 8 vezes nas páginas 17–19, 21–23.

WICKENS, Christopher D. Multiple Resources and Mental Workload. *Human Factors*, SAGE Publications, v. 50, n. 3, p. 449–455, 2008. Citado 2 vezes na página 26.

APÊNDICE A – Código Fonte do Assistente de Voz Industrial

```
import asyncio
from asyncua import Client
import pyttsx3
import time
import queue
import threading

# Endereço do servidor OPC UA utilizado no sistema de supervisão
URL_PROSYS = "opc.tcp://DESKTOP-QAMQMM8:53530/OPCUA/SimulationServer"

# Intervalo entre mensagens faladas (Silêncio ergonômico)
TEMPO_DE_SILENCIO = 3

# Tempo mínimo para repetir um alarme ainda ativo na tela
INTERVALO_REPETICAO = 8

# Fila responsável por armazenar os alertas que serão convertidos em voz
fila_de_voz = queue.Queue()

# Variáveis compartilhadas entre as threads (Estado em tempo real)
nivel_atual = 50
temp_atual = 100
vazao_atual = 25
bomba_atual = 0

# Controle de tempo do último disparo de cada alarme (Prevenção de saturação)
ultimos_disparos = {
 "nivel_alto": 0,
 "nivel_baixo": 0,
 "temp_alta": 0,
 "temp_baixa": 0,
 "vazao_alta": 0,
 "vazao_baixa": 0
```

```
}

# Usado para detectar mudança de estado da bomba
ultimo_estado_bomba = None


def converter_para_escala_real(sinal_bruto, minimo, maximo):
"""
Converte o valor bruto do servidor OPC para a escala real da variável.
O servidor de simulação fornece valores normalizados entre -2 e +2.
"""
calculo = ((sinal_bruto + 2) / 4) * (maximo - minimo) + minimo
return round(calculo, 0)


def alarme_ainda_valido(mensagem):
"""
Antes de emitir o áudio, verifica se a condição de alarme ainda está
presente no processo.
"""
if "Nível muito alto" in mensagem and nivel_atual < 90:
return False
if "Nível muito baixo" in mensagem and nivel_atual > 15:
return False
if "Temperatura muito alta" in mensagem and temp_atual < 220:
return False
if "Temperatura muito baixa" in mensagem and temp_atual > 50:
return False
if "Vazão muito alta" in mensagem and vazao_atual < 45:
return False
if "Vazão muito baixa" in mensagem and vazao_atual > 10:
return False
return True


async def monitoramento_opc():
"""
Rotina assíncrona responsável por ler continuamente os dados
do servidor OPC UA e gerar os gatilhos de alarme por prioridade.
```

```
"""
global ultimo_estado_bomba
global ultimos_disparos
global nivel_atual, temp_atual, vazao_atual, bomba_atual

async with Client(url=URL_PROSYS) as client:
sensor_temp = client.get_node("ns=3;i=1003")
tag_bomba = client.get_node("ns=5;s=Int32DataItem")
sensor_nivel = client.get_node("ns=3;i=1004")
sensor_vazao = client.get_node("ns=3;i=1002")

while True:
try:
bruto_temp = await sensor_temp.get_value()
bruto_bomba = await tag_bomba.get_value()
bruto_nivel = await sensor_nivel.get_value()
bruto_vazao = await sensor_vazao.get_value()

temp_atual = converter_para_escala_real(bruto_temp, 25, 250)
nivel_atual = converter_para_escala_real(bruto_nivel, 0, 100)
bomba_atual = int(bruto_bomba)

vazao_normal = converter_para_escala_real(bruto_vazao, 0, 50)
vazao_atual = 50 - vazao_normal

print(
f"Painel -> Nível: {nivel_atual}% | "
f"Temp: {temp_atual}°C | "
f"Vazão: {vazao_atual}m³/h | "
f"Bomba: {bomba_atual} | "
f"Fila: {fila_de_voz.qsize()}"
)

if nivel_atual >= 90:
if time.time() - ultimos_disparos["nivel_alto"] > INTERVALO_REPETICAO:
fila_de_voz.put((1, "Nível muito alto."))
ultimos_disparos["nivel_alto"] = time.time()
else:
ultimos_disparos["nivel_alto"] = 0
```

```
if nivel_atual <= 15:
if time.time() - ultimos_disparos["nivel_baixo"] > INTERVALO_REPETICAO:
fila_de_voz.put((1, "Nível muito baixo."))
ultimos_disparos["nivel_baixo"] = time.time()
else:
ultimos_disparos["nivel_baixo"] = 0

if temp_atual >= 220:
if time.time() - ultimos_disparos["temp_alta"] > INTERVALO_REPETICAO:
fila_de_voz.put((2, "Temperatura muito alta."))
ultimos_disparos["temp_alta"] = time.time()
elif temp_atual <= 50:
if time.time() - ultimos_disparos["temp_baixa"] > INTERVALO_REPETICAO:
fila_de_voz.put((2, "Temperatura muito baixa."))
ultimos_disparos["temp_baixa"] = time.time()
else:
ultimos_disparos["temp_alta"] = 0
ultimos_disparos["temp_baixa"] = 0

if vazao_atual >= 45:
if time.time() - ultimos_disparos["vazao_alta"] > INTERVALO_REPETICAO:
fila_de_voz.put((3, "Vazão muito alta."))
ultimos_disparos["vazao_alta"] = time.time()
elif vazao_atual <= 10:
if time.time() - ultimos_disparos["vazao_baixa"] > INTERVALO_REPETICAO:
fila_de_voz.put((3, "Vazão muito baixa."))
ultimos_disparos["vazao_baixa"] = time.time()
else:
ultimos_disparos["vazao_alta"] = 0
ultimos_disparos["vazao_baixa"] = 0

if ultimo_estado_bomba is not None:
if bomba_atual != ultimo_estado_bomba:
status = "ligada" if bomba_atual == 1 else "desligada"
fila_de_voz.put((4, f"Bomba {status}."))

ultimo_estado_bomba = bomba_atual
```

```
except Exception as e:
print(f"Erro na leitura OPC UA: {e}")

await asyncio.sleep(2)


def rodar_loop_opc():
loop = asyncio.new_event_loop()
asyncio.set_event_loop(loop)
loop.run_until_complete(monitoramento_opc())


if __name__ == "__main__":
threading.Thread(target=rodar_loop_opc, daemon=True).start()

while True:
if not fila_de_voz.empty():
alertas_da_rodada = []

while not fila_de_voz.empty():
alertas_da_rodada.append(fila_de_voz.get())

alertas_da_rodada.sort(key=lambda x: x[0])

for prioridade, message in alertas_da_rodada:
if alarme_ainda_valido(message) or prioridade == 4:
print(f"-> Emitindo áudio validado: {message} (Prioridade {prioridade})")
try:
engine = pyttsx3.init()
engine.setProperty('rate', 165)
engine.say(message)
engine.runAndWait()
del engine
except Exception as e:
print(f"Erro físico no driver de áudio: {e}")

time.sleep(TEMPO_DE_SILENCIO)
else:
print(f"-> [REJEITADO] Estado normalizado. Alarme descartado: {message}")
```

```
time.sleep(0.5)
```