Universidade Federal de Santa Catarina
Centro de Blumenau
Departamento de Engenharia de
Controle e Automação e Computação

Stephan August Roehrig Domingues dos Santos

Programação e simulação de um sistema de automação do setor operacional de uma indústria de grãos

Blumenau
2018

Stephan August Roehrig Domingues dos Santos

Programação e simulação de um sistema de automação do setor operacional de uma indústria de grãos

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Santa Catarina como parte dos requisitos necessários para a obtenção do Título de Engenheiro de Controle e Automação.
Orientador: Prof. Dr. Daniel Martins Lima

Universidade Federal de Santa Catarina
Centro de Blumenau
Departamento de Engenharia de
Controle e Automação e Computação

Blumenau
2018

Stephan August Roehrig Domingues dos Santos

Programação e simulação de um sistema de automação do setor operacional de uma indústria de grãos

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Santa Catarina como requisito parcial para a obtenção do título de Engenheiro de Controle e Automação.

Comissão Examinadora

Prof. Dr. Daniel Martins Lima
Universidade Federal de Santa Catarina
Orientador

Prof. Dr. Leonardo Mejia Rincon
Universidade Federal de Santa Catarina

Prof. Dr. Marcos Vinicius Matsuo
Universidade Federal de Santa Catarina

Blumenau, 20 de fevereiro de 2019

Dedico este trabalho a todos aqueles que, de alguma forma,
auxiliaram para a concretização desta etapa.

Agradecimentos

Gostaria de agradecer imensamente aos meus pais, Andre e Patricia, e ao meu irmão Johann.

A André Reuter, Brunno Vanelli, Enrique Geske, Guilherme Manerichi, Juliano Masson e Rômulo Pacher, grandes amigos que estiveram em todos os momentos da minha trajetória acadêmica.

Agradeço aos meus colegas de classe e com certeza futuros excelentes profissionais.

Agradeço à empresa GreyLogix Brasil, pela oportunidade de fazer estágio supervisionado. Foi com essa oportunidade que tornou esse trabalho possível. Obrigado Rafael pela oportunidade e Adir, por me ensinarem a prática dos conhecimentos que adquiri na faculdade.

Sou grato a todos os professores que contribuíram com a minha trajetória acadêmica, especialmente ao Daniel Martins Lima, responsável pela orientação do meu projeto de pesquisa, e deste presente trabalho.

À UFSC Blumenau, que apesar das dificuldades, ofereceu um ambiente de estudo agradável, motivador e repleto de oportunidades.

“Deyr fé
deyja frændr
deyr sjálfr it sama
ek veit einn
at aldri deyr
dómr um dauðan hvern″
(Hávamál 77)

Resumo

Com o crescimento do agronegócio, a cooperativa agroindustrial COAMO, viu a oportunidade de expandir os seus negócios, instalando uma nova indústria de extração de óleo de soja na cidade de Dourados/MS. Dentro desse projeto a empresa GreyLogix Brasil tem como objetivo fazer a programação e criação dos sistemas de controle da planta, simulação de toda a unidade industrial e criação da documentação. Dentro desse projeto, uma das áreas que deve ser finalizada por primeiro é o chamado setor operacional, responsável pela limpeza, pré-limpeza, secagem e estocagem dos grãos de soja.

Na unidade do operacional, ou entreposto, a soja pode chegar com diversas qualidades. Isso faz que a unidade possua diferentes rotas para movimentar os grãos entre os diferentes equipamentos. Para isso esse trabalho se propõem em fazer a programação de um sistema distribuído de automação aplicado ao setor. Bem como a simulação da área, afim de validar a programação feita.

Para cumprir esses objetivos foram usadas as ferramentas de engenharia PCS7, COMOS e SIMIT, da multinacional Siemens. Na programação foi usado a abordagem de intertravamentos, para limitar as ações dos operadores. Já a simulação foi feita usando a linguagem de blocos do *software* SIMIT.

Como principal resultado do projeto pode-se citar as primeiras versões da simulação e do código de controle. Bem como, uma primeira versão da biblioteca desenvolvida para fazer a simulação de transporte de grãos, que futuramente poderá ser utilizada em outros projetos da empresa GreyLogix Brasil.

Todos esses resultados se mostraram satisfatórios, levando em conta o tempo e as informações sobre o processo recebidas. Porém ainda será necessário fazer revisões, principalmente, durante o comissionamento da planta.

Palavras-Chave: 1. Transporte de grãos. 2. Simulação. 3. Automação. 4. Distribuited Control System. 5. Setor Operacional.

Abstract

With growing of the agribusiness, the agro-industrial cooperative COAMO, saw the oportunity to expand theres business, building a new soya-bean oil extraction industry, in the city of Dourados/MS. Inside this project the company GreyLogix Brasil have the objective to program the control systems, create the simulation of the entire facility and write of the documentation. One of the parts of this facility that needs to be conclude for first is the operational warehouse, responsible for the receiving, cleaning, pre-cleaning, drying and grain storage.

In the unity of the operational warehouse, the soya can arrive with diferents qualities. With this, the unity need to have different routes to move the grains between the diferents equipaments. So, this work proposes to program the control of the distribuited control system, as well as the programming of a simulation for this unity, aiming to validate the control system. Beyond that, it was necessary to make some documents revision.

To fullfill this objectivesthe engineering tools PCS7, COMS and SIMIT, from SIEMENS, were used. An interlock approach was used to program the control system, limiting the actions of the future unity operator. And, to make the simulation, the block programming language from SIMIT was used.

The simulation and the control software were the main results of this project. Besides that, the other achieved result, was the library developed to make the simulation. In the future this library may be used in others GreyLogix Brasil projects.

Those results shows a satisfactory behavior, taking in the acount the received information. However, some revisions are still necessary, mainly during the commissioning of the unity.

Keywords: 1. Soya-bean Transport. 2. Simulation. 3. Atomation. 4. Distribuited Control System. 5. Operational Warehouse.

Lista de figuras

Lista de tabelas

Lista de siglas e abreviaturas

UFSC	*Universidade Federal de Santa Catarina*
CMT	*Control module types*
CM	*Control module*
OSI	*Open System Interconnection*
P&ID	*Pipes and Instruments Diagram (Diagrama de tubos e instrumentos)*
AS-Interface	*Actuator and Sensors Interface*
POO	*Programação Orientada a Objetos*
VC	*Virtual Controler*
DCS	*Distribuited Control System*
CLP	*Controlador Lógico Programavél*
LT	*Ladder Diagram*
ST	*Structured Text*
IL	*Instruction List*
SFC	*Sequential Function Chart*
FBD	*Function Block Diagram*
CFC	*Continuous Function Chart*
GSD	*Geral Station Description*
NRT	*Non-real time*
RT	*Real time*
IRT	*Isochronous Real time*

Sumário

1 Introdução

1.1 Contextualização

Segundo o ministro da agricultura, Blairo Maggi, no ano de 2017 o setor do agronegócio foi responsável por um terço do crescimento total do pais [21]. Dentre as principais *commodities* do agronegócio a soja vem ganhando destaque principalmente devido aos seus usos na produção de ração animal e óleo vegetal. O óleo da soja pode ser usado tanto na fabricação de combustíveis, quanto como óleo alimentício. A produção da soja dentro do Brasil vem crescendo desde os anos 2000, esse acelerado crescimento deu ao pais, no ano de 2014, o título de maior produtor de soja do mundo [22].

Dentro deste cenário de crescimento do agronegócio a COAMO, uma das maiores empresas do ramo, viu a oportunidade de expansão de suas operações implementando um novo parque industrial de extração e refino de óleo de soja. Neste projeto, a empresa GreyLogix Brasil é responsável por fazer a integração dos diversos equipamentos, bem como a programação das lógicas de automação, desenho das telas de controle das plantas, escrita da documentação técnica e as simulações de todos os processos da indústria.

Um dos primeiros setores que devem ser finalizados nessa obra é o chamado setor operacional, responsável pelo: recebimento, pré-limpeza, secagem, limpeza e armazenamento dos grãos de soja. Antes desse setor, a soja recebida no parque industrial é analisada. Com os resultados dessa classificação é definido por quais equipamentos a soja recebida irá passar. Por exemplo, supondo que a soja que chegou no parque industrial está muito úmida, então, antes dela ser armazenada é necessário passa-lá pelo secador. Os diferentes caminhos que o grão da soja pode fazer dentro do setor de recebimento é o principal desafio de controle da planta.

Sendo assim, o presente trabalho realizado na GreyLogix Brasil tem por objetivo projetar parte da automação da nova unidade industrial da COAMO. Mais especificamente no setor operacional, focando em, principalmente, fazer a programação da automação e a simulação desse processo.
É importante ressaltar que as informações aqui apresentadas serão parcialmente censuradas para evitar expor os segredos industriais, mas mantendo o mínimo necessário para o entendimento do trabalho.

1.1.1 Bilfinger GreyLogix GmbH

Fundada a partir do departamento de automação da empresa HGC (Hamburg Gas Consult) em 2000 a Bilfinger GreyLogix GmbH é uma da maiores empresa de soluções em automação industrial. Sediada na Alemanha, a empresa hoje conta com o título de maior

Solution Partner da *SIEMENS* na Alemanha e uma das maiores empresas de automação no mundo [23].

Em 2013 a GreyLogix GmbH foi comprada pela Bilfinger *Industrial Technologies*, empresa especializada em projetos e construção de plantas industriais. Atualmente a Bilfinger GreyLogix conta com mais de 600 funcionários e 23 escritórios em diferentes países da Europa [23].

Desde 2016 a empresa ampliou seu leque de atuação, passando a ser uma construtora de plantas industriais sob o conceito de Indústria 4.0, ao adquirir conhecimentos da própria Bilfinger [23].

1.1.1.1 GreyLogix Brasil

A GreyLogix Brasil foi fundada por dois sócios, Renato Leal e Rafael Gonçalves (ex-alunos do curso de engenharia de controle e automação da UFSC), que estagiaram na GreyLogix da Alemanha. Ao retornarem do estágio, os dois sócios quiseram replicar o modelo de negócio descentralizado utilizado na empresa alemã. Hoje a GreyLogix Brasil é uma das maiores empresas no ramo de automação contando com uma sede administrativa localizada na cidade de Mafra/SC, 6 escritórios espalhados pelo Brasil, uma fábrica de montagem de painéis elétricos em Rio Negro/PR e mais de 70 colaboradores.

A GreyLogix é uma das mais completas empresas de automação do país. Sendo assim ela atua desde a parte de consultoria básica até o pós-venda. Atualmente os dois principais ramos nos quais a empresa atua são o de papel e celulose e o de comidas e bebidas, porem fazem projetos em outras áreas, como a manufatura [24]. Além disso a GreyLogix Brasil é uma das 20 empresas nacionais com a certificação de *solution partner* da multinacional *SIEMENS*.

1.1.2 COAMO

A cooperativa agroindustrial morouense, COAMO, foi fundada em 1970, inicialmente focada apenas na estocagem e comercialização de grãos de trigo. Foi somente em 1975 que foi implementada o moinho de trigo, que foi a primeira indústria da cidade de Campo Mourão/PR. Já nos anos 80, quando o mercado agroindustrial teve um grande impulso, foi montada as plantas de fiação de algodão e extração de óleo de soja, ampliando o mercado da cooperativa. Hoje a COAMO atua principalmente, na produção e venda de produtos derivados da soja, como o óleo-de-soja, gordura vegetal e margarina.

Além desses produtos derivados da soja, as industrias da COAMO também produzem farinha de trigo e café. Sua produção, segundo dados de 2016, corresponde por 3,5% da produção nacional de grãos e fibras [25]. Além disso a empresa se destaca como a segunda maior empresa do ramo no Brasil e a maior do estado do Paraná [26].

1.1.2.1 Projeto de Dourados

Com o crescimento do mercado do agronegócio no Brasil, a COAMO viu a necessidade de expandir suas operações industriais, que até então eram feitas somente na cidade de Campo Mourão/PR. Nesse contexto a empresa decidiu montar um novo parque industrial no Mato Grosso do Sul, na cidade de Dourados. Com previsão de inauguração no segundo semestre de 2019, a fábrica terá a capacidade de armazenar 3000 toneladas/dia, e com capacidade de produção de 720 toneladas/dia de óleo, mais que dobrando a capacidade de refino de óleo de soja, indo de 660 toneladas/dia para 1380 toneladas/dia [27].

1.1.3 Automação da operação

O setor operacional de uma industria de óleo de soja pode ser entendido como um conjunto de processos responsáveis por: receber, pré-limpar, secar, limpar e armazenar os grãos. Como alguns produtores possuem parte deste setor em suas fazendas é necessário que o setor tenha diversas rotas, possibilitando que o produto pule etapas. Sendo assim o setor operacional, no ponto de vista da automação é um problema de criar as rotas entre as diversas etapas do processo. Assim cabe ao operador da unidade definir em quais das etapas os grãos deverão passar. Fora isso, dependendo do tempo de armazenagem, os grãos já secos podem voltar a ter uma alta umidade, assim é necessário que os equipamentos desse setor possibilitem manobras de recirculação.

A definição das rotas é feita por uma série de válvulas direcionadoras e *trippers* de esteiras, alguns desses equipamentos serão explicados na seção 2.1.1 do capítulo 2. Além desses equipamentos o setor operacional é composto, também, por moegas, esteiras, elevadores, silos, maquinas de limpeza e pré limpeza e um secador de grãos.

1.2 Motivações

A principal motivação desses projeto é tentar fazer com que o setor operacional da nova fábrica da COAMO, seja o mais automatizado possível. Bem como fazer a simulação desse setor, o que é algo inédito, pelo menos, dentro da empresa GreyLogix. Os resultados desse projeto poderão servir de base para fazer simulações de projetos parecidos facilitando a programação para os colaboradores da empresa.

Além disso, existe a motivação pessoal desses projeto, de: ganhar experiencia no mercado de trabalho e aprimorar e aplicar os conhecimentos adquiridos durante o curso de engenharia de controle e automação em um projeto real de engenharia.

1.3 Objetivos

1.3.1 Objetivos Gerais

Os objetivos gerais deste trabalho serão: fazer a programação da automação e da simulação do setor operacional da nova indústria da COAMO.

1.3.2 Objetivos Específicos

Sendo assim como objetivos específicos deste trabalho ter-se-á:

- Revisão das documentações fornecidas;
- Revisão das redes de comunicação industrial;
- Programação das lógicas de controle;
- Programação da simulação do setor em questão;
- Revisão das telas do supervisório.

1.3.2.1 Revisão das documentações fornecidas

Deve-se revisar e estudar as documentações fornecidas pelas as diversas empresas parceiras. Essas documentações são constituídas de Diagramas de Tubos e Instrumentações (P&ID), listas dos instrumentos e motores.

Um aspecto importante que a revisão e o estudo da documentação que deve ser levado em conta é o fato de que isso ajuda a aprofundar o conhecimento do processo. Fato esse que torna mais fácil a conclusão dos próximos objetivos abordados nesta seção.

1.3.2.2 Revisão das Redes de Comunicação Industrial

A tarefa de modelar as redes, consiste em criar e atribuir os endereços dos equipamentos nas redes Profinet e AS-i dentro do PCS7. A criação da rede, no PCS7, já define o endereçamentos dos lógicos do PLC.

A decisão de qual tipo de rede a ser usada não faz parte do escopo deste trabalho, pois essa decisão foi feita por parte da empresa cliente.

1.3.2.3 Revisão e Correção das Telas do Supervisório

Deve-se revisar e corrigir a versão preliminar das telas dos supervisórios, bem como restrutura-las de uma forma mais eficiente e simples.

1.3.2.4 Programação das Lógicas de Controle

A programação da lógica de controle consiste em programar a unidade lógica de controle afim de possibilitar o uso das diferentes rotas do produto, bem como os intertravamentos de segurança.

1.3.2.5 Simulação do Processo para Validação das Lógicas de Controle

Talvez uma das tarefas mais trabalhosa seja, a simulação da planta já que dentro da empresa GreyLogix é algo novo, e os softwares de simulação utilizados pela empresa não possuem bibliotecas prontas para esse tipo de processo. Essa simulação não deve ser muito aprofundada, devendo simular apenas as dinâmicas relevantes do ponto de vista da automação.

1.4 Metodologia

A metodologia do trabalho consiste em adquirir informações sobre o processo que irá ser automatizado e fazer a automação e simulação do mesmo. Cada uma dessas etapas, revisão técnica, programação das lógicas de controle e programação da simulação, usa um *software* diferente: o COMOS, PCS7 e o SIMIT, respectivamente. Esses por sua vez serão detalhados no capítulo 3 onde é explicado essas ferramentas de engenharia. Observando a Figura 1, nota-se que é possível dividir a metologia em três partes, cada uma relacionada a um *software*.

Primeiramente a revisão dos diagramas P&ID será feito através do estudo dos documentos de processo mais atualizados da planta fornecidos pelas empresas colaboradoras do projeto. Sendo assim, serão utilizados essas informações para fazer a revisão dos diagramas dentro do COMOS.

Em seguida deve-se criar elementos base de programação, os chamados Control Module Types (CMT's), dentro do PCS7 ("criação da biblioteca do projeto" Figura 1). Esses, por sua vez, servirão para a criação automatizada da base do *software*. A automatização ocorre pela comunicação entre o PCS7 e COMOS, sendo assim é imprescindível que a base das informações do COMOS esteja atualizada.

Assim ter-se-a a base da programação do sistema de controle no software PCS7. Dessa forma deve-se usar essa base da programação gerada automaticamente e os descritivos de automação do processo para criar o *software* da planta. A criação desse *software* também engloba a estruturação das redes de comunicação interna da planta ("Montagem do *hardware*" e "Declaração da *Sybol Table*" Figura 1), que servirão para a unidade lógica de controle saber em quais das suas portas de entrada e saída estão os diferentes equipamentos de campo. Além disso será feita a revisão das telas de operação.

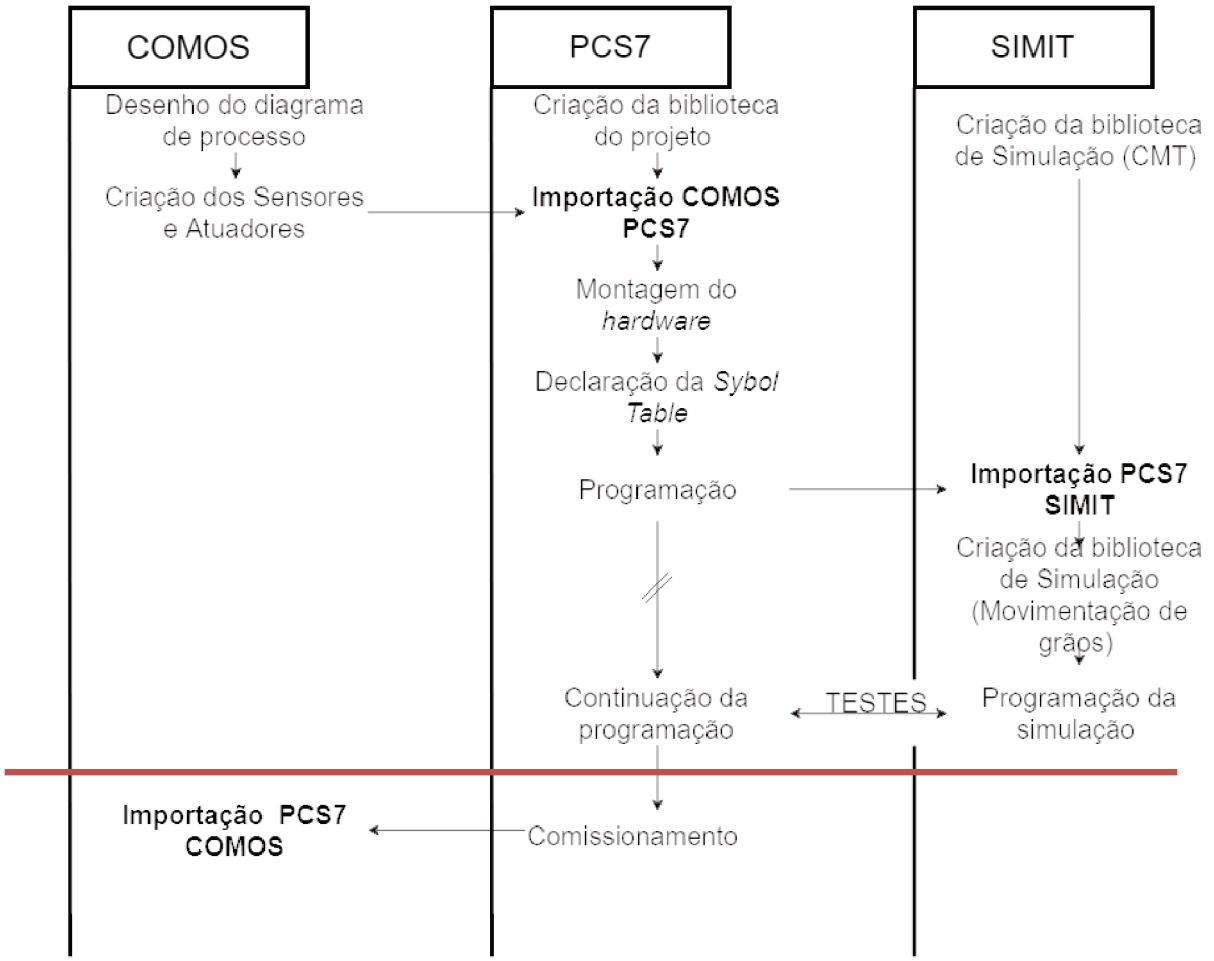


Figura 1 – Metodologia do projeto. Fonte: Original.

Por fim a última etapa é a programação da simulação no *software* SIMIT, que da mesma forma que o *software* PCS7 se relaciona com o COMOS o SIMIT se relaciona com o PCS7. Em outras palavras cria-se elementos base de programação no SIMIT e usa artifícios de importação para gerar a base da simulação. Essa base depois poderá ser complementada com a programação da simulação mais geral da planta. Essa última etapa também contempla a utilização da simulação para validar a programação do software ("Testes" Figura 1).

Além disso mostrou-se necessário criar uma biblioteca para os equipamentos presentes na planta que será automatizada. Vale ressaltar, que o comissionamento e a importação final da documentação do projeto não fazem parte do escopo deste trabalho, abaixo da linha vermelha na Figura 1.

No caso da revisão das telas usa-se uma diferente metodologia, onde a tela é projetada e ou revisada e enviada para a aprovação do cliente. Dessa forma, há um constante trabalho de refino das mesmas.

Um outro ponto geral, que deve ser abordado sobre a metodologia utilizada, é que os documentos, diagramas e informações da planta que está sendo automatizada estão foram

produzidos em paralelo com a execução desse trabalho. Sendo assim, poderá ser necessário ter que corrigir algumas partes consideradas concluídas. Isso também pode gerar algum impedimento de avanço, já que pode ocorrer a necessidade de adquirir informações que ainda não existem.

1.5 Trabalhos Relacionados

Esse trabalho tem dois principais trabalhos relacionados. O primeiro intitulado "Implementação de um sistema de controle distribuído em uma linha de produção de margarina" de Roger Perin [19] e o segundo de Cintia Sabrina Weirich e Luís Henrique Slongo "Modelo de Controle para o Processo de Beneficiamento de Sementes Baseado Em Eventos Discretos e Teoria de Controle Supervisório" [28].

No primeiro trabalho, de R. Perin, feito dentro das mesmas condições: a empresa GreyLogix Brasil prestando serviço para a COAMO, porém para a unidade de produção de margarina de Campo Mourão/PR. Além disso esse trabalho também se assemelha no uso das ferramentas de engenharias, nele foram utilizadas as ferramentas COMOS, PCS7 e SIMIT. Porém a programação da simulação do SIMIT não fazia parte do escopo do trabalho.

Já no trabalho de C. S. Weirich e L. H. Slongo, destaca-se que a planta automatizada também é um entreposto de soja. Porém com um tamanho e uma complexidade menor da planta automatizada no presente trabalho.

Pode-se ainda citar as experiências em outros trabalhos de automação de entrepostos feitos pela GreyLogix Brasil. Esses foram essenciais para fazer troca de informações, e correções no trabalho como um todo. Assim ajudando a diminuir erros básicos, bem como a auxiliar em soluções de problemas já enfrentados nos outros projetos.

1.6 Estrutura e Organização do Trabalho

Esse trabalho está dividido em 5 capítulos. O presente Capítulo, apresenta a introdução do trabalho, onde é feito uma contextualização do problema que será abordado, e um possível caminho de solução. No Capítulo 2 é onde é mostrado a fundamentação teórica, ou seja, é onde será explicado os conceitos básicos usados neste projeto. Já no Capítulo 3 é onde é explicado as ferramentas de engenharia utilizadas. No Capítulo 4 são mostrados os métodos de implementação dos resultados e os resultados. Os trabalhos futuros e a conclusão serão abordados no Capítulo 5.

2 Fundamentação Teórica

Neste capítulo é abordada a fundamentação teórica usada no projeto. Primeiramente mostra-se uma visão geral do processo de extração de óleo de soja, aprofundando mais no setor operacional. Depois está sendo explicado a teoria básica dos diagramas de processos. Em seguida, os componentes básicos de hardware utilizados no projeto. A seção seguinte apresenta os conceitos básicos de redes industriais, bem como um resumo sobre cada um dos modelos utilizados.

É importante ressaltar que não será explicado a escolha dos equipamentos de hardware e dos protocolos de redes utilizadas, pois não estão no escopo do trabalho, visto que essas decisões já tinha sido tomadas pelos projetistas e clientes do projeto.

2.1 Extração de Óleo de Soja

Toda esta seção tem como base o descritivo interno da empresa, sobre a extração do óleo de soja [4].

Dentro do agronegócio da soja, uma das principais atividades é a extração do óleo, que pode ser tanto para uso em biocombustíveis, quanto em produtos alimentares. Além disso a extração de óleo de soja gera um segundo sub-produto, o farelo da soja, que normalmente é usado na produção de ração animal.

O processo de extração do óleo de soja pode ser dividido em três etapas: preparação, extração e refino. Após a colheita, a soja deve ser pré-limpa, secada e limpa, todas essas atividades fazem parte do setor operacional. Essas etapas podem ser feitas, total ou parcialmente, pelo produtor. Na empresa, que irá beneficiar a soja, deve-se realizar as etapas restantes, caso haja, e assim armazenar ou usar os grãos na linha de produção. Por isso existe o já apresentado problema das rotas do setor operacional. Esse setor será mais aprofundado na seção seguinte.

Continuando o fluxo do processo de extração, temos que, na preparação o primeiro passo é a quebra do grão, que ocorre em uma máquina chamada moinho de rolo. Em seguida a soja é aquecida com vapor, a fim de matar micro-organismos e desnaturar enzimas. O próximo passo é a compressão da soja quebrada, na máquina de laminação, essa compressão leva o grão a ter uma espessura bem fina, o estado da soja na saída deste equipamento pode ser observada na Figura 2.

O próximo equipamento da linha da preparação é o *expander*, que expande o grão laminado, colocando em condição de alta temperatura e pressão. Assim, na saída deste equipamento ocorre uma redução brusca na pressão, que faz a água interna evaporar e expandir a soja laminada, Figura 3. Essa soja expandida é então resfriada e enviada para

Figura 2 – Soja laminada. Fonte: Bianchini-Produtos S/A [1].

o setor de extração propriamente dito.

Figura 3 – Soja Expandida. Fonte: Bianchini-Produtos S/A [1].

A extração do óleo da soja ocorre através da aplicação de um solvente na soja, tanto expandida quanto laminada, o solvente utilizado é o hexano. A soja banhada em solvente tem grande parte do seu óleo removido, que, por sua vez, fica misturado juntamente com o hexano.

A soja carregada de solvente então é aquecida com vapor, a fim de extrair o resto do óleo de dentro do farelo. Esse processo ocorre dentro de um equipamento destilador chamado dessolventizador tostador (DT) e os produtos de saída são a miscela rica em óleo, farelo branco e o hexano, que pode ser recuperado para ser reutilizado no processo.

Na próxima etapa, a evaporação, é onde o hexano será extraído da mistura de óleo. Neste processo a mistura é aquecida até temperaturas nas quais não comprometam a qualidade do óleo. Esse aquecimento tem a função de remover boa parte do solvente. A remoção do resto do hexano é feita através de um processo de arraste, utilizando vapor d'água.

Por fim, a mistura, já considerada óleo bruto, passa pelas seções de refino. Inicialmente, em um processo chamado degomagem, que basicamente adiciona água quente ao produto e o centrifuga, afim de retirar as impurezas como fosfatídeos, proteínas e subs-

tâncias coloidais. Em seguida, o óleo passa por um processo de neutralização que remove os ácidos graxos. Após isso, é removido as ceras do óleo através de um processo chamado de Winterização. E por fim o óleo é branqueado e desodorizado. Esses processos são responsáveis por remover os pigmentos e odores indesejados.

A continuação do processo é o engarrafamento do óleo. As garrafas posteriormente são embaladas. Todas essas etapas podem ser visualizadas no diagrama da Figura 4.

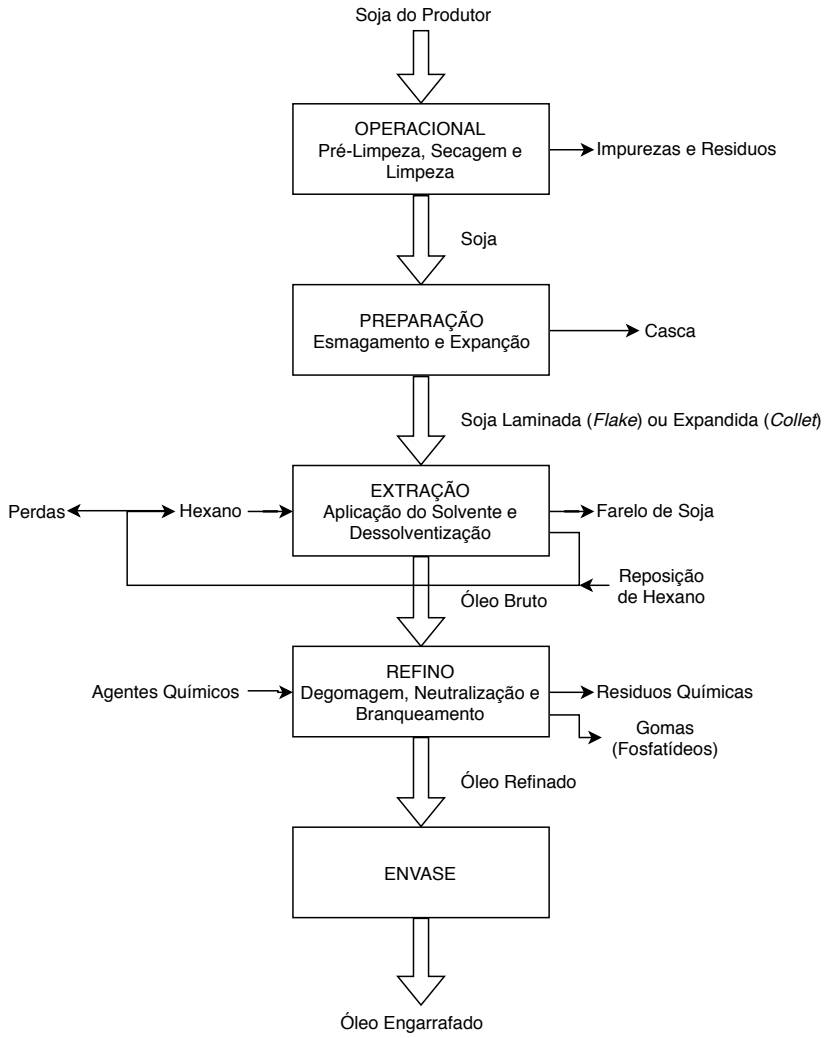


Figura 4 – Diagrama processo de extração de óleo de soja. Fonte: Original

2.1.1 Setor Operacional

O setor operacional de uma indústria de óleo de soja também é chamado de entreposto. O entreposto, por sua vez, é a parte do processo que engloba desde o recebimento do grão da soja misturada com impurezas, provenientes da colheita, até o envio para a preparação da extração do óleo. O processo consiste em passar o produto em etapas de pré-limpeza, limpeza e secagem do grão, para que este fique sem impurezas e em condições ideais para o processo de preparação.

A planta do operacional também pode ser usada para o armazenamento de outros tipos grãos, como, por exemplo, milho e arroz. Porém, o presente trabalho limita-se à soja.

O processo do setor operacional pode ser dividido em: Moegas, Pré-limpeza, Secagem, Limpeza e Armazenagem.

2.1.1.1 Moegas

A parte das Moegas podem ser entendidas com grandes funis, onde o grão e descarregado dos caminhões, através de tombadores. No fundo das Moegas encontram-se válvulas que tem por objetivo habilitar, ou não, a descarga das moegas em esteiras, localizadas logo abaixo Figura 5.

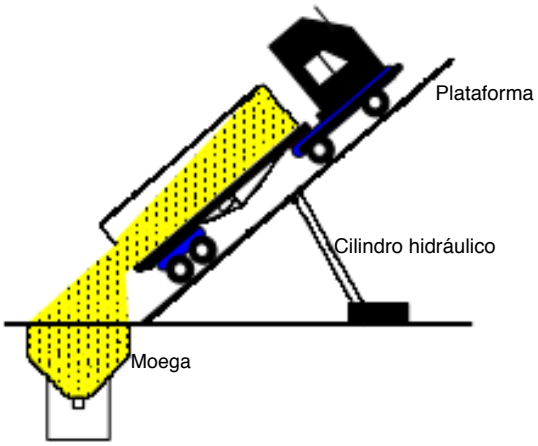


Figura 5 – Esquema do processo de descarga da soja. Fonte: Estrutura para armazenagem de grãos a granel [2].

2.1.1.2 Pré-limpeza e Limpeza

Se o grão que vem do produtor não foi pré-separado das impurezas é necessário que esse produto passe por uma etapa de pré-limpeza. A máquina responsável por tal tarefa e pela etapa de limpeza são compostas de peneiras que podem ser tanto rotativas quanto por vibração, Figura 6 e 7 respectivamente.

A etapa de limpeza é onde mais uma vez é feita a separação de resíduos e impurezas. As impurezas são galhos, pedaços de metal, terra entre outros. E os resíduos são as vagens de soja, que, posteriormente, são abertas em uma máquina chamada de trilhadeira (Figura 8). Vale destacar que as impurezas que saem dessas máquinas também podem ser utilizadas em outras etapas do processo de extração de óleo de soja.

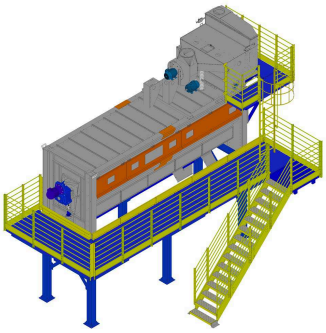


Figura 6 – Maquina de limpeza rotativa. Fonte: Carpan [3].

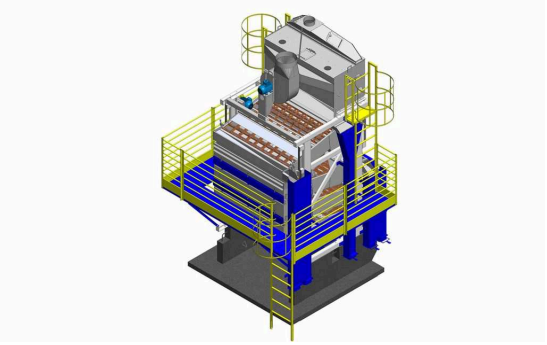


Figura 7 – Maquina de limpeza vibratória. Fonte: Carpan [3].

Figura 8 – Trilhadeira de soja. Fonte: Carpan [3].

2.1.1.3 Secagem

O secador de grãos faz a secagem dos grãos de soja ainda úmidos. Esse tem por objetivo reduzir a umidade do grão de, aproximadamente, 14% para 10% [4]. Esse processo ocorre com o aquecimento do grão a cerca de $65°C$, para isso os grãos são colocados em calhas que facilitam a troca térmica. Na parte de baixo do secador, existe uma etapa de refrigeração, porém o grão só volta à temperatura ambiente dentro dos silos. A Figura 9 demonstra esse funcionamento.

O aquecimento da soja ainda ajuda a reduzir o número de contaminantes biológicos na soja, além de desnaturar proteínas, facilitando assim a retirada do óleo. O ar aquecido, necessário para a secagem, vem de uma fornalha, posicionadas ao lado secador. Essa fornalha pode aquecer com a queima de lenha, cavaco de madeira, ou até com a queima de gases.

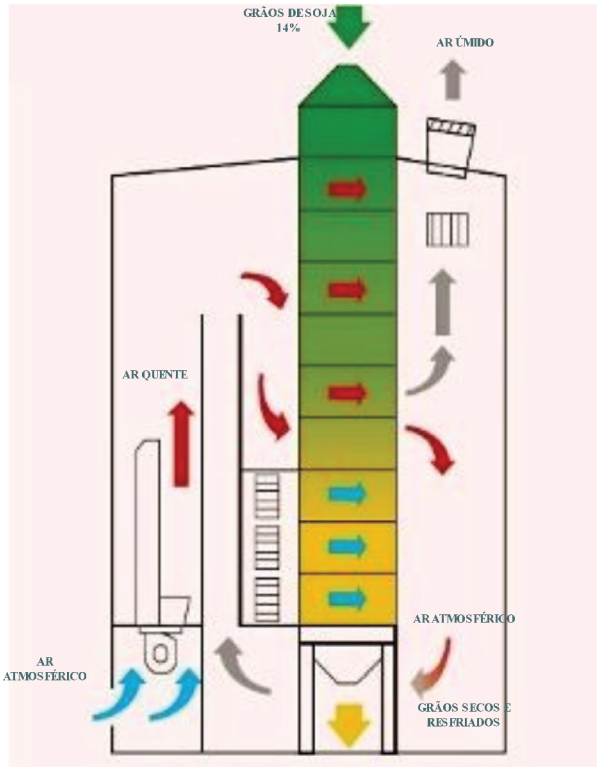


Figura 9 – Esquema do processo de aquecimento e secagem dos grãos. Fonte: Processo de Extração e Refino de Óleo de Soja [4].

2.1.1.4 Armazenagem

A armazenagem dos grãos ocorrem em grandes silos que, geralmente, possuem uma estrutura cilíndrica (Figura 10).

Figura 10 – Silo plano. Fonte: Kepler Weber [5].

Esse e outros modelos de silo de grãos devem possuir um sistema de aeração, que tem por objetivo manter o grão seco e resfria-lo. O silo da Fig. 10 possuí, também, um sistema de rosca varredora, que move o grão das extremidades do silo para o centro. Essa funcionalidade tem por objetivo fazer com que o processo de descarga seja mais linear, já que o processo de descarga desse e de outros silos graneleiros pode ocorrer de maneira desuniforme, onde o centro do silo esvazia antes que as extremidades (Figura 11). Assim,

caso ocorra uma descarga muito desuniforme, deixando um lado do silo com mais soja que o outro, pode causar o colapso da estrutura do silo.

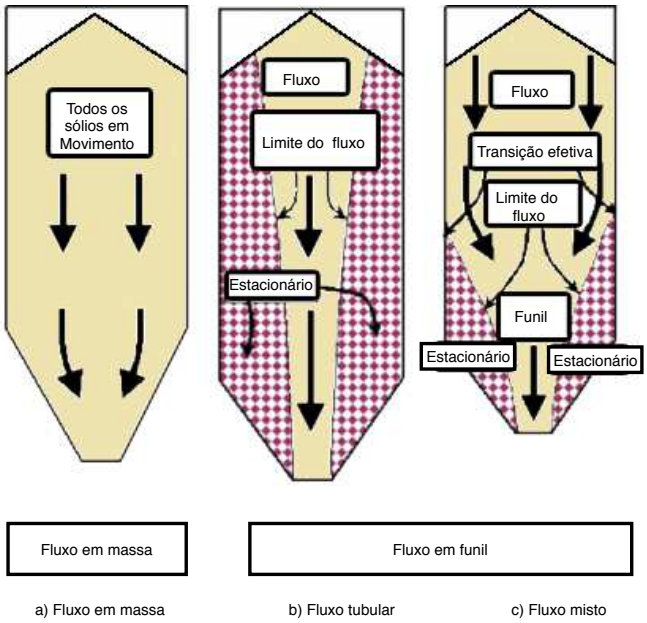


Figura 11 – Modos de descarga dos silos. Fonte: *Silo Pressure Mesuriments* [6].

2.1.1.5 Transportadores

Os dois principais mecanismos de transporte de grãos são os elevadores caçamba (Figura 12) e as esteiras transportadoras. Caso algum deles receba muito produto, pode ocorrer o fenômeno de embuchamento. Nesse, problema a esteira recebe mais produto do que ela pode carregar. Outros problemas comuns a esses equipamentos é o levantamento de pó dos grãos, que são altamente inflamáveis, para isso, sensores de desalinhamento das fitas transportadoras e elevadores, ajudam a a indicar que o elevador não está gerando algum tipo de calor por atrito contra a carcaça do equipamento.

2.1.1.6 Válvulas Seletoras

Os equipamentos que definem os caminhos de descarga desses equipamentos de transporte são as chamadas válvulas bifurcadas e trifurcadas, elas fazem a seleção de qual dos caminhos o elevador ou esteira irá mandar o produto. É por meio desses que será feito o controle das rotas. Essas válvulas podem ser vistas nas Figuras 13 e 14.

2.1.1.7 Requisitos de Automação

O setor operacional tem alguns requisitos de automação, que são:

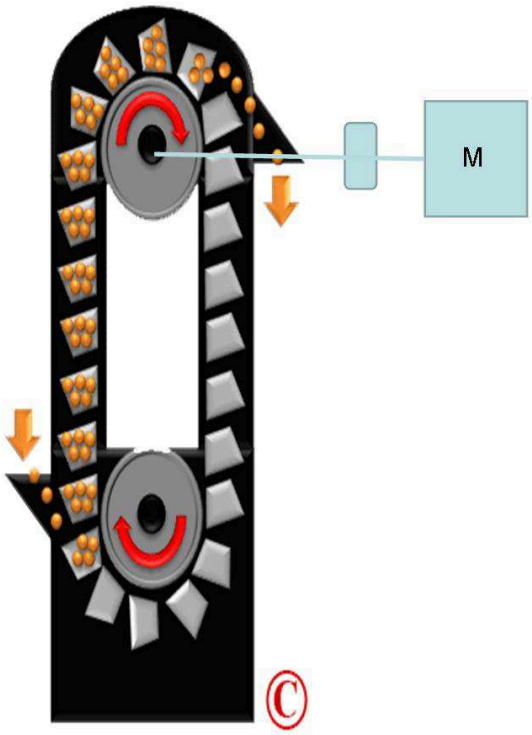


Figura 12 – Diagrama de funcionamento do elevador caçamba. Fonte: *Mechanic Info* [7].

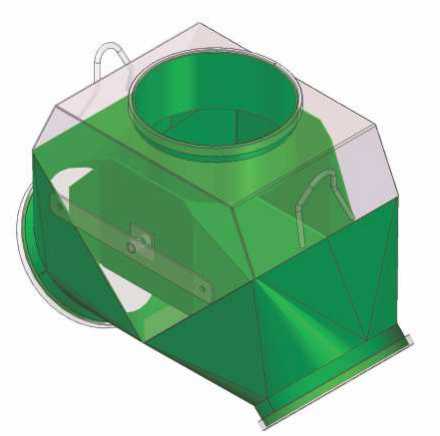


Figura 13 – Válvula bifurcada. Fonte: Agriaço [8].

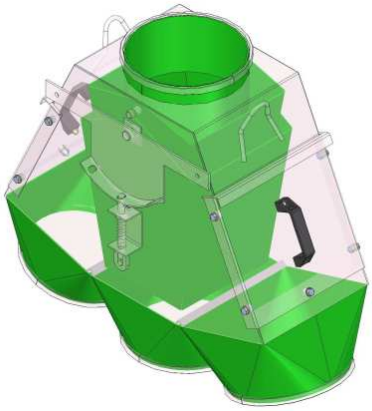


Figura 14 – Válvula trifurcada. Fonte: Agriaço [8].

- Os transportadores devem ser ligados do seguindo a ordem reversa da rota, ou seja, do final da rota para o início, dessa forma, o equipamento que irá receber produto estará sempre ligado;
- Caso algum transportador ou equipamento pare de funcionar, deve ser desligado os equipamentos anteriores em cascata, assim não haverá envio de produto para os equipamentos desligados;
- A troca das posições das válvulas direcionais só pode ser feita caso os equipamentos

de transporte estejam desligados. Isso deve ser feito afim de evitar que a soja se acumule na válvula, impedindo entupimentos e quebra dos mecanismos de acionamento das válvulas direcionais.

2.2 Diagrama de Tubos e Instrumentos (P&ID)

Os diagramas de tubulação e instrumentação, ou do inglês *pipes and instrumentation diagrams* P&ID, são documentos que mostram toda a cadeia produtiva do ponto de vista da engenharia de processos. Para isso, esses diagramas tem como objetivo representar os diferentes sensores, atuadores e como eles estão dispostos dentro das malhas de controle. Além disso, esses diagramas mostram os diferentes equipamentos do processo de forma simplificada.

Essas representações usualmente seguem normas pré-estabelecidas. As quais destacam-se:

- ANSI (American National Standard Institute)
- API (American Petroleum Institute)
- ASME (American Society of Mechanical Engineers)
- ASTM (American Society for Testing & Materials)
- BSI (British Standards Institution)
- ISA (International Society for Measurement & Control)
- ISO (International Standard Organization)
- DIN (Deutsches Institut für Normung)
- DNV (Det Norske Veritas) & BV (Bureau Veritas)
- JIS (Japanese Industrial Standards)

No presente trabalho foi usada a norma da ISA [9], seguindo o padrão nacional e as exigências do cliente.

A representação de um processo em um P&ID tem por objetivo ser uma representação simplificada de todo o processo em questão. Para isso algumas regras foram convencionadas, principalmente em relação ao desenho dos equipamentos e de como eles devem ser nomeados.

A regra que define como que devem ser desenhados os elementos (Típicos) nos diagramas separa os diferentes equipamentos padronizando como eles devem ser representados. Nas Figuras 15, 16 e 17 são mostrados, partes da norma que indicam como deve ser desenhados os elementos das categorias de válvulas, atuadores e tipos de sinais, respectivamente.

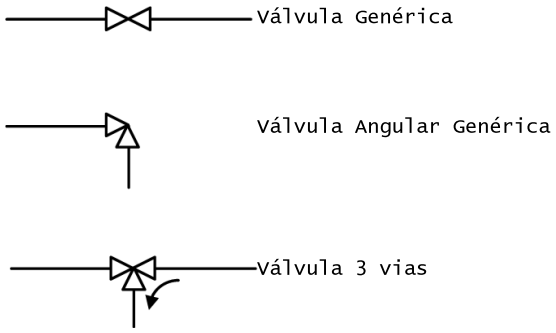


Figura 15 – Típico válvula P&ID ISA. Fonte: ISA 5-1 [9].

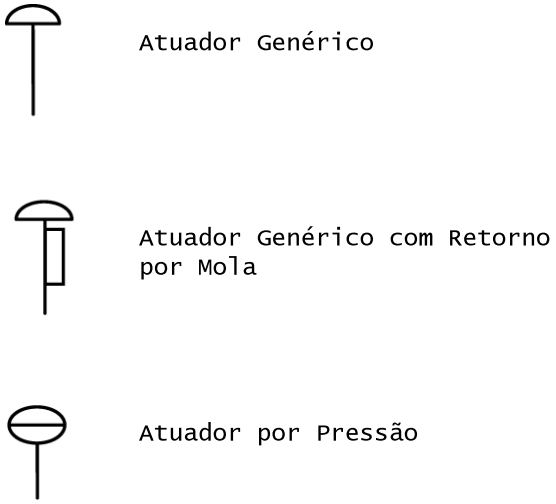


Figura 16 – Típico de atuador P&ID ISA. Fonte: ISA 5-1 [9].

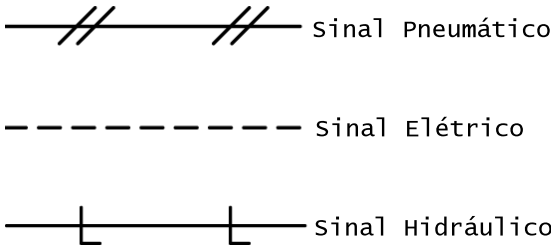


Figura 17 – Tipos de sinais P&ID ISA. Fonte: ISA 5-1 [9].

Além desses elementos, exemplificados nas Figuras 15, 16 e 17, existem modelos para os desenhos de indicadores, controladores, tanques e outros equipamentos relevantes da indústria de processo. Esses podem ser vistos nas páginas das normas [9]

Uma outra regra que merece destaque é a das *TAG's*, que estabelece que cada componente deve ter um nome único [9]. Além disso, as *TAG's* seguem regras de construção que conseguem indicar a função e a localização, no ponto de vista de processo, dos diferentes elementos. Na Figura 18, pode-se observar um exemplo de um de P&ID, este segue a norma ISA.

Analisando a Figura 18 pode-se inferir que:

- O tanque representado tem o seu nível medido pelo transmissor de nível `LT01`;

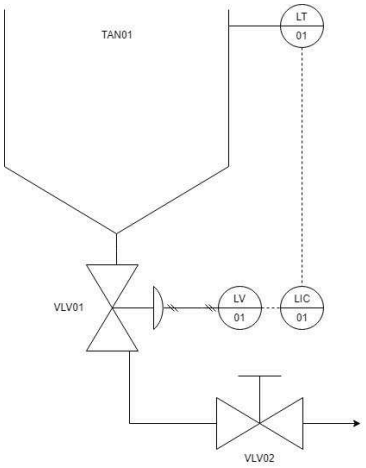


Figura 18 – Exemplo de P&ID. Fonte: Original.

- O sinal do transmissor é enviado eletricamente para o controlador `LIC01`, que por sua vez envia o sinal de atuação para o acionamento `LV01`;
- A válvula VLV01 é acionada pneumaticamente pelo atuador da válvula;
- A válvula VLV02 é uma válvula manual.

Assim, pode-se concluir que a vasão da válvula VLV01 é o que controla o nível do tanque. E que, além disso, existe também uma válvula manual para para interromper o fluxo de saída do tanque. É importante ressaltar que os desenhos dos equipamentos e o nome dos elementos é padronizado por cada uma das normas. No caso da Figura 18 foi feito seguindo a ISA.

2.3 *Hardware* de Automação e Sistemas de Controle Distribuídos (DCS)

Uma importante parte que deve ser compreendida para ter-se o entendimento do trabalho são os equipamentos e como eles são utilizados em sistemas modernos de automação de processos.

Tomando como base a Figura 19, onde é possível observar a estrutura mais comum de um sistema de automação a chamada estrutura de sistemas de controle distribuído, ou do inglês *distributed control system* (DCS) [29].

Dessa forma, na Figura 19 pode-se destacar os seguintes elementos de *hardware*: os Controladores Lógico Programáveis (CLP's), os Computadores e os hardwares de supervisório. É importante comentar que as redes de comunicação serão abordada na Seção 2.4. Para essa seção basta saber que essas redes servem para fazer a troca de informações entre as diferentes maquinas conectadas na rede.

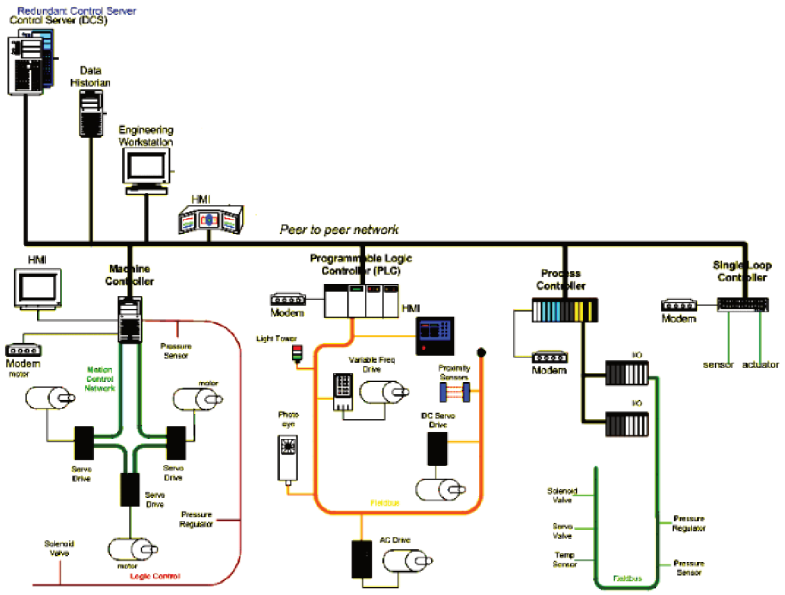


Figura 19 – Estrutura Distribuída de controle industrial. Fonte: Remote monitoring system for distributed control of industrial plant process [10].

2.3.1 Controladores Lógicos Programáveis CLP

Os controladores lógicos programáveis são computadores desenvolvidos para o ambiente industrial, com interfaces de entrada e saída e uma linguagem de programação para desenvolver a lógica de controle. Os CLP's devem ser robustos o suficiente para suportar os intempéries do ambiente industrial, como ruídos eletromagnéticos, poeira, entre outros. A arquitetura do processador de um CLP é feito para que ele possa executar uma tarefa, no caso a lógica de controle. Além disso, essas máquinas devem atender requisitos de tempo real, esses requisitos podem ser entendidos como garantir uma resposta dentro de um intervalo de tempo pré-definido, em outras palavras uma previsibilidade dos tempos de resposta [29] [30].

Os CLP's podem ser usados sozinhos para controlar máquinas, equipamentos ou até mesmo plantas de lógicas um pouco mais simples e que funcionam totalmente em automático. Essas aplicações, somente com CLP's, normalmente são feitas com uma abordagem *bottom up*. Normalmente essas aplicações necessitam de respostas mais rápidas. Sendo assim, outro ponto que pode-se levar em conta é que as abordagens só com CLP's tendem a ter dinâmicas mais rápidas, fazendo com que o tempo de resposta seja, algo entorno de, $10ms$ [31].

2.3.2 Sistemas de Controle Distribuídos (DCS)

Em contra ponto ao CLP, os DCS são sistemas distribuídos, como o próprio nome já diz. Isso faz com que o processamento do código de controle seja feito em todas as

unidades de processamento da rede, isso inclui CLP's e PC's.

Nas abordagens de automação usando essas técnicas, é mais comum usar a metodologia *top-down*. Dessa forma, usa-se linguagens de programação de mais alto nível com SFC, diagrama de blocos e CFC (derivado do diagrama de blocos). Devido a isso, os sistemas DCS usam bibliotecas prontas, ou seja, o controle de baixo nível dos equipamentos vem de dentro das bibliotecas.

Assim o operador desempenha um papel mais ativo, já que ele é responsável por tomar decisões de processo, como no caso desse projeto, definir qual rota o grão irá passar e ativar manobras de descarga dos silos. Deve-se notar também que o tempo de resposta desses sistemas é um pouco mais lenta, cerca de 100ms [31].

2.3.3 Linguagens de Programação de Sistemas de Automação

A padronização das linguagens de programação utilizadas em CLPs é definida pela norma IEC 61131-3 [20]. Esta norma foi criada com o objetivo de padronizar a programação entre CLPs de diferentes fabricantes, pois mesmo utilizando a mesma linguagem, os programas não podiam ser portados de maneira simples para CLPs de fabricantes diferentes. Alguns problemas provenientes dessa falta de padronização podem ser citados [32]:

- Diferenças entre símbolos e capacidades de programação variam de um CLP para outro, obrigando o programador a reaprender a codificar em uma mesma linguagem.
- A falta de suporte para a criação de funções, com entradas e saídas, que facilitariam a reutilização de código.
- A inexistência de variáveis locais obrigava o programador a ter um cuidado maior na separação da memória do CLP para cada segmento do programa.
- Na maioria dos CLPs não existia a possibilidade de modificar o ciclo de execução de cada programa separadamente, sendo assim o ciclo de execução iria depender apenas do tempo que o programa leva para ser executado. Isso pode ser um problema especialmente para controles PID (Proporcional Integral Derivativo), que se beneficiam de um tempo de execução fixo.

Para resolver isso a padronização trouxe um conjunto de sugestões para os fabricantes, que podem escolher ou não segui-las. Entre essa sugestões está a definição e padronização das cinco linguagens padrões definidas na IEC 61131-3, sendo elas [32]:

- **Diagrama *ladder*** (*Ladder diagram* - LT)
- **Texto estruturado** (*Structured Text* - ST)

- **Lista de instruções** (*Instruction List* - IL)
- **Diagrama de funções sequenciais** (*Sequential Function Chart* - SFC)
- **Diagrama de blocos** (*Function Block Diagram* - FBD)

Nas Seções seguintes serão descritos as duas linguagens de programação usadas nesse projeto, o Diagrama de funções sequenciais SFC e Diagrama de blocos FBD.

2.3.3.1 Diagrama de Funções Sequenciais-SFC

A linguagem de programação SFC é uma linguagem baseada na metodologia das redes de Petri. Esta linguagem é especialmente indicada para processos que possam ser divididos em uma sequência de etapas bem definidas.

A linguagem é composta de três elementos básicos as etapas, transições e ações. Cada etapa está ligada a outra através de uma transição, e cada etapa contém uma ou várias ações. Cada etapa é sempre intercalada por uma transição. Pela norma, nunca se deve ter duas etapas ou duas transições seguidas. Porém, uma etapa pode levar a bifurcações de transições, assim como uma transição pode levar a uma sequência paralela ocorrendo [33] [20].

De maneira geral, um programa em SFC tem a seguinte aparência (Figura 20):

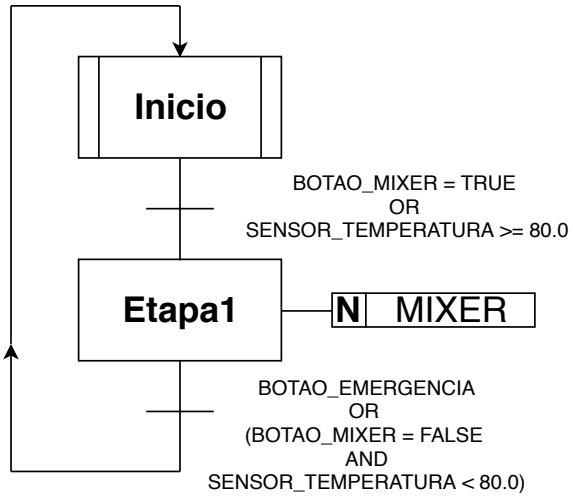


Figura 20 – Exemplo de código em SFC. Fonte: Original.

Na Figura 20, é interessante notar que as transições são definidas como condições. Essas condições quando atendidas habilitam a troca de etapas. Nas etapas por sua vez, existem as diferentes ações conectadas, essas ações são as que ocorrem durante a execução da etapa [20].

O caracter N na frente da etapa indica o tipo de ação que está sendo executada. Na Tabela 1 pode-se ver todas os tipos de ações previstas pela norma IEC 61131-3 [20].

Tabela 1 – Qualificadores das ações da linguagem SFC [20]

Qualificador	Descrição
Nenhum	Não armazena
N	Não armazena
R	*Reset*
S	*Set*
L	Limitado por tempo
D	Atrasado por tempo
P	Pulso
SD	Armazena e atrasa
DS	Atrasa e armazena
SL	Armazena e limitado por tempo
P1	Pulso (Borda de subida)
P0	Pulso (Borda de descida)

2.3.3.2 Diagrama de Blocos-FBD

O diagrama de blocos ou *Function Block Diagram* (FBD) segue o fluxo de execução da esquerda para direita e de cima para baixo. E toda a programação é feita através de blocos, onde cada bloco pode ter várias entradas e várias saídas. As entradas do bloco são posicionadas a esquerda, e as saídas a direita. Na Figura 21 mostra um exemplo de código em FBD [20] [32].

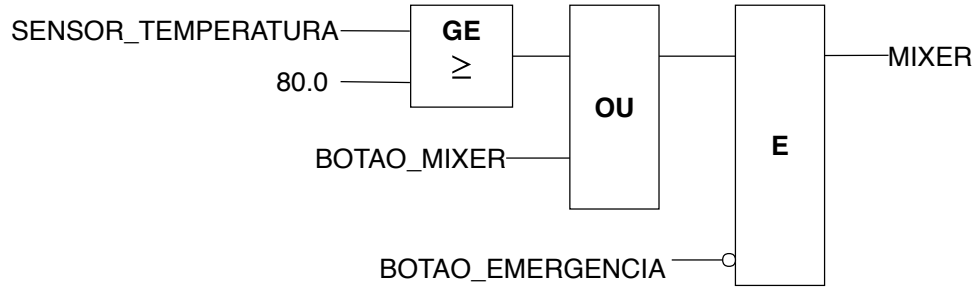


Figura 21 – Exemplo de código em FBD. Fonte: Original.

A linguagem FBD permite a existência de vários tipos de sinais numérico, como inteiros e *floats*, sinais de texto (sinais que mandam texto para outros blocos) e sinais *booleanos*. A execução dos códigos nessa linguagem inicia-se com a leitura de todos as entradas de todos os blocos, em seguida é calculada as saídas e por fim, são aplicados os sinais nas saídas dos blocos [32].

2.3.3.3 *Continuous Function Chart* (CFC)

O *Continuous Function Chart* ou CFC é uma linguagem de programação que deriva do FBD. As principais diferenças estão em poder selecionar a ordem de execução dos blocos e a complexidade dos mesmos. Por exemplo, nas bibliotecas utilizadas nesse projeto, tem-se blocos que já fazem lógica de acionamento dos motores, recebendo sinais de

intertravamento, acionamento manual e automático, detecção de erro interna aos blocos e afins. Na Figura 22 pode-se ver um bloco de um motor e suas configurações internas.

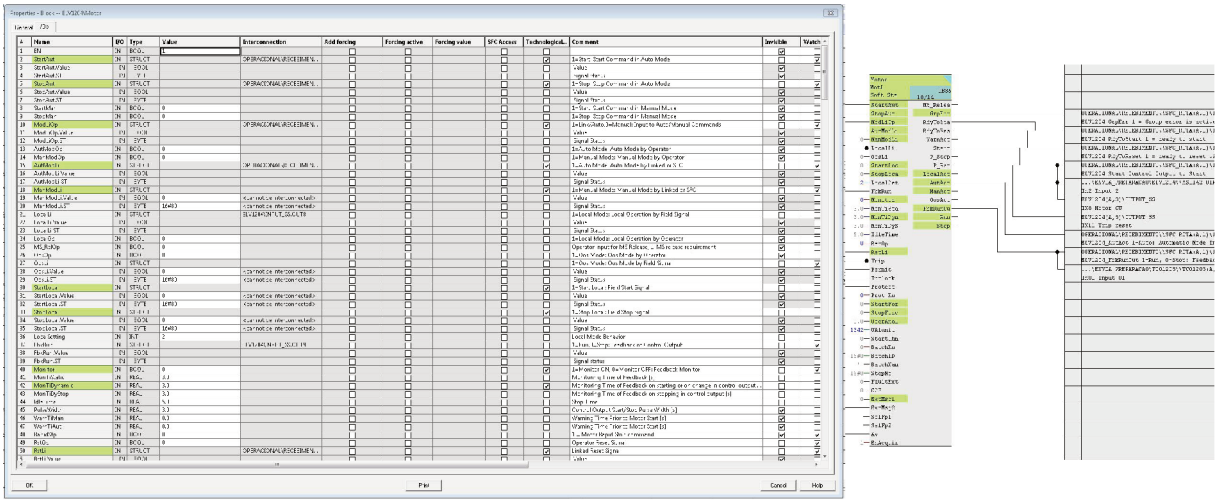


Figura 22 – Bloco de Motor em CFC. Fonte: Original.

Já um *chart* nessa linguagem é um conjunto de páginas onde tem-se os diversos blocos de um equipamento (Figura 23). Um programa em CFC é composto por diversos *charts* ou CFC's, que por sua vez são formados de diversos blocos. Sobre os blocos, vale ressaltar que, a lógica que está programado dentro deles é feita com linguagens de programação de baixo nível, normalmente Texto Estruturado.

Como dito, é possível configurar da ordem de execução dos blocos. Na Figura 24, é observado que todos os *chart's* são executados com o tempo de amostragem de 100 ms, e dentro de cada *chart* existem alguns blocos. Já a sequência de execução dos blocos segue a ordem apresentada dentro da pasta. Também, é possível saber a posição do bloco na ordem de execução observando o número que aparece dentro do retângulo azul claro na Figura 22, onde está escrito "22/11" leia-se 11º bloco a ser executado do CFC que será executado por 22º .

2.4 Redes de Computadores

As redes de computadores tem por objetivo fazer a comunicação entre duas máquinas eletrônicas. Essa comunicação deve ser entendida como a troca de informações.

Segundo Tanembau e Wetherall [34], uma rede de computadores pode ser entendida como uma um sistema onde não depende do *software*, como por exemplo a internet, onde diferentes computadores com diferentes *softwares* (Sistemas operacionais), se comunicam e trocam informações. Por outro lado, para eles um sistema distribuído é considerado um software montado encima de uma rede, ou seja, do ponto de vista do usuário seria como se a rede toda fosse apenas uma máquina [34].

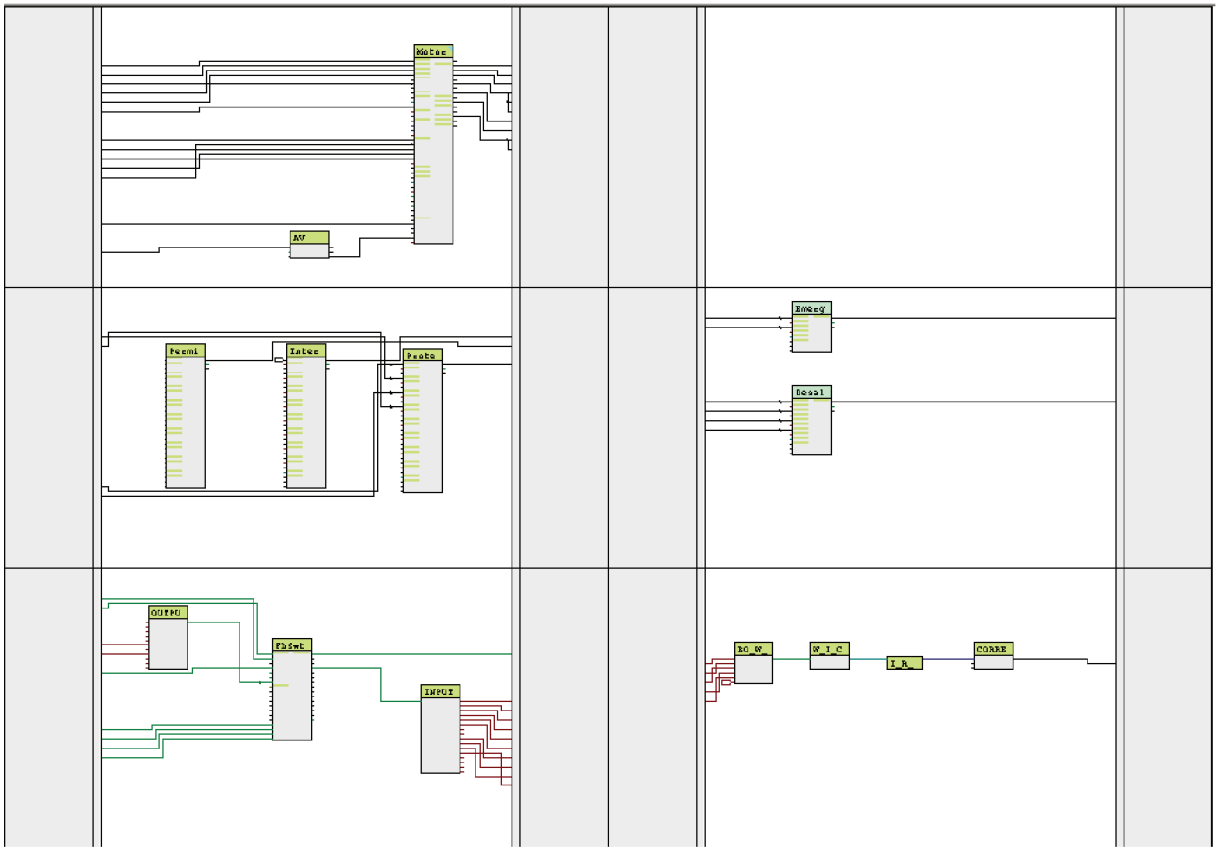


Figura 23 – *Chart* CFC. Fonte: Original.

Independentemente da classificação, rede ou sistema distribuído, pode-se usar o modelo OSI (Open System Interconnection) para entender como a rede funciona. O modelo OSI divide a comunicação entre máquinas em 7 camadas, onde cada uma das camadas tem funções únicas e bem definidas (Tanembau e Wetherall) [34]:

1. Física: é o canal de comunicação propriamente dito, onde existe o tráfego de bits. Sua função é garantir que o que é transmitido da máquina 1 é recebido pela máquina 2;

2. Enlace: divide a palavra de comunicação em *frames*, além de detectar os erros provenientes da camada física. No receptor essa camada tem por objetivo mandar a confirmação de recebimento;

3. Rede: define o caminho da comunicação, de forma a controlar quais são os pontos nos quais os pacotes de dados irão passar. Além disso essa camada também faz correções de endereçamento entre as redes e sub-redes;

4. Transporte: define a quem será o receptor, ao contrário das camadas anteriores, essa camada é implementada em *Software*. Além disso essa é a camada que monta o pacote que será trafegado, sendo que ela divide a informação em provinda das camadas superiores e acrescenta um *header* de comunicação;

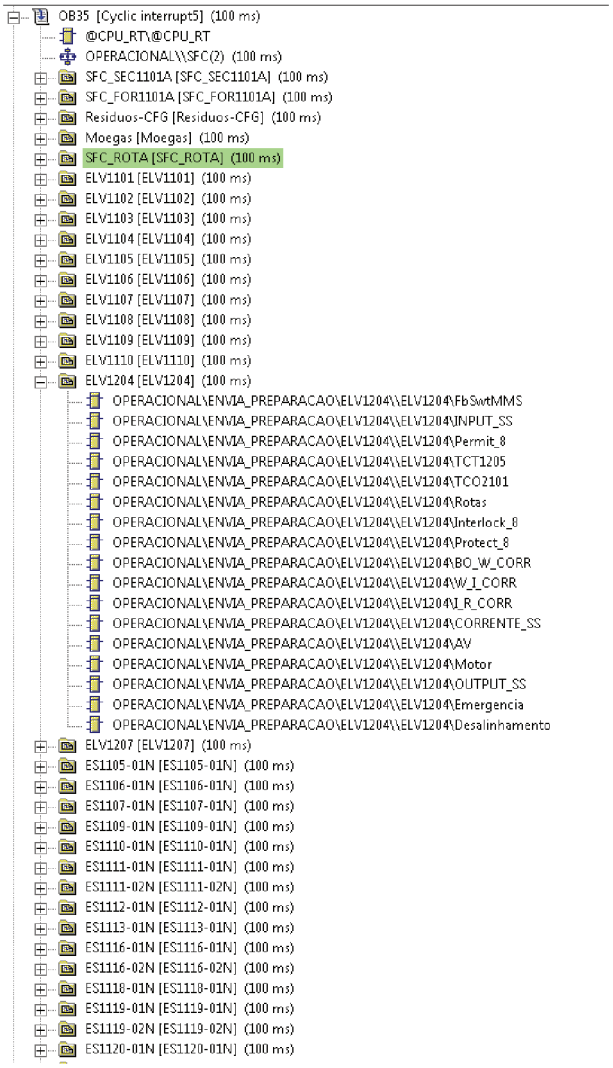


Figura 24 – configuração da ordem execução de cada CFC. Fonte: Original.

5. Seção: faz com que máquinas de diferentes sistemas operacionais se comuniquem. Ela também é responsável por fazer a sincronia da comunicação, dessa forma o meio de comunicação é usado somente por uma máquina por vez;

6. Apresentação: é onde são definidos a semântica e a sintaxe utilizadas na comunicação, dessa forma essa camada é a camada que interpreta a informação comunicada, e da mesma forma, à prepara para ser enviada;

7. Aplicação: por fim na Aplicação é onde está o *software* que está sendo diretamente utilizado. Por exemplo, é nessa camada é onde programa-se um *Web Site*.

Dependendo do protocolo, nem sempre se tem todas as camadas. Isso influencia a qualidade, complexidade e velocidade da rede.

2.4.1 Redes Industriais

Do ponto de vista industrial, existem diversos níveis de redes industriais, onde em cada nível temos diferentes protocolos de comunicação. Cada um desses protocolos atende a diferentes necessidades. Na Figura 25 pode-se ver a chamada pirâmide de redes industriais.

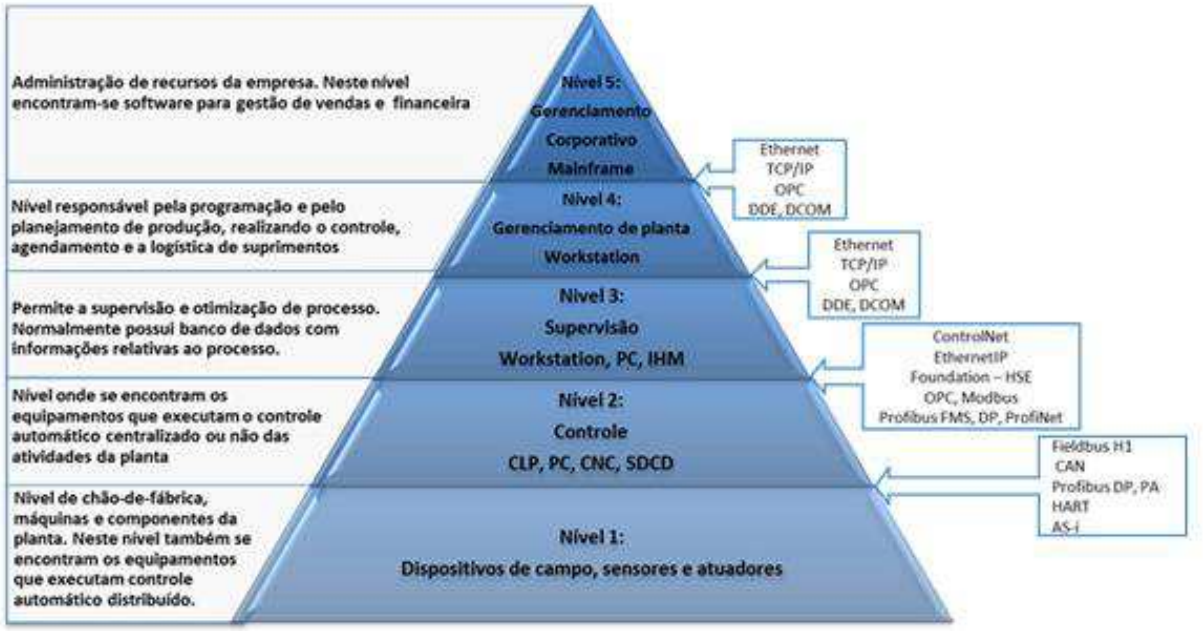


Figura 25 – Piramide dos níveis de automação. Fonte: *SMAR Industrial Networks - Part 1* [11].

Na pirâmide (Figura 25) pode-se notar que conforme aumenta o nível da rede, aumenta-se a complexidade da informação enviada, da mesma forma que incrementa o tempo de envio de cada pacote de informação. Ou seja, nas camadas mais baixas existe a informação "crua" dos sensores e atuadores. E no nível mais alto existe as informações de gerência da planta.

Da Figura 25, ainda destaca-se os protocolos de comunicação estão entre as camadas. Além disso sobre cada um dos níveis ressalta os seguintes tópicos:

- No nível do sensores e atuadores, tem-se uma frequência de comunicação maior, e uma rede mais confiável. As informações que trafegam desse nível para o nível acima tem normalmente o tamanho de poucos *bits* e *bytes* e tem uma frequência de comunicação alta, cerca de 1 a 20 ms;

- Os CLP's, no segundo nível, trocam informações com os Sensores e Atuadores (presentes no nível anterior) e o nível acima (Supervisório). Ele funciona, interpretando as operações pedidas pelo supervisório e as verifica e depois envia comando aos atuadores. As informações de monitoramento (sinais de sensores) são interpretadas e processadas para serem apresentadas no nível do supervisório. As informações tra-

fegadas nesse nível tem uma frequência de 100 ms e tamanho próximos aos *kilobytes* [11];

- O nível do supervisório é acessível aos operadores da planta. É nele que as decisões de processo são tomadas. Além disso esse nível deve se comunicar com o nível superior o de produção, informando relatórios de produtividade. Aqui o tempo de entre o tráfego das informações é entre 1 s a 1 min dependendo do tipo de informação, com o pacote de dados possuindo entorno de 1 *megabyte* [11];

- No nível de produção é de onde sai os comandos de produção de toda a fabrica, é com as informações desse nível que são tomadas as decisões de logística de produção. Já os tempos de entre os envios de informações chegam a questões de horas [35];

- No ultimo nível (gerência) é o nível mais alto, nele as informações trafegadas são os números das notas de vendas, quantidade de produtos vendidos e qual a direção que a produção deve tomar. Aqui os pacotes de dados podem ser entendidos até como os *e-mail* entre cliente e fornecedor.

Uma última informação relevante em relação a pirâmide de redes industriais ilustrado na Figura 25, é que ela se caracteriza como uma pirâmide, porque relaciona não só o nível da informação trafegada, mas também a quantidade dessa informação e também o número de dispositivos nela inserido.

Dentro desse contexto uma das principais vantagens do uso de redes na indústria, é que facilita na montagem e estruturação da parte física da planta, visto que ao invés de ter vários cabos, um para cada dispositivo de campo, tem-se alguns cabos que circulam toda a planta e não mais um cabo para cada dispositivo.

Visto a importância do uso de redes, é interessante detalhar os dois protocolos usados neste trabalho, o *Actuators and Sensors Interface* e o *PROFINET*.

2.4.1.1 *Actuator and Sensors Interface* (AS-Interface)

O protocolo Actuator and Sensors Interface (AS-Interface) é regulamentado pela norma $IEC62026-6$. Seguindo a ideia do modelo de referencia OSI ele é dividido em três camadas Física Enlace e Aplicação [14].

Na camada física destaca-se o cabo utilizado que tem um formato específico para facilitar a conexão de novos módulos, além de que a rede pode ser construída nas topologias árvore, estrela, linear e ramificada, desde que não ultrapasse $100m$ de extensão sem repetidor e $300m$ com dois repetidores [12]. A Figura 26 mostra o formato do cabo, já a Figura 27 demonstra como é feita a conexão dos módulos. Pela característica da cor amarela do cabo, as redes AS-I são representadas em diagramas de rede, normalmente, na cor amarela.

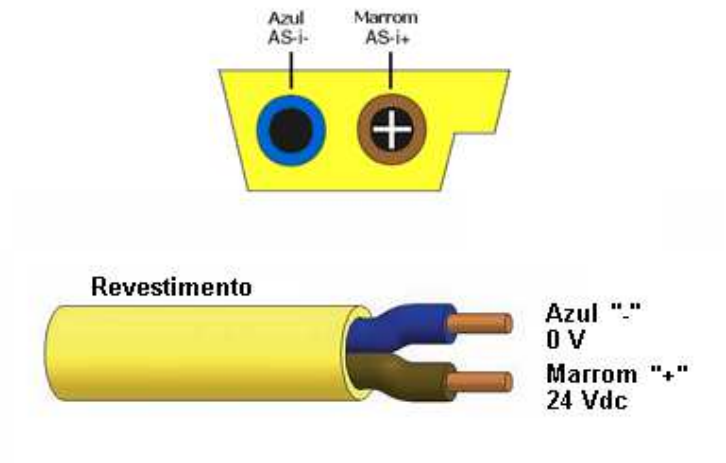


Figura 26 – Cabo AS-I. Fonte: SMAR Tudo sobre a tecnologia AS-I [12].

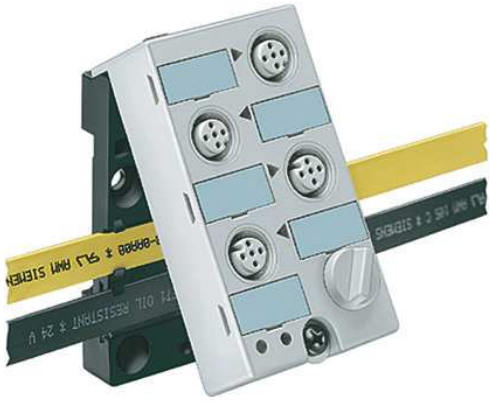


Figura 27 – Conexão de módulos AS-I. Fonte: Manual AS-I Siemens [13].

Sobre a Camada de enlace da rede, pode-se ressaltar que a comunicação utilizada é feita na técnica mestre-escravo. Já o pacote da comunicação da rede, ele se limita em 5 bits de endereçamento na rede mais 4 bit's de dados de comunicação. Sendo assim nota-se que a rede, em sua primeira versão era limitada em 31 escravos, sendo que atualmente a rede é limitada em 64 escravos [13].

A primeira versão da rede limitava a comunicação em para apenas sinais lógicos, ou seja, apenas um bit por comando. A comunicação de sinais analógicos discretizados foi implementada na versão 3.0 de 2004, porém, eles necessitam de mais de um ciclo de comunicação para serem enviados, caso possuam mais de 3 bit's. A palavra de comunicação da rede AS-I pode ser observada na Figura 28.

Por fim, um último ponto que merce destaque sobre a rede AS-I é o chamado AS-I *safety*. Que são módulos de escravos redundantes, usados em aplicações de segurança. Eles ocupam dois endereços na rede afim de serem lidos mais rapidamente além de possuírem prioridade na comunicação.

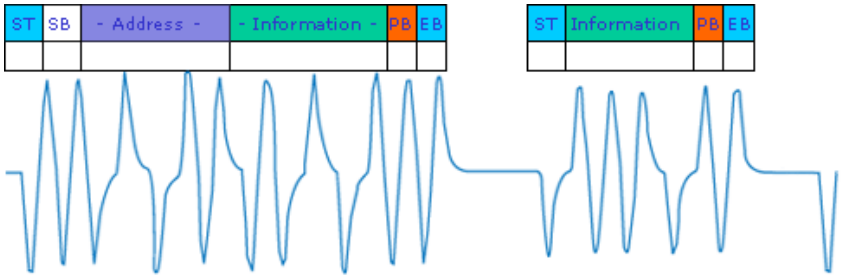


Figura 28 – Pacote de dados da rede AS-I. Fonte: AS-International Academy [14].

2.4.1.2 PROFINET

A rede PROFINET usa o padrão Ethernet(802.3) e o protocolo TCP/IP, dessa forma é possível conectar outros equipamentos como computadores, servidores e impressoras usados na internet comum. O protocolo em si foi padronizado e incluído nas normas: "Industrial communication networks" IEC 61158 e IEC 61784 [36].

A rede PROFINET pode ser dividida em dois tipos a saber: PROFINET CBA e PROFINET I/O [15]. Podemos ver o relacionamento deles através da Figura 29.

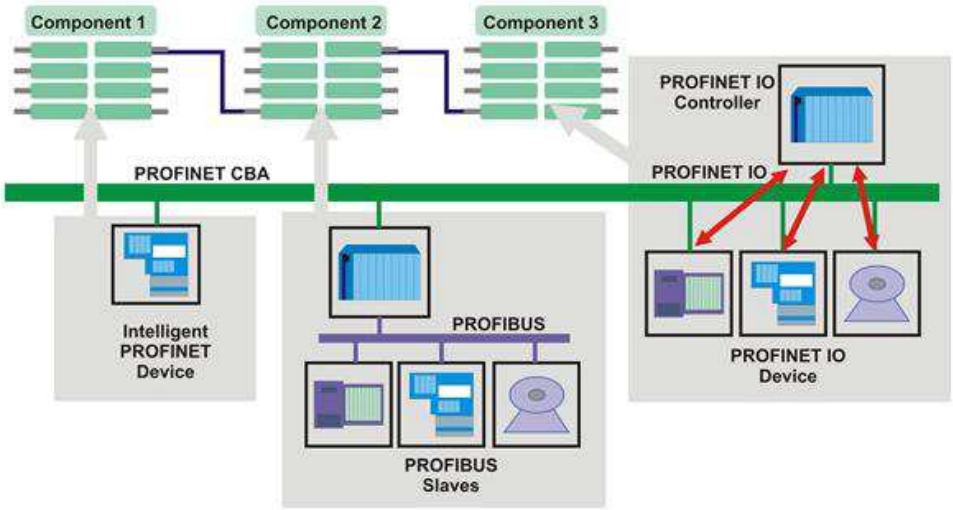


Figura 29 – Relacionamento entre PROFINET CBA e PROFINET I/O. Fonte: Industrial Networks SMAR [15].

O PROFINET CBA é indicado para aplicações onde o tempo não é crítico, por exemplo, servir de *gateway* para uma rede PROFIBUS DP [37], ou seja, nos níveis de gerência do sistema de controle. Já o PROFINET I/O, baseado no PROFIBUS DP, é usado nas aplicações de tempo real [37], sendo assim indicado para a comunicação dos baixos níveis da pirâmide de comunicação industrial, Figura 25.

Os dispositivos de campo no PROFINET I/O recebem um arquivo baseado em XML com a descrição do elemento. A extensão desse tipo de arquivo é o GSD (*Geral Station Description*)

Já a comunicação pode-ser divida em três tipos *Non-real time* (NRT): comunicação em tempo não real (suíte de protocolos TCP/IP padrão), *Real time* (RT): comunicação em tempo real e *Isochronous Real Time* (IRT): comunicação em tempo real isócrono [38], cada um desses tipos tem suas camadas do modelo OSI diferentes. Na Figura 30 pode-se observar as camadas de cada um dos modos de comunicação.

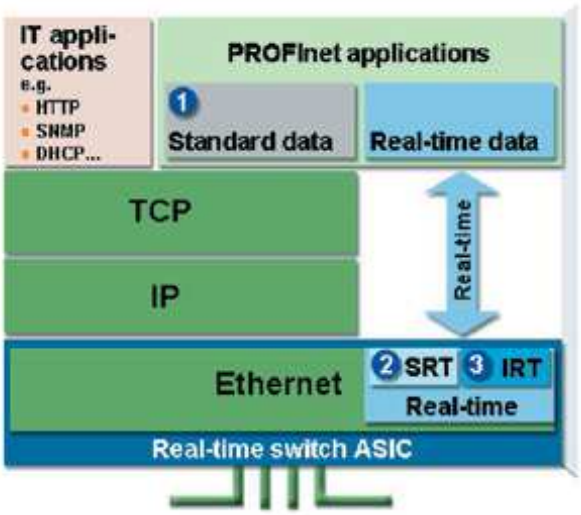


Figura 30 – Camadas Profinet. Fonte: PROFINET Real-Time Communication [16].

Nota-se que o modo *Non-real time* não tem as camadas 5 e 6 do modelo ISO/OSI, para acelerar o tempo de comunicação. Já a aplicação RT tem somente as camadas 1, 2 e 7 descartando as camadas TCP e a IP, afim de acelerar ainda mais a comunicação, chegando na ordem de alguns milissegundos e com um *jitter* de 15% [16].

No caso da IRT existe a sincronia dos *clocks* de todos os equipamentos da rede, levando o *jitter* da rede na ordem de 1 microssegundo. Porém são necessários *hardwares* específicos para essa aplicação.

Com relação ao quadro de comunicação, pode-se observá-lo na Figura 32, ressalta-se os campos VLAN que indica qual é a rede virtual local em que o dispositivo está conectado. E o campo Ether-type que é onde é definido qual é o tipo de comunicação utilizada NRT, RT ou IRT. Nota-se também que a quantidade de dados trafegados, supera e muito os do AS-I chegando ao máximo de 40 bytes de comunicação.

Com relação ao meio físico, a topologia da rede pode ser feita em linha, estrela, anel e árvore. Já sobre os meio de comunicação pode-ser qualquer um compatível com ethernet, como bluetooth, Wlan, fibra ótica, cabo de cobre, etc. Porém cada um desses meios recebe uma classificação diferente, que em resumo diz quais os tipos de comunicação do Profinet eles suportam, por exemplo o Wlan suporta apenas as aplicações NRT, já a fibra ótica suporta todos. Já os conectores das redes físicas existem nas mais váriadas formas, Figura 32. Um último detalhe é que as redes *Profinet* em diagramas de rede são representadas na cor Verde.

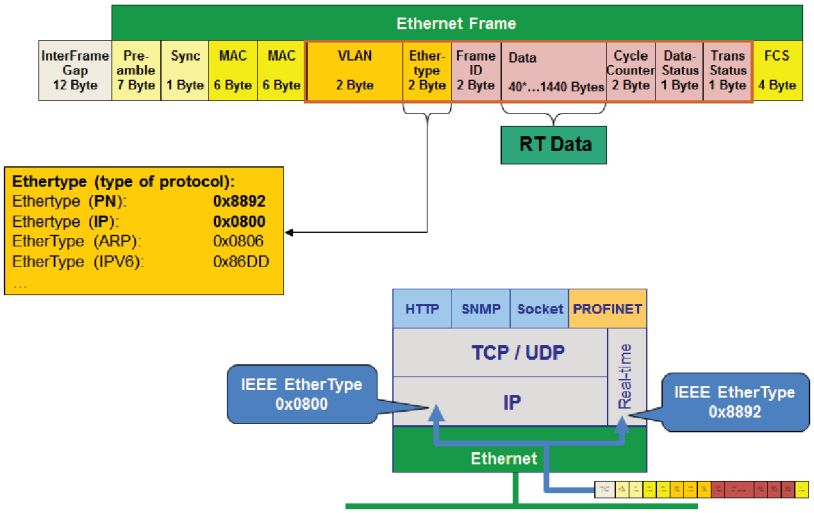


Figura 31 – Quadro de comunicação Profinet. Fonte: PROFINET Real-Time Communication [17].

Figura 32 – Tipos de Conectores PROFINET. Fonte: Structured industrial cabling and connection under Profinet [18].

2.5 Conclusão

Nessa seção foi abordado os principais assuntos que serão usados como base para fazer o projeto em questão. Inicialmente foi abordado um descritivo do geral do processo de extração de óleo de soja, detalhando o setor de recebimento e transporte de grãos (Operacional) e seus equipamentos e requisitos de automação. Dentre esses requisitos destaca-se a necessidade de não permitir carregar dos elevadores, transportadores e máquinas que estão desligados.

Também, foi explicado sobre diagrama de processos, arquitetura de sistemas distribuídas de rede industriais (DCS), bem como as linguagens de programação utilizadas no projeto (CFC e SFC). No fim, foi abordado um pouco das estruturas de redes industriais, focando nos protocolos utilizados no projeto, Actuators and Sensors Interface e PROFINET.

3 Ferramentas de Engenharia

A metodologia utilizada para fazer o presente trabalho foi basicamente estudar as ferramentas de engenharia utilizadas pela empresa GreyLogix Brasil e aplicá-las para o presente projeto.

Assim, neste capítulo será explicado brevemente as principais ferramentas utilizadas. Elas, por sua vez, têm em comum que são *softwares* da empresa SIEMENS, sendo que cada uma delas foi desenvolvida para atuar em partes específicas do processo de criação de uma solução de engenharia de controle e automação. Além disso, por serem da mesma empresa elas têm um certo grau de integração entre elas. Essa integração foi brevemente citada na seção 1.4.

3.1 COMOS

A ferramenta COMOS, da empresa SIEMENS é um software que unifica e indexa a grande parte dos documentos indústriais como, por exemplo, diagramas elétricos unifilares/multifilares, diagrama de processos *PI&D*, descritivos de equipamentos, entre outros. Além de ter ferramentas que auxiliam tanto na hora da montagem, na manutenção, quanto no gerenciamento da fábrica. Esse software tem dois grandes conceitos base para a estruturação das diversas informações, o primeiro conceito é o uso de um banco de dados interno e o segundo está relacionado como as informações são criadas, usando conceitos de programação orientada a objetos (POO).

Dentro disso, esse software trata cada equipamento como um único objeto com diferentes representações, com várias propriedades (metadados). Dessa forma, facilitando nas futuras consultas dos documentos.

Para ilustrar isso podemos tomar o seguinte exemplo: suponha que é necessário encontrar as documentações de um determinado motor da linha de produção que queimou. Para encontrar as informações do "jeito clássico" seria necessário encontrar os documentos dos armários de acionamento dos motores, o fluxograma do processo, o *datasheet* do motor entre outras informações. Com o uso da ferramenta COMOS essas tarefas ficam mais simples, visto que basta acessar o programa, digitar o nome/TAG do motor, e todas as informações que estão relacionadas a quele equipamento estarão indexadas junto ao objeto.

Além disso o software conta com um sistema de versionamento, onde cada vez que alguma informação é alterada ela é registrada, porém não é colocada no banco de dados de imediato. O sistema de versionamento também não sobreescreve informações, apenas as registra. Nos dois casos, cabe ao administrador do sistema, ou a algum revisor validar

as informações antes de elas entrarem no sistema.

Na interface do software ele divide um projeto em duas principais visões a *unit* e a *locations* além disso ainda há uma treceira visão, *base objects*, que está relacionada ao projeto indiretamente.

A visão de *Unit*, na Figura 33, é onde ficam as informações de processo. Na figura pode-se observar como o *software* divide a hierarquia dos processos Projeto > Planta > Unidade > Subunidade > Pastas divididas em categorias, que são subdivididas em equipamentos, motores, válvulas, tubulação, malhas de instrumentação e controle, equipamentos de transporte e outros. Nessa aba é onde ficam os diagramas de P&ID.

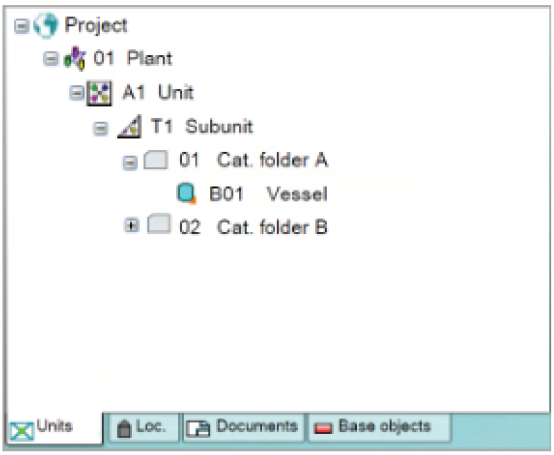


Figura 33 – Visão *Unit* COMOS. Fonte: Implementação de um sistema de controle distribuído em uma linha de produção de margarina [19].

A visão de *locations*, é onde ficam os diagramas elétricos e de automação (Figura 35). Nessa aba a hierarquia é: Projeto > Planta > Prédio > Andar > Sala > Painel. Ainda pode-se criar mais um nível, uma gaveta para o painel. Como pode-se perceber, essa aba está mais relacionada com a parte física, tanto que a hierarquia é separada em prédio, andar, etc.

A aba *Documentations*, segue a mesma ideia das abas *Locations* e *Unit*, porém é onde é armazenado a documentação. Já a aba *base objects* é a qual onde esta a base de dados dos objetos, que incluem os modelos de desenhos, tabelas, modelos de documentação elétrica, de processo, e afins.

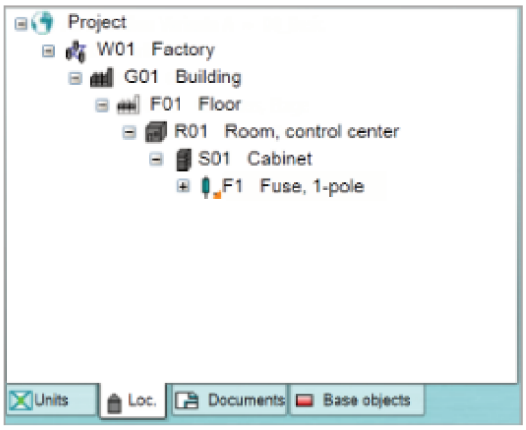


Figura 34 – Visão *Location* COMOS. Fonte: Implementação de um sistema de controle distribuído em uma linha de produção de margarina [19].

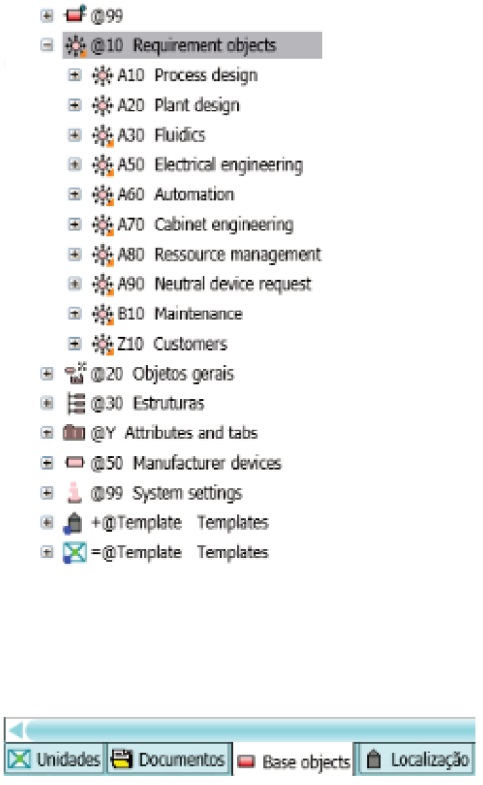


Figura 35 – Visão *Location* COMOS. Fonte: Implementação de um sistema de controle distribuído em uma linha de produção de margarina [19].

3.2 PCS7

O PCS7 é uma ferramenta de desenvolvimento de sistemas de controle de processo, feito pela multinacional Siemens. Ele é capaz de integrar todos os sistemas de controle de uma indústria, desde os sistemas de controle contínuo ou por batelada, intertravamentos, sistemas de segurança e todos os dispositivos de campo (instrumentação e acionamento de

motores). Assim, o desenvolvimento do *software* de controle é totalmente unificado. Internamente, o ambiente de engenharia é divido em 4 partes *Master Data Library, Component View, Plant View* e *Process Object View.*

A *Master Data Library* é onde está toda a biblioteca usada no projeto. Nela é possível adicionar biblioteca extras fornecidas pela Siemens, bem como fazer modificações, específicas para o projeto, em alguns dos componentes das biblioteca pré-existentes. Por exemplo pode-se fazer CFC's (linguagem de programação utilizada pelo PCS7), padrão para um determinado aplicação, já que geralmente em projetos grandes, tem-se centenas de equipamentos com lógicas idênticas ou muito parecidas, bem como, em caso de modificação atualiza-los todos, sem ser necessário abrir CFC por CFC. Esses CFC modelos recebem o nome de CMT (*controle module type*).

A *Component View* é onde são feitas as declarações e as configurações dos elementos de *hardware*, indo desde as CP's e CLP's até os motores, válvulas e atuadores. Na Figura 36, mostra essa visão do projeto nela pode-se ver as diferentes separações físicas dos projetos, nela o Operacional é onde esta o hardware de baixo nível e o OS01 é onde está declarado o servidor da que irá executar as aplicações do supervisório.

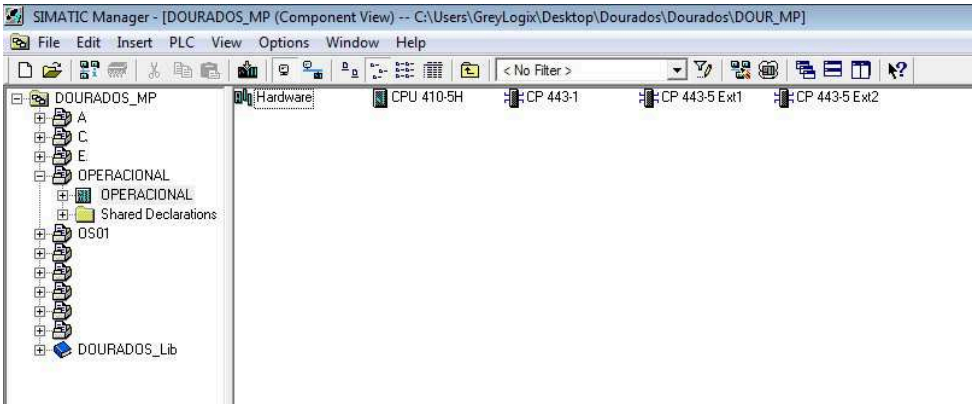


Figura 36 – *Component View* PCS7. Fonte: Original.

A *Plant View*, Figura 37 é a janela onde é separado a estrutura hierárquica do ponto de vista de processo. Nessa janela é onde se cria novos CFC's, SFC's e *pictures*, tipo de arquivo das telas dos supervisórios. Esses depois serão abertos e programados em uma janela própria de programação.

Por último, A *Process Object View* (Figura 38), é uma tela que, hierarquicamente, se assemelha com a *Plant view*. Porém, sua função é a de acessar todos os dados dos chart's, facilitando assim a programação e a modificação de parâmetros em massa.

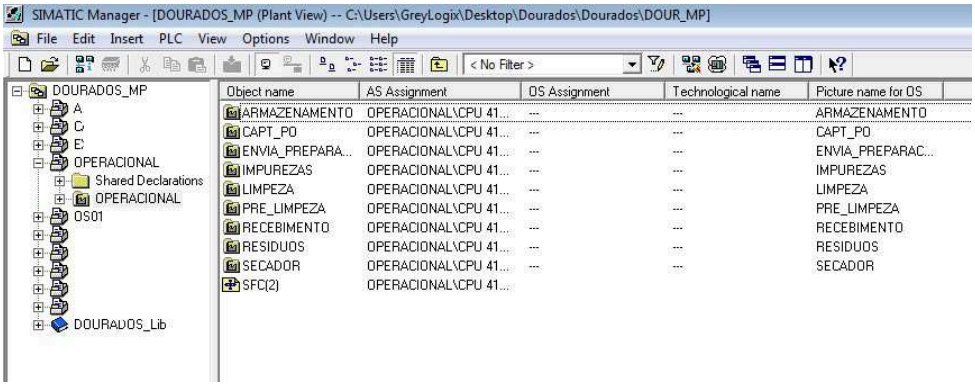


Figura 37 – *Plant View* PCS7. Fonte: Original.

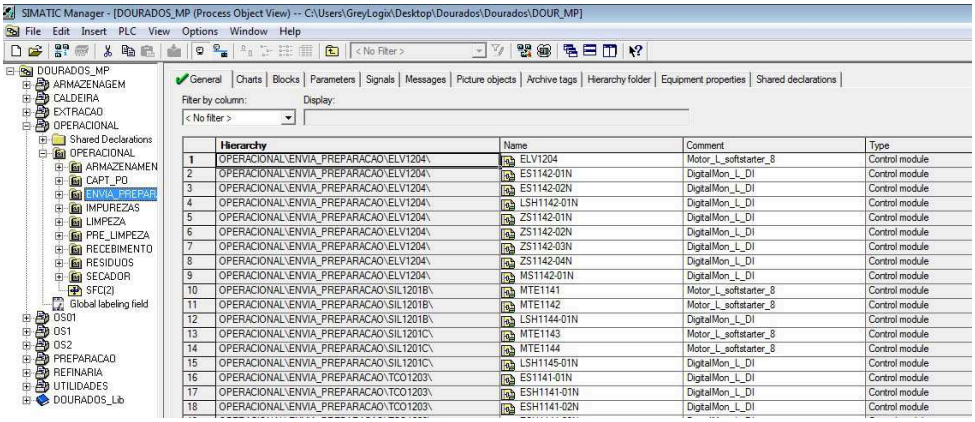


Figura 38 – *Process Object View* PCS7. Fonte: Original.

3.3 SIMIT

SIMIT é um *software* de simulação que tem integração com os outros softwares da SIEMENS. No caso deste projeto o principal objetivo das simulações feitas no *software* tem por objetivo fazer testes da programação, simulando dinâmicas de processo, como a simulação do enchimento de um tanque, e eventos como o sinal de nível alto do tanque.

Além de fazer a simulação do processo o SIMIT tem um *software* interno chamado *Virtual Controller* (VC), que tem a função de simular o CLP da planta. Esse por sua vez, tem total integração com o *software* do PCS7, sendo assim o programa ainda em produção no PCS7 já pode ser compilado no VC. Também é possível controlar a simulação a partir das telas dos supervisórios que estão em desenvolvimento.

Dentro do SIMIT as simulações ocorrem em basicamente três níveis diferentes. O primeiro nível é o nível de sinal onde são gerados apenas os sinais que o CLP irá receber, o segundo nível é a dinâmica do equipamento e no terceiro nível temos a simulação da planta toda. Esses três níveis de simulação, são executados ao mesmo tempo.

Para exemplificar esses níveis de simulação tomamos o exemplo da simulação de uma válvula seletora, em que os seus três níveis podem ser divididos da seguinte forma:

1. no primeiro nível temos a simulação dos sinais, nele pode-se definir, manualmente,

os valores dos *feedbacks* enviados ao PLC, bem como visualizar os sinais de comando recebidos;

2. Nesse nível tem-se a simulação da dinâmica da válvula. Assim ao invés de configurar manualmente os sinais lidos do PLC, pode-se simular o tempo de abertura da válvula, dessa forma quando a simulação da válvula receber o sinal de abrir, ela irá simular a dinâmica de abertura da válvula, e só depois de um determinado tempo a simulação envia o *feedback* ao VC;

3. Por fim, no terceiro nível tem-se a função da válvula no processo, nesse caso direcionar o fluxo de produto para uma das suas saídas.

Outros pontos que merecem destaque nessa hierarquia de simulação são: no primeiro nível da simulação é feito dentro do SIMIT atráves de uma aba chamada *coupling*, que é a ligação direta entre a simulação (SIMIT) e o PLC simulado (VC), essa aba funciona como uma tabela que mostra os endereços dos sinais PLC simulado, bem como o nome a descrição e o valor atual. Em simulação os valores desses sinais podem ser trocados. A Figura 39 mostra essa aba.

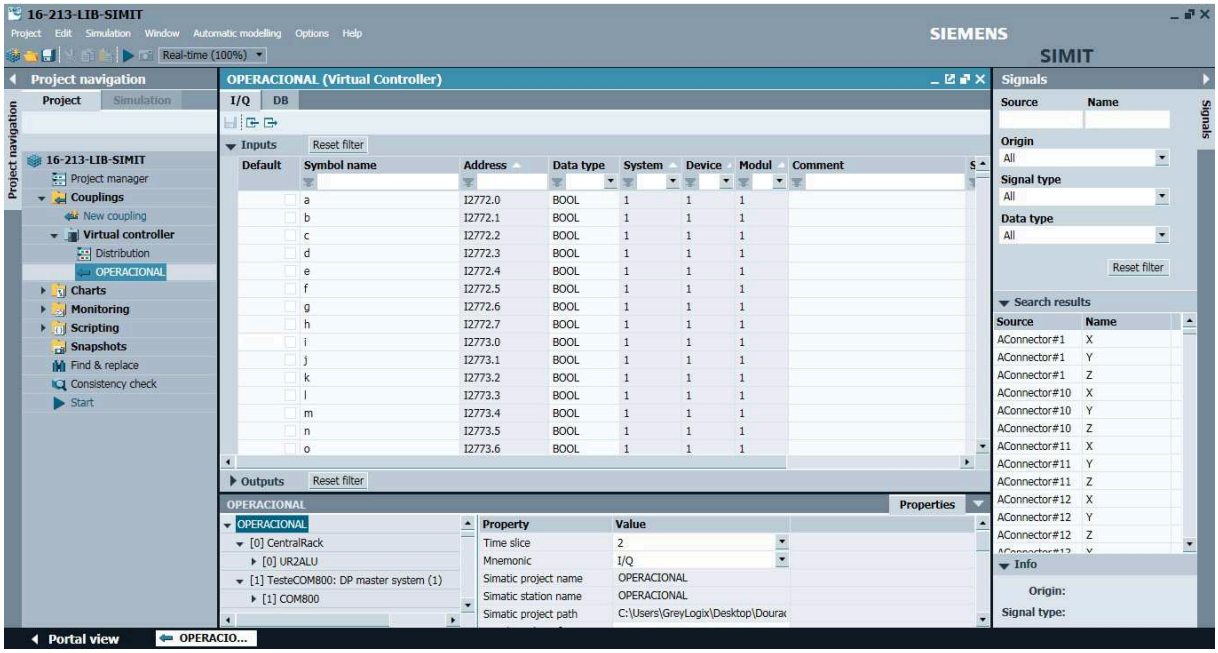


Figura 39 – Interface de simulação nível de Sinal. Fonte: Original.

A programação da simulação em si (a partir do segundo nível) é feita através de uma interface de blocos básicos, na Figura 40 pode-se observar a estrutura básica da simulação de um silo. Por comparação, pode-se dizer, que ele é bem parecido com o *Simulink* do *MATLAB*.

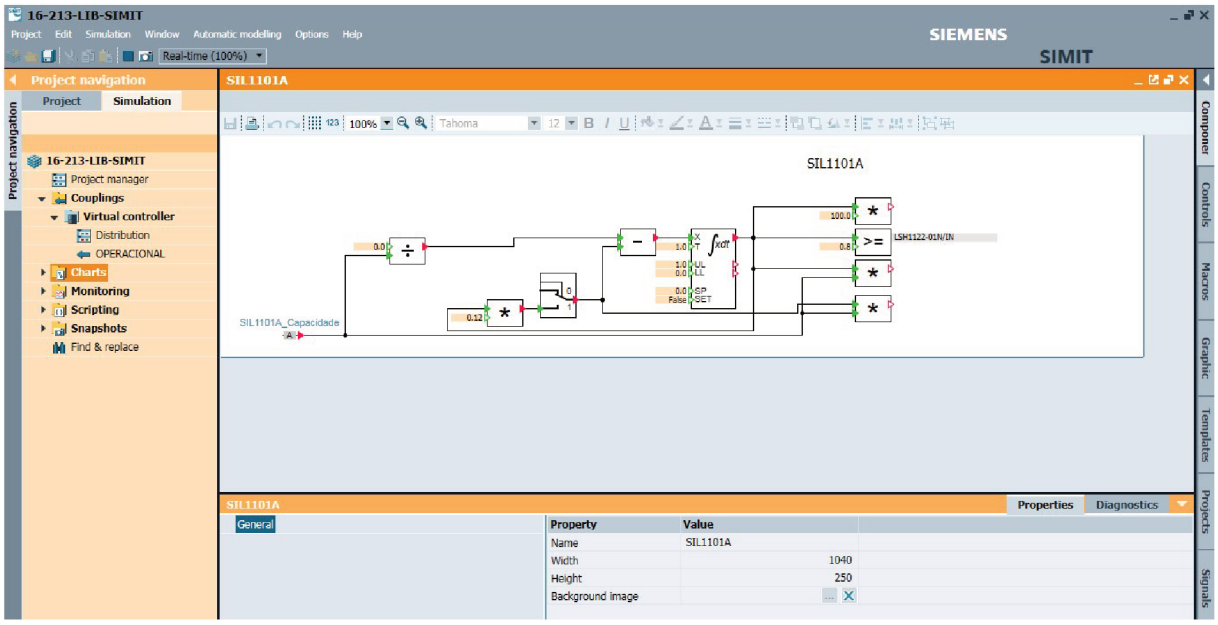


Figura 40 – Interface de simulação Silo SIMIT. Fonte: Original.

Além de programar os elementos da simulação com os blocos base, o SIMIT ainda permitem a criação de blocos, com programação em texto, usando uma linguagem de programação própria baseada em C. A Figura 41 mostra essa interface.

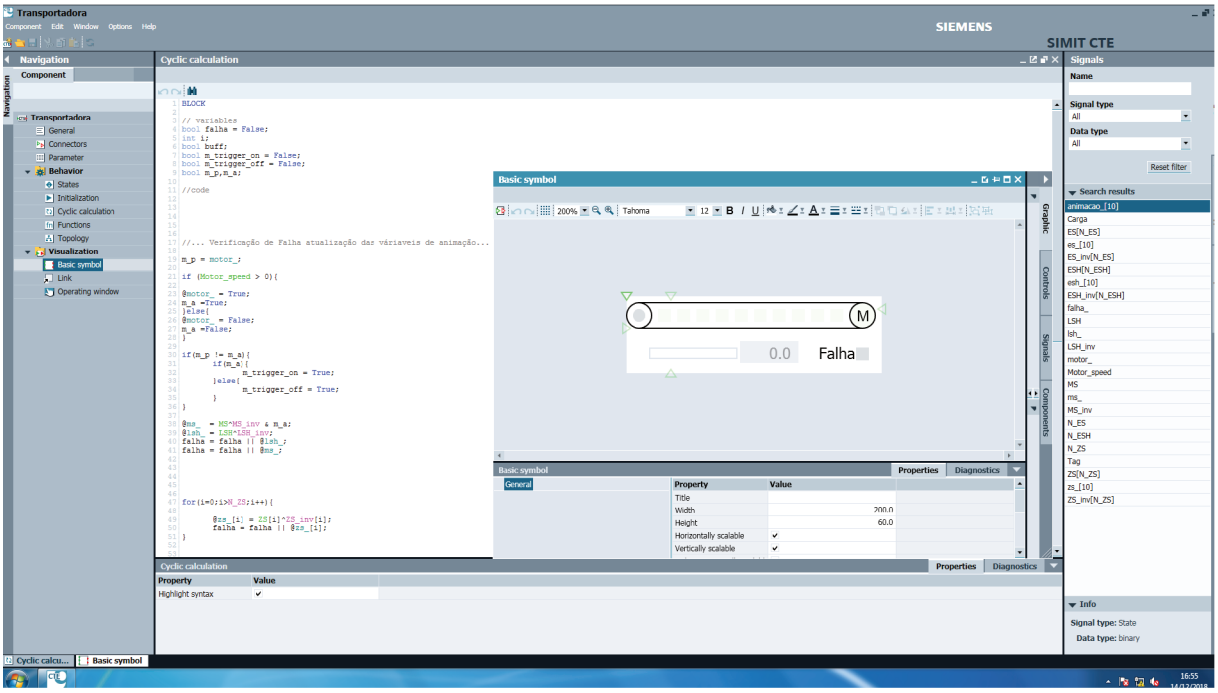


Figura 41 – Interface de criação de Blocos SIMIT. Fonte: Original.

Dentro da simulação no SIMIT é possível aplicar *Scripts* para fazer teste de funcionamento. Além disso, uma vez validada a simulação, é possível usar os *scripts* para colocar a planta simulada em situações de risco e emergência, e com isso treinar os operadores para tais situações. Na figura 42 mostra-se a interface de programação dos *scripts*.

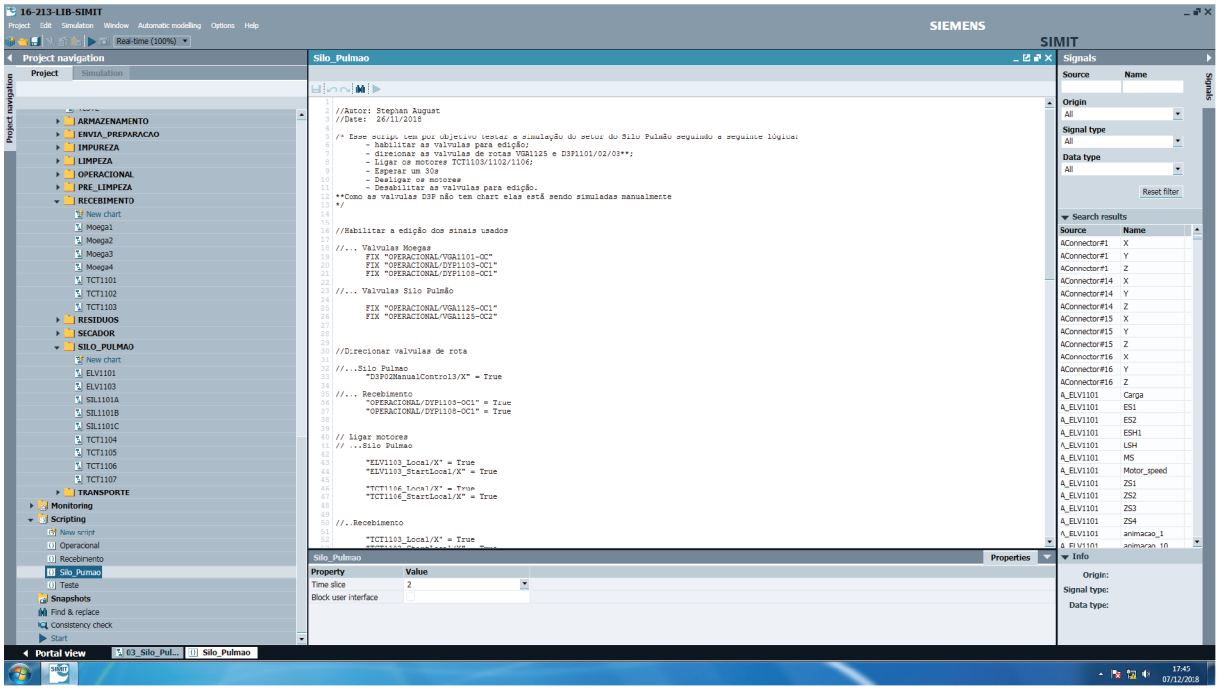


Figura 42 – Tela de *script* SIMIT. Fonte: Original.

Uma outra funcionalidade interessante do SIMIT é o poder de se comunicar com outros *softwares* usando os recursos de memória compartilhada do *Windows*. Dessa forma pode-se, por exemplo, comunicar a simulação com o MATLAB.

3.4 Conclusão

Nessa seção foram apresentados os softwares utilizados no projeto. Assim deve-se ressaltar que o *software* COMOS é uma ferramenta de agregação de documentação inteligente. Por sua vez, o PCS7 é utilizado para fazer a programação das estruturas de controle, tornando o processo de programação de sistemas DCS simples. Por fim o SIMIT é uma ferramenta de desenvolvimento de simulações, que usa a linguagem de programação em blocos e tem integração com o PCS7.

4 Resultados

Neste capítulo serão mostrados os resultados adquiridos com esse projeto. Para não expor dados sigilosos do projeto, foi escolhida uma das sub-áreas do setor operacional para ser apresentada. Essa sub-área é referente a parte de enviar a soja para preparação (Sub setor Envia Prepearação), que consiste em duas esteiras transportadoras, um elevador, dois silos e uma válvula seletora, que define se a soja é enviada diretamente para a preparação ou se ela é enviada para um dos silos pulmão, o diagrama dessa parte da planta será apresentado na Figura 43.

Depois, será apresentado a biblioteca base do projeto, destacando as modificações implementadas5 para atender aos requisitos da planta (Seção 4.2). Em seguida, mostrar-se-á o diagrama de rede e a representação do mesmo no PCS7 (Seção 4.3).

Na Seção 4.4, serão abordados os intertravamentos e sequência feitas para o programa de controle da sub-área "envia preparação". Posteriormente, serão apresentadas as modificações feitas nas telas do sistema supervisor. Finalmente, na ante-penúltima seção será abordada a criação da biblioteca de simulação, assim como a simulação da planta. Na penúltima seção, deste capítulo, ter-se-á alguns testes que mostrarão a simulação funcionando juntamente com o sistema de controle. A última parte será uma conclusão dos resultados.

4.1 P&ID

Para fazer a revisão do P&ID no *software* COMOS foram analisados os documentos disponibilizados pelas empresas colaboradoras com a revisão das TAG's e indicação dos instrumentos de cada equipamento. Até então no diagrama do COMOS não havia esses instrumentos. A alocação deles na hierarquia do COMOS deveria ser feita seguindo uma estrutura que facilitasse a posterior exportação para o PCS7. Na Figura 43 pode-se ver o resultado dessa revisão com correção no *software* COMOS, em vermelho estão circulados os instrumentos colocados e em amarelo estão as TAG's atualizadas.

Sobre a hierarquia (pasta) na qual esses instrumentos foram criados, como se pode observar na Figura 44, não é o mesmo de onde estão os equipamentos e motores da planta. Isso se deve ao fato de que a pasta chamada de *DCS* foi criada para conter uma documentação do *software* que está controlando os sistemas de cada área, além de, como já citado, tornar mais simples fazer o processo de migração do instrumentos para o PCS7.

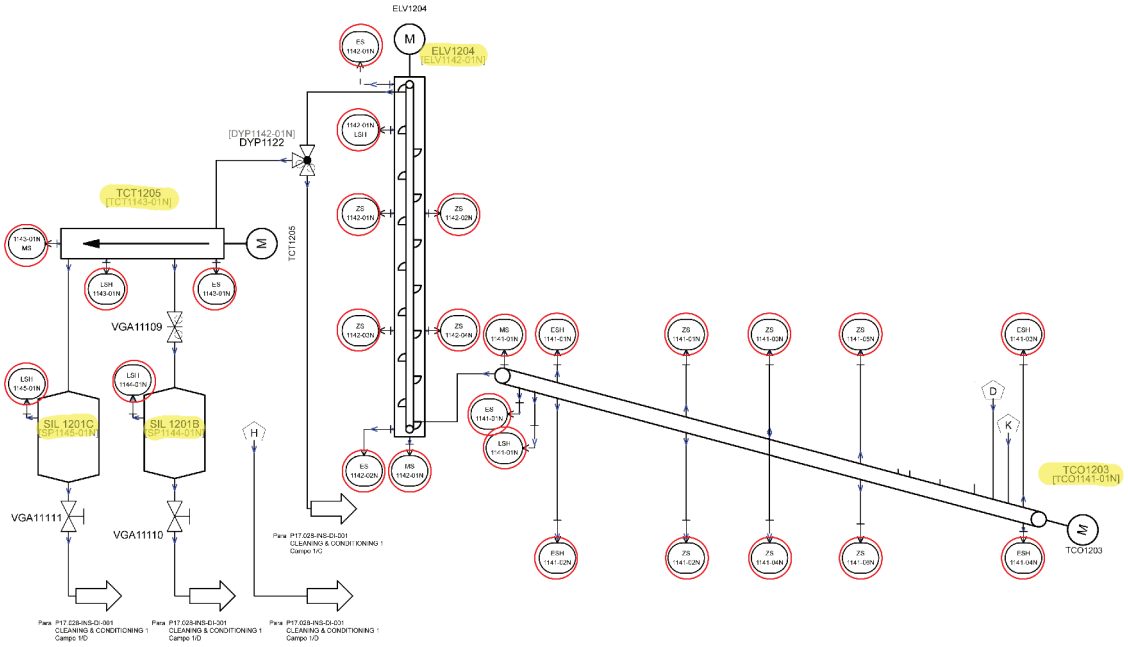


Figura 43 – P&ID COMOS. Fonte: Original.

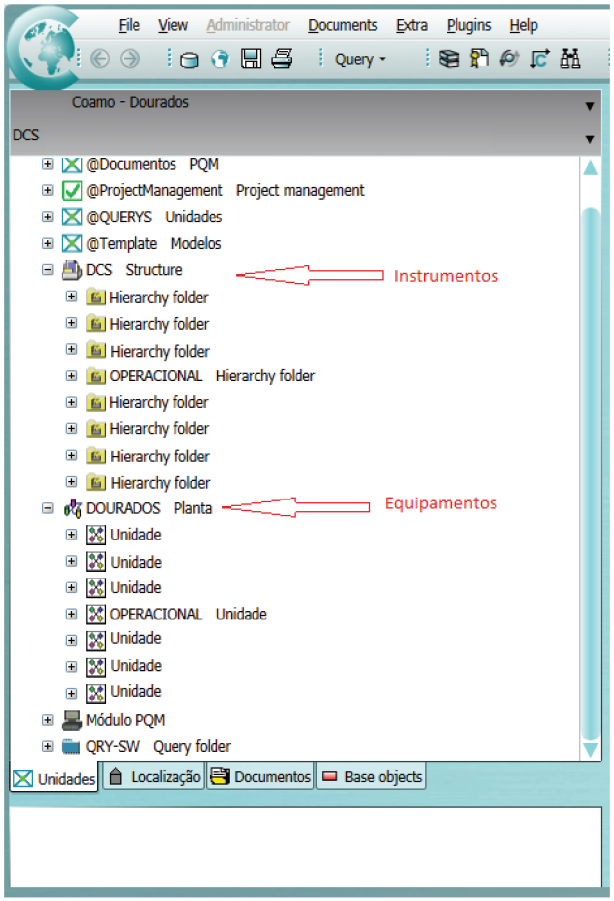


Figura 44 – Estrutura hierárquica da pasta de instrumentos. Fonte: Original.

4.2 *Controle Module Types* CMT's

A programação do PCS7 é feita em CFC's, sendo assim, devido a grande quantidade de elementos semelhantes, optou-se em usar CMT's (*controle module types*) que são modelos de CFC's, quando eles são instanciados, constroem o CFC do equipamento. Além disso, para eles existem *plugin's*, que fazem o instanciamento dos CMT's a partir do *softwar* COMOS. Assim, o trabalho de atualizar o COMOS, tem o objetivo de ajudar na criação dos CFC's do PCS7.

Na biblioteca dos CMT's, dentro do PCS7, existem CMT's padrão de cada tipo de equipamento, por exemplo, motor acionado por inversor, motor acionado por *softstarter* ou sensor capacitivo de desalinhamento, esse detalhamento de blocos, é específico no ponto de definir até o tipo de comunicação usada para receber e enviar os sinais, ou seja, praticamente, cada CMT tem uma versão AS-I e PROFIBUS/PROFINET. A biblioteca usada na importação pode ser observada na Figura 45.

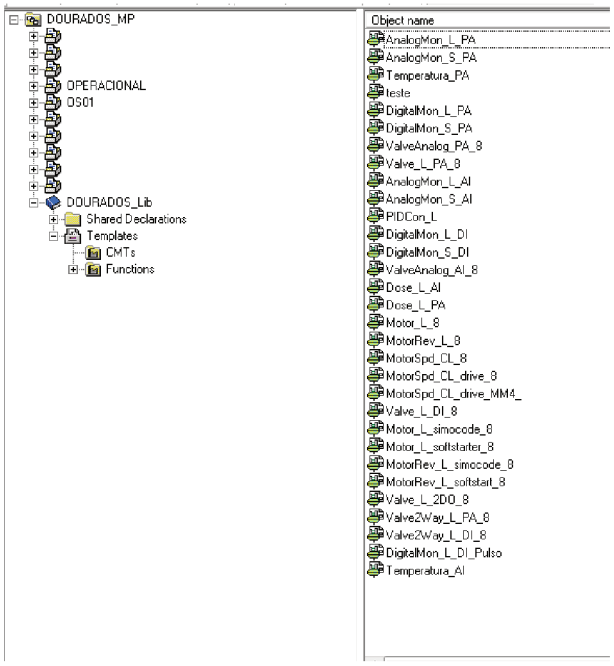


Figura 45 – Biblioteca dos CMT's utilizada no projeto. Fonte: Original.

Pode-se observar na Figura 45, que existem CMT's para os diferentes tipos de equipamentos, vale ressaltar que as duas últimas letras do CMT indicam o tipo de comunicação, onde PA indica PROFIBUS PA e AI indica AS-I.

Assim a função do *plugin* de importação (COMOS → PCS7) é fazer a sincronização dos equipamentos dos P&ID e diagrama elétricos, e cria os *chart's* faltantes no PCS7. Usando esse *plugin* foram gerados os CFC's base de cada um dos equipamentos. Esse *plugin* também funciona no sentido contrário, atualizando o COMOS com as informações do PCS7, esse procedimento deverá ser feito após as correções finais feitas no comissionamento do planta, e que não pertence ao escopo deste trabalho.

O primeiro passo após esse processo de migração foi alocar os *charts* em pastas dividindo o setor do operacional em 9 sub-áreas. Aqui, vale explicar que pelo tamanho da área do operacional, foi optado por dividi-la em 9 sub-áreas, essa divisão foi feita tomando como base a primeira versão das telas. O antes e o depois da separação em hierarquias pode ser observado nas Figuras 46 e Figura 47 respectivamente.

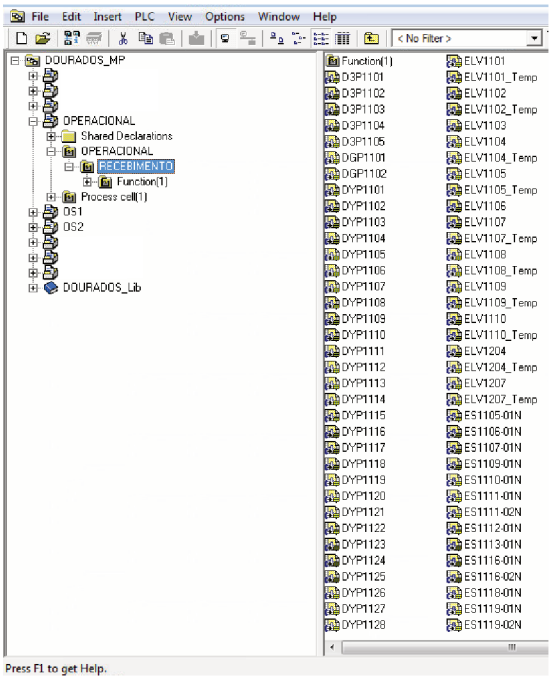


Figura 46 – Hierarquia antes da separação. Fonte: Original.

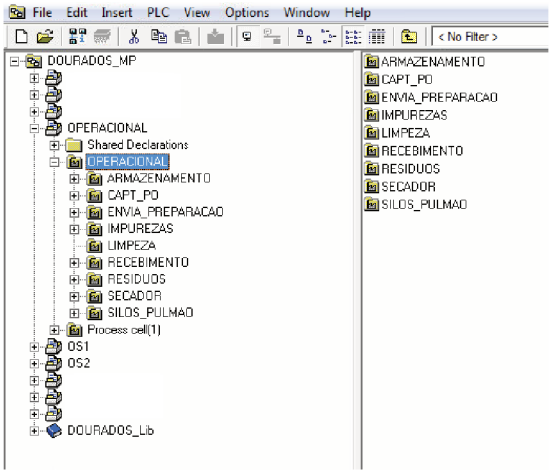


Figura 47 – Hierarquia depois da separação. Fonte: Original.

A primeira migração, usando a biblioteca desse projeto, foi na área em questão. Assim, foi necessário acrescentar e corrigir alguns CMT's.Um dos CMT's que precisou ser adicionado foi o CMT da Chave de Velocidade, usado em todas as esteiras e elevadores. Inicialmente, havia sido pensado em utilizar o *chart* de sinal digital, Figura 48.
Mas, o sensor em questão (Chave de rotação), funciona da seguinte forma: ele é posicionado fisicamente no eixo final de uma esteira, quando o motor liga ele faz mover a correia, e por consequência, rodar os eixos. No último eixo exite uma engrenagem e por cima desta está posicionado um sensor indutivo. Com o movimento de rotação da engrenagem o sensor indutivo pulsa informando que o motor está de fato movendo a esteira, e não girando em falso. A ideia do funcionamento desse sensor pode ser observado na figura 49. A mesma ideia aplica-se aos sensores de rotação dos elevadores.

Sendo Assim, o sinal gerado pelo sensor é um sinal pulsante, nesse caso o *chart* de sinal digital simples não serviria para fazer a detecção da rotação. Para isso foi implementado o bloco (`DigitalMon_L_DI_Pulso`) (Figura 50), que tem a função de transformar o sinal pulsante recebido do sensor em um sinal binário, onde 1 indica o funcionamento correto, ou seja, motor parado esteira parada ou motor ligado transportadora em movimento e 0 indica o funcionamento incorreto motor ligado e transportadora parada.

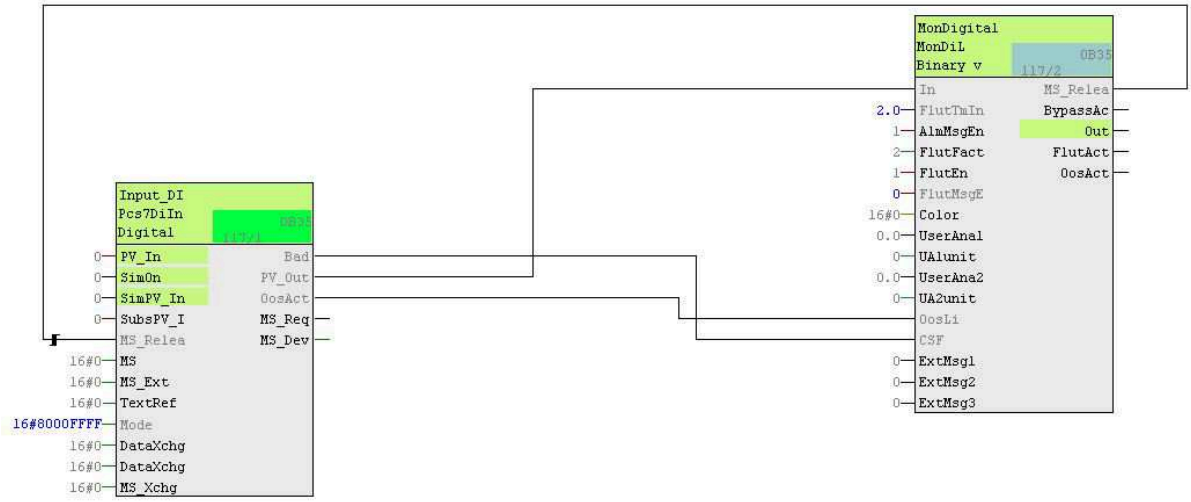


Figura 48 – *Chart* Bloco digital simples. Fonte: Original.

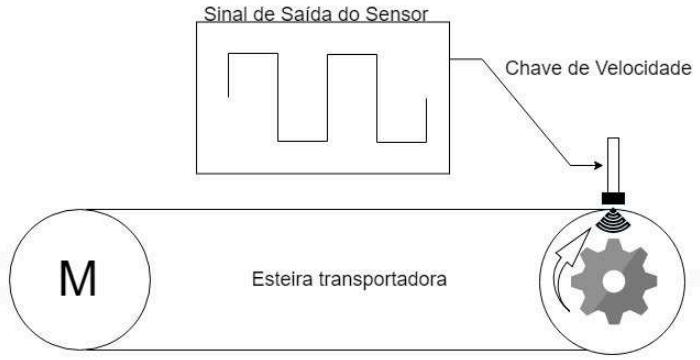


Figura 49 – Diagrama de funcionamento da Chave de Rotação. Fonte: Original.

Figura 50 – CMT Chave de Rotação Fonte: Original

O CMT, em questão `DigitalMon_L_DI_Pulso`, de maneira geral, a logica de funcionamento ocorre da seguinte forma: o bloco `input_Di`, tem a função de fazer o tratamento primário do sinal, esse é o tipo de bloco é usado para fazer o tratamento do sinal recebido e enviado do sistema e faz a adaptação do sinal dos diferentes tipos protocolos de rede. Em seguida existe o bloco `trigger_pul` que emite sinais de pulso, tanto na subida ($0 \rightarrow 1$) quanto na descida do sinal de entrada ($1 \rightarrow 0$), esses dois sinais são posteriormente "somados" com o bloco `OR_pul`. O a segunda entrada do bloco `AND_pul` deverá ser conectada ao motor do equipamento ao qual o sensor está relacionado, dessa forma fazendo com que o sinal na entrada do `timer_pul` só seja 1 caso o motor estiver ligado. O timer, por sua vez, é configurado no modo *On-Delay*, dessa forma se o sinal oscilante na entrada do bloco, ficar com sinal baixo por mais de 1s o *timer* irá ter uma saída com o valor lógioco 0, esse é enviado ao bloco `MonDigital`, que replicará o sinal na saída do *chart*. O

bloco `MonDigital` tem a função de fazer alguns tratamentos de erros e atualizar e criar os indicadores do sinal nas telas de operação (será explicado na seção 4.5).

O teste dessa lógica será apresentada posteriormente na parte de simulação (seção 4.6).

4.3 Arquitetura da Rede e Declaração de *Hardware*

A construção da arquitetura da rede foi feita com base nos documentos já existentes do projeto. Os protocolos escolhidos na área do operacional foram o PROFINET, e o AS-I. Para facilitar o entendimento podemos dividir a rede em três partes, rede entre o supervisório e o CLP, e as partes de baixo nível, rede entre o CLP e os comandos dos motores e a rede entre o CLP e os *gateways* AS-I. Essa estrutura pode ser observada na Figura 51.

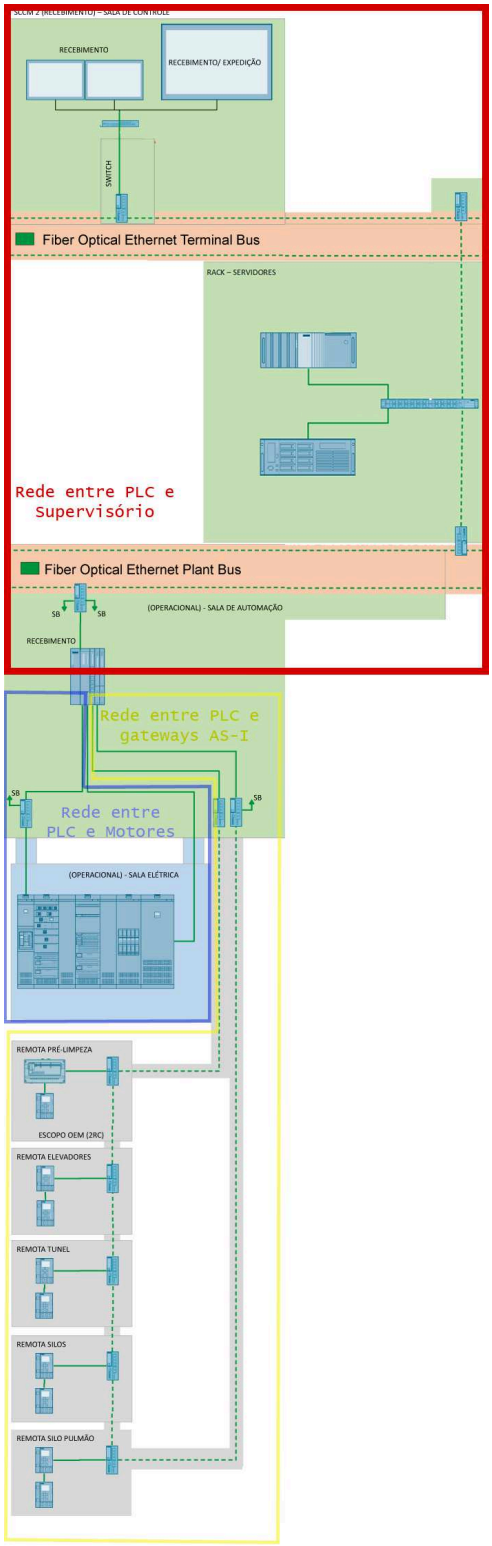


Figura 51 – Arquitetura de rede. Fonte: Original.

Sobre a rede entre o supervisório e o CLP, destaca-se que, essa rede é toda em fibra ótica e em PROFINET. Nessa camada encontra-se o servidor onde está parte da lógica do sistema (estrutura apresentada na seção 2.3.2). Foi optado por fazer duas redes, uma entre o servidor e os CLP's e outra entre o servidor e os computadores onde serão colocados os sistemas dos supervisórios. Com as abas de declaração de *hardware* do PCS7, o software já entenderá essa estrutura e fará a compilação do *software* para aquela estrutura de *hardware*.

Dentro da separação da rede, apresentada anteriormente, na rede entre o CLP e os comandos dos motores, deve ser apresentado que dentro do PCS7 foi necessário declarar, o todos os mais 150 acionamentos de motores, que ficam nos painéis da sala elétrica do setor operacional. Para isso foi necessário encontrar os *GSDś* dos diferentes tipos de acionamento dos motores, entre eles estão partidas diretas, inversores de frequência e SIMOCODE (partida direta mais uma grande gama de sensoriamento do motor), assim os programas de controle podem enviar e receber corretamente as informações dos motores. GSD é a extensão do arquivo que está relacionada a rede PROFINET IO que contem as principais informações de *hardware* necessárias para o funcionamento da rede de comunicação [37]

Dessa parte da rede, o principal trabalho foi montar a estrutura, definindo os IP's de cada um dos diferentes equipamentos, bem como encontrar os tipos de partidas. Apesar de ser uma tarefa, relativamente simples, ela é um tanto quanto trabalhosa, visto que, no caso eram mais de 150 partidas de motores, que deveriam ser criadas, identificadas com TAG's e, posteriormente, conecatadas com seus respectivos CFC's. Essa mesma, ideia se aplica à rede entre o CLP e os *gateways* AS-I, porém são bem menos *gateways*. Essas estrutura pode ser observada na Figura 52, onde mostra-se as duas redes PROFINET conectadas no CLP e em seus respectivos dispositivos periféricos.

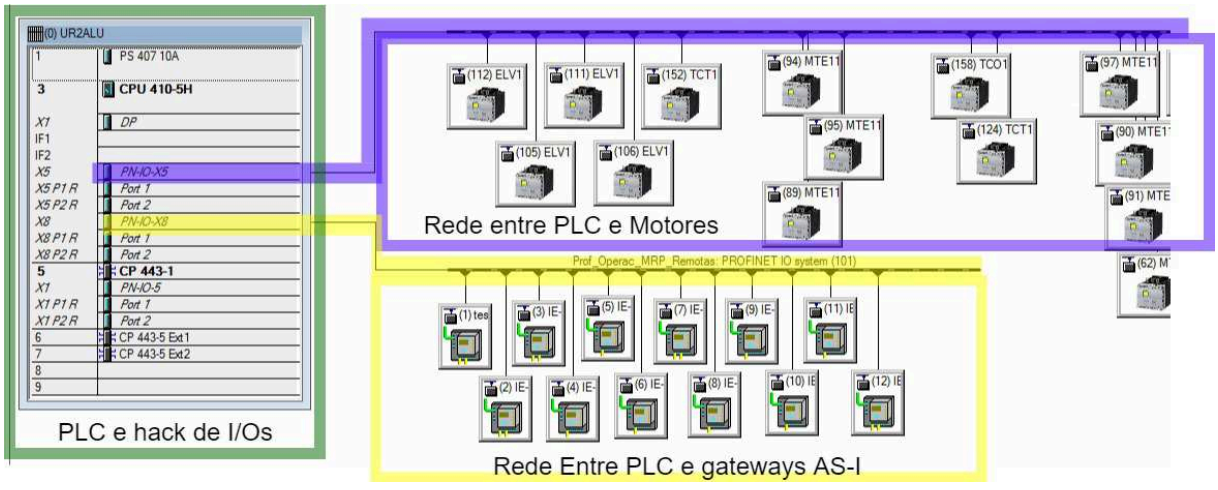


Figura 52 – Estrutura da Rede. Fonte: Original.

Para esclarecer, o CLP e seu *rack* é representado pela janela `UR2ALU`, em verde na

Figura 52. As placas de rede desse equipamento são os periféricos conectgados ao *rack* `PIN-IO-X5` e `PIN-IO-X8` onde estão conectadas as duas redes *PROFINET*. Dentro da rede dos motores (azul) nota-se alguns blocos, esses representam que a placa de rede estará a comunicação das partidas diretas dos motores (blocos conectados dentro do quadrado azul).

Pela Figura 52 pode-se notar 12 *gateways* AS-I conectados na segunda placa de rede do CLP (quadro amarelo). Nessa rede 4, dos *gateways*, são duplos e o restante são simples, sendo que um dos duplos é um equipamento para teste da compilação dessa estrutura de rede. Sobre esses *gateways* considera-se que eles são elementos que fazem a comunicação entre a rede AS-I e a PROFINET, internamente eles tem um ou dois mestres AS-I, que comunicam com uma placa de rede PROFINET.

Inicialmente haviam sido escolhidos cerca de 6 mestres duplos para comunicar com todos os elementos periféricos AS-I do setor operacional, porém houve um erro conceitual, foi pensado a quantidade de mestres considerando que cada dispositivo AS-I ocuparia apenas um endereço. Porém, a maioria dos componentes ocupam na realidade dois endereços, visto que os elementos AS-I mais comuns da rede são os leitores de temperatura com 4 entradas e existe pelo menos um equipamento desse por motor (Pelas especificações do projeto todos os motores teriam no mínimo 3 leituras de temperaturas). Alguns outros equipamentos AS-I também tiveram esse problema, são eles: os leitores de 4 entradas digitais do tipo *safe* e os botões de emergência. Um outro fator relevante, dentro dessa discussão dos endereços da rede AS-I, é que o cliente solicitou que as redes AS-I tivessem 20% de margem de folga para futuras expansões, para futura expansões. Em resumo, com a identificação desse erro, foi refeito a distribuição da rede e comprado os novos equipamentos.

Já no PCS7, a construção das redes AS-I são feitas adicionando elementos aos *gateways*. A estrutura interna de um dos mestres AS-I pode ser observada na Figura 53.

O processo de declarar os diferentes equipamentos das redes no PCS7 é chamado de "declaração de *hardware*". Com a compilação dessa estrutura o código de controle poderá já saber quais equipamentos estão em conectados a quais I/O's.

4.3.1 *Symbol Table*

Na Figura 53 pode-se notar uma janela *edit symbols*, essa faz parte da criação da *symbol table*. A *symbol table*, ou tabela de símbolos, é uma tabela que relaciona um determinado I/O do CLP com um nome (símbolo), um tipo de variável e um comentário. Dessa forma, na programação conecta-se os sinais com esses símbolos ao invés de usar diretamente endereço dos I/O's. Por exemplo na Figura 53 nota-se que a entrada `IW 1168` foi associada ao simbolo `MTE1180-PV1`. A grande vantagem de usar os símbolos ao invés dos sinais é a possibilidade de mudar os endereço lógicos (endereço das entradas e

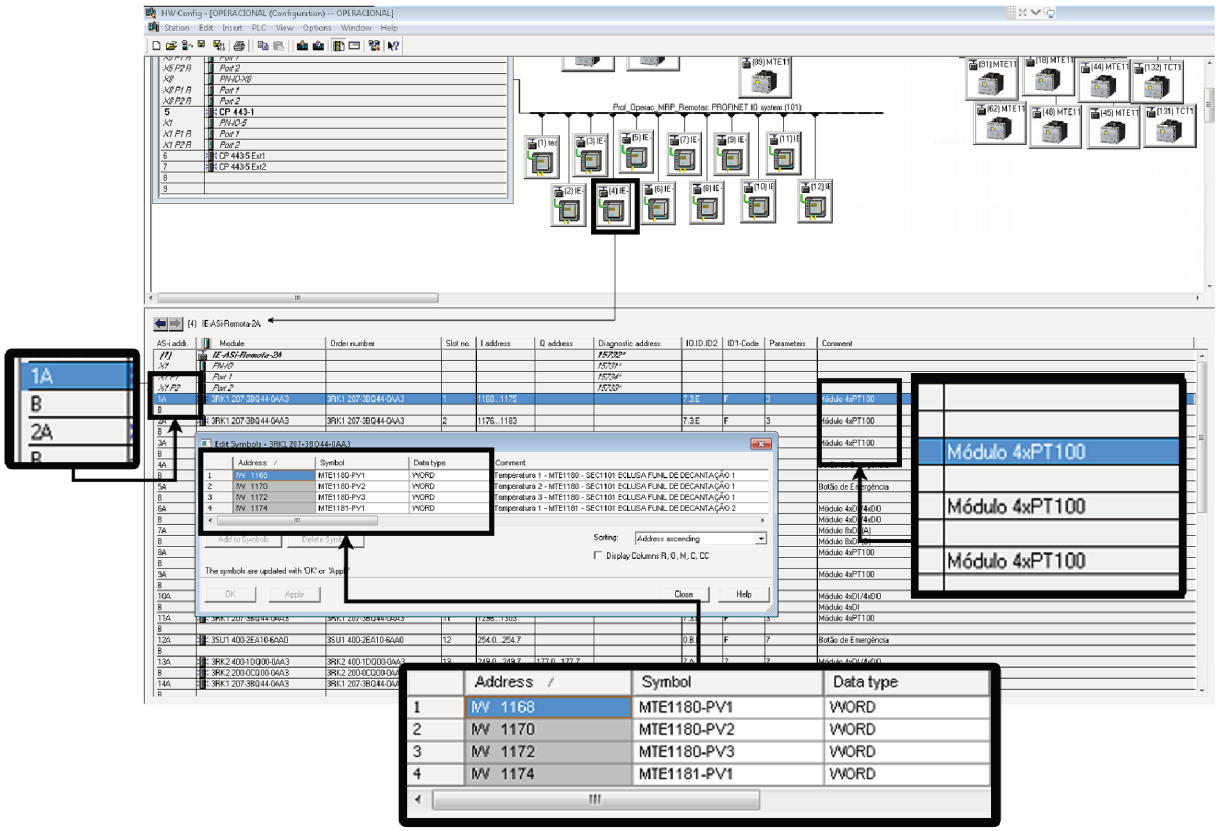


Figura 53 – Alocação dos periféricos AS-I. Fonte: Original.

das saídas) sem precisar alterar a programação.

A criação da tabela pode ser considerada o primeiro passo da programação, no caso desse projeto, ela foi preenchida usando as *symbol table's* internas de cada um dos equipamentos. Parte da *symbol table* podem ser visa na Figura 54. Então como resultado desta seção temos a *symbol table* (Figura 54) e arquitetura devidamente montadas (Figura 52 e 53).

Symbol Editor - [S7_OPERACIONAL (Symbols) -- OPERACIONAL\OPERACIONAL\CPU 410-5H]

Symbol Table Edit Insert View Options Window Help

All Symbols

	Status	Symbol	Address	Data type	Comment
256		ELV1101-ES1	I 169.0	BOOL	Botão de Emergência 1 ELV1101
257		ELV1101-ES2	I 210.0	BOOL	Botão de Emergência 2 ELV1101
258		ELV1101-LSH	I 209.2	BOOL	Sensor de Embuchamento ELV1101
259		ELV1101-MS	I 166.0	BOOL	Sensor de Movimento ELV1101
260		ELV1101-PV1	IW 848	WORD	Temperatura 1 ELV1101
261		ELV1101-PV2	IW 850	WORD	Temperatura 2 ELV1101
262		ELV1101-PV3	IW 852	WORD	Temperatura 3 ELV1101
263		ELV1101-PV4	IW 854	WORD	Temperatura 4 ELV1101
264		ELV1101-PV5	IW 840	WORD	Temperatura 5 ELV1101
265		ELV1101-PV6	IW 842	WORD	Temperatura 6 ELV1101
266		ELV1101-STW1	QW 3	WORD	ELEVADOR EC30 80X17
267		ELV1101-ZS1	I 166.2	BOOL	Sensor de Desalinhamento 1 ELV1101
268		ELV1101-ZS2	I 166.4	BOOL	Sensor de Desalinhamento 2 ELV1101
269		ELV1101-ZS3	I 209.4	BOOL	Sensor de Desalinhamento 3 ELV1101
270		ELV1101-ZS4	I 209.6	BOOL	Sensor de Desalinhamento 4 ELV1101
271		ELV1101-ZSW1	IW 9	WORD	ELEVADOR EC30 80X17
272		ELV1102-ES1	I 170.0	BOOL	Botão de Emergência 1 ELV1102
273		ELV1102-ES2	I 190.0	BOOL	Botão de Emergência 2 ELV1102
274		ELV1102-LSH	I 189.0	BOOL	Sensor de Embuchamento ELV1102
275		ELV1102-MS	I 166.6	BOOL	Sensor de Movimento ELV1102
276		ELV1102-PV1	IW 768	WORD	Temperatura 1 ELV1102
277		ELV1102-PV2	IW 770	WORD	Temperatura 2 ELV1102
278		ELV1102-PV3	IW 772	WORD	Temperatura 3 ELV1102
279		ELV1102-PV4	IW 774	WORD	Temperatura 4 ELV1102
280		ELV1102-PV5	IW 776	WORD	Temperatura 3 ELV1102
281		ELV1102-PV6	IW 778	WORD	Temperatura 3 ELV1102
282		ELV1102-STW1	QW 5	WORD	ELEVADOR EC50 80X28(2)
283		ELV1102-ZS1	I 167.0	BOOL	Sensor de Desalinhamento 1 ELV1102
284		ELV1102-ZS2	I 168.0	BOOL	Sensor de Desalinhamento 2 ELV1102
285		ELV1102-ZS3	I 189.2	BOOL	Sensor de Desalinhamento 3 ELV1102
286		ELV1102-ZS4	I 189.4	BOOL	Sensor de Desalinhamento 4 ELV1102
287		ELV1102-ZSW1	IW 11	WORD	ELEVADOR EC50 80X28(2)
288		ELV1103-ES1	I 171.0	BOOL	Botão de Emergência 1 ELV1103
289		ELV1103-ES2	I 212.0	BOOL	Botão de Emergência 2 ELV1103
290		ELV1103-LSH	I 211.0	BOOL	Sensor de Embuchamento ELV1103
291		ELV1103-MS	I 167.4	BOOL	Sensor de Movimento ELV1103
292		ELV1103-PV1	IW 844	WORD	Temperatura 1 ELV1103
293		ELV1103-PV2	IW 846	WORD	Temperatura 2 ELV1103
294		ELV1103-PV3	IW 832	WORD	Temperatura 3 ELV1103

Press F1 to get Help.

Figura 54 – *Symbol Table*. Fonte: Original.

4.4 Programação

Para que a programação atenda os requisitos de automação, seção 2.1.1.7, foi escolhido usar a abordagem dos intertravamentos. Sendo assim, não será possível acionar um determinado equipamento caso o equipamento seguinte a ele não esteja ligado, essa abordagem, tem por consequência já atender ao segundo requisito da programação, a do desligamento em cascata. Além disso, foram criados sequencias de acionamento das sub-áreas, afim de facilitar na hora da operação.

Uma rota pode ser entendida pelo caminho no qual o produto irá passar, sendo assim

uma rota é definida pelos equipamentos e posições das válvulas direcionais. Usando o diagrama da Figura 55 como exemplo nota-se que exitem duas válvulas direcionais, um elevador, quatro esteiras, uma Máquina de limpeza e bloco "Envia Preparação", que representa um outro setor da planta. Assim pode-se destacar cada uma das 4 possíveis rotas Figura 56. Porém, no fim, independentemente da rota, o produto irá para o bloco de "Envia Preparação".

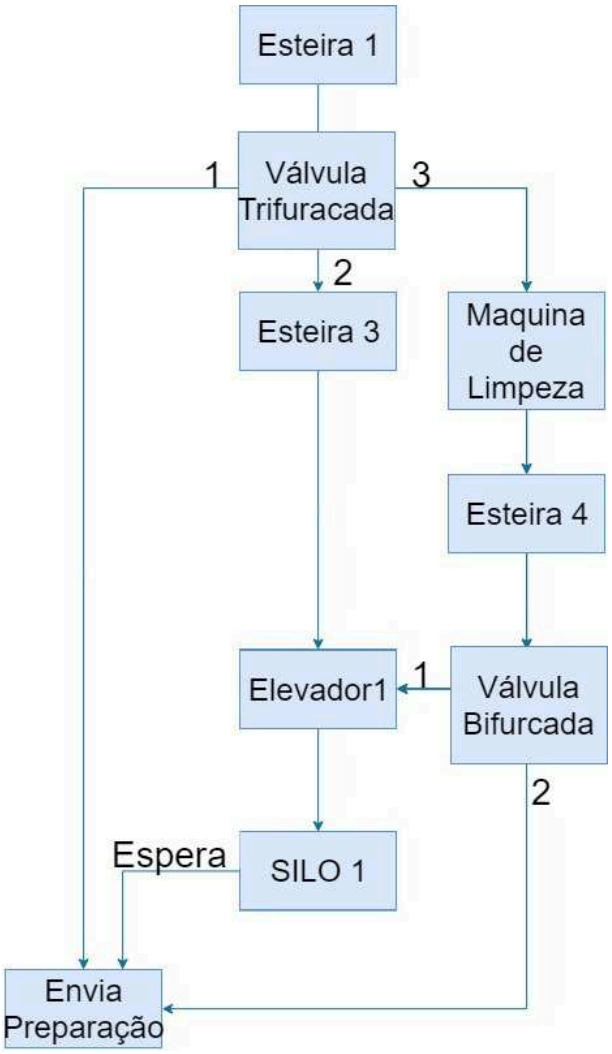


Figura 55 – Diagrama Exemplo. Fonte: Original.

O primeiro passo da programação, foi fazer os intertravamentos (*interlocks*) dos sensores e de rotas de cada um dos equipamentos, esses serão mostrados na Seção 4.4.1. Já a programação das rotas automáticas será mostrado na Seção 4.4.2.

4.4.1 Intertravamentos

Nos CMT's de todos os equipamentos, existem três tipos de blocos de intertravamentos, `Permit`, `Interlock` e o `Protect`. O `Permit` é um bloco de intertravamento que permite a a ligação do equipamento, ou seja, caso alguma condição não seja antes da ligação do

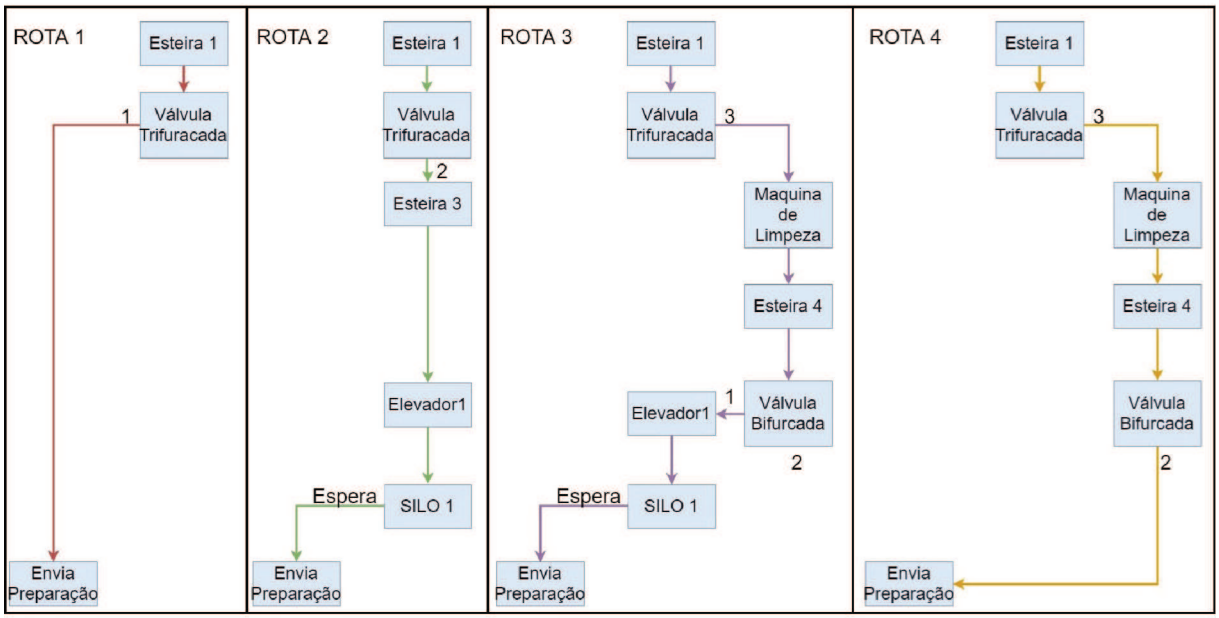


Figura 56 – Possíveis Rotas do Diagrama Exemplo Fonte: Original

equipamento ele não ligará, e após a ligação, caso algum sinal mude de valor ele não o desligará. Já para o bloco de `Interlock`, as condições devem ser atendidas a todo o momento. Por fim, o `Protect` funciona igual ao `Interlock`, mas em caso de as condições de intertravamento não sejam atendidas é necessário reiniciar o equipamento.

Dessa forma, os sensores devem ser conectados no bloco `Protect`, pois o sinal de algum deles indicará que há um problema físico com o equipamento. Na Figura 57 temos o `Protect` de uma esteira que tem três sensores de tipos diferentes: botão de emergência, sensor de embuchamento e chave de velocidade. Ainda nota-se que foi escolhido usar a lógica *AND*, assim todas as condições devem ser satisfeitas para que o `Protect` envie o sinal de lógico 1 indicando que a esteira pode entrar e ou estar em funcionamento.

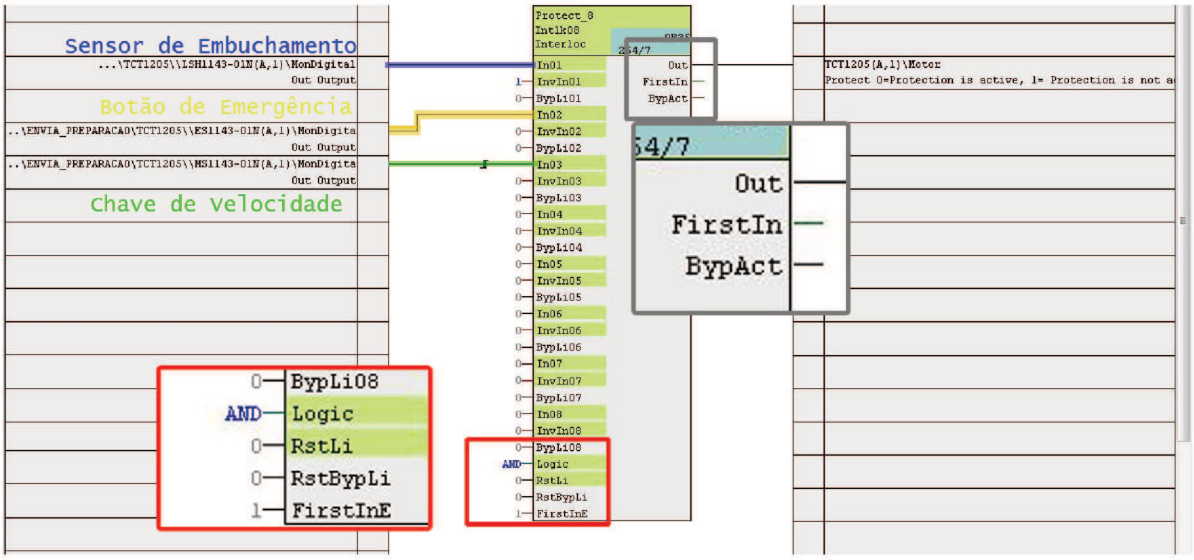


Figura 57 – `Protect` TCT11205. Fonte: Original.

No caso do `Protect` dos elevadores (Figura 58) que tem múltiplos sensores do mesmo tipo, foi necessário juntar cada um dos diferentes tipos de sensores em um *interlock's* próprios, assim permitindo ao operador navegar para as descrições dos intertravamentos nos *popup's* dos equipamentos.

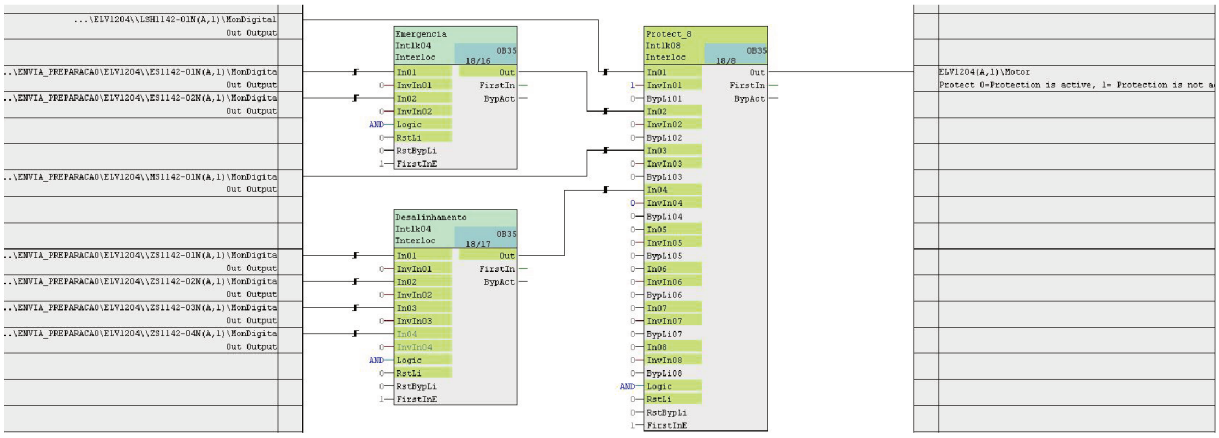


Figura 58 – `Protect` ELV1204. Fonte: Original.

Já no caso do bloco `Interlock` é onde estão os intertravamentos das rotas. No caso o *software* só poderá permitir ligar algum motor caso, o equipamento seguinte a ele esteja ligado, e em caso de desligamento deve-se desligar os equipamentos anteriores, de modo a tentar evitar o embuchamento dos outros equipamentos. Assim tomando o caso do elevador mostrado no P&ID da Figura 43, o **ELV1204** para que ele possa ligar, uma das esteiras TCT1205 ou TCT2101, dependendo da posição da válvula DYP1122, deverão estar ligadas. Assim o intertravamento das rotas pode ser escrito pela expressão booleana (4.1).

$$\text{Interlock} = (\text{TCT1205-FBR} \quad \text{AND} \quad \text{DYP1122-FB1}) \quad \text{OR} \quad (\text{TCT2101-FBR} \quad \text{AND} \quad \text{DYP1122-FB2}) \tag{4.1}$$

Dessa forma pode-se "converter" a expressão (4.1), no seguinte diagrama de blocos, Figura 59.

Nota-se que, onde deveria ser conectado o sinal de *feedback* da esteira TCT2101 há apenas uma legenda, isso ocorre porque o sinal desse equipamento vem de outra área e será conectado posteriormente.

Para montar as lógicas dos intertravamentos de rotas de todas as áreas, foi feito um fluxograma de toda a área do setor operacional. Parte desse fluxograma pode ser visto na Figura 60.

Cada cor no fluxograma, Figura 60, corresponde a uma sub-divisão do processo. Na Figura 61 pode-se ver o fluxograma do sub-setor Envia Preparação .

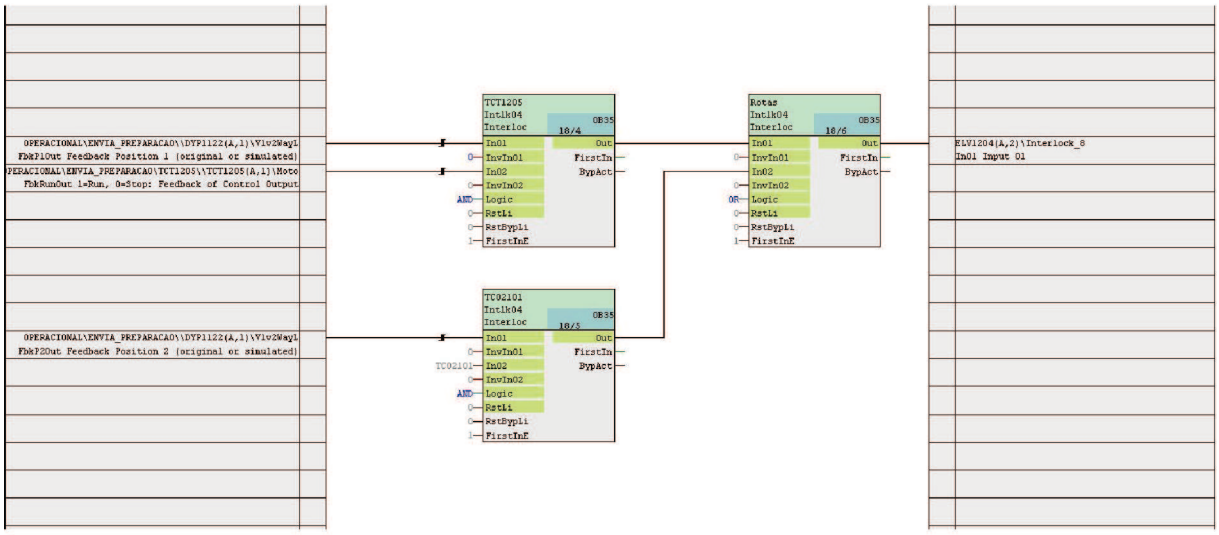


Figura 59 – *Interlock* ELV1204. Fonte: Original.

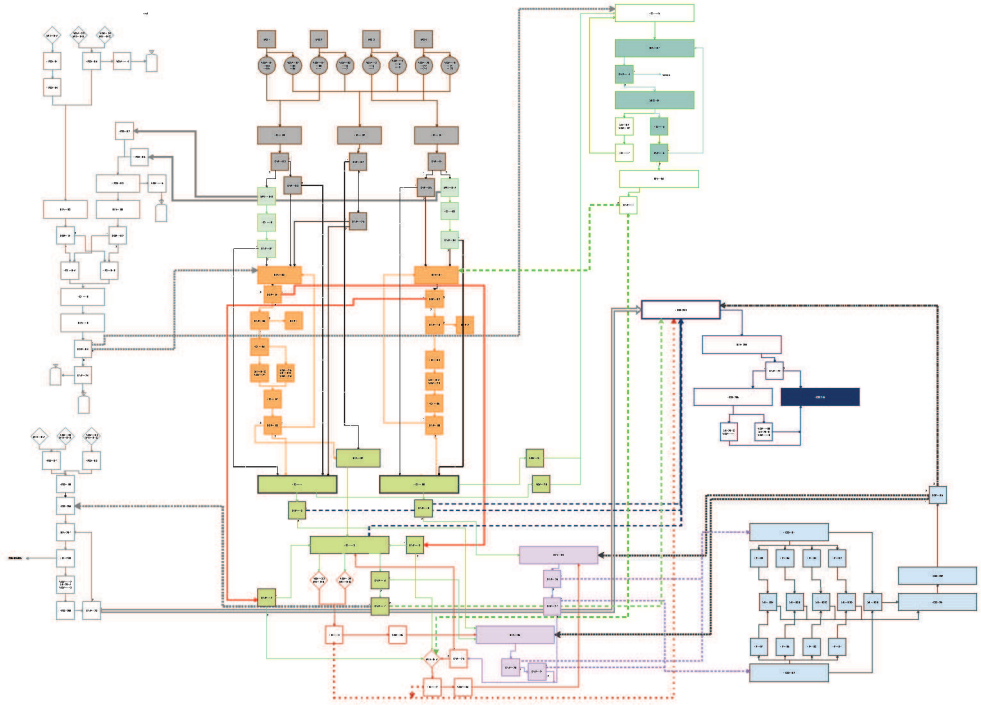


Figura 60 – Fluxograma completo da planta. Fonte: Original.

4.4.2 Rotas Pré-definidas

O PCS7 permite programar SFC's atraves de um recurso chamado *SFC type*. Os *SFC's type's*, são um ou mais diagramas SFC's que tem uma representação em único bloco de CFC. Sendo assim, temos um bloco em CFC, que tem um ou mais sequências SFC dentro. Cada um dos diversos SFC's, que um *SFC type* pode ter, é chamada de *control strategy*, cada uma dessas *control strategy's* recebe um número. No bloco a escolha de qual *control strategy* será executada é feita com a atribuição de um valor inteiro na entrada CS do bloco.

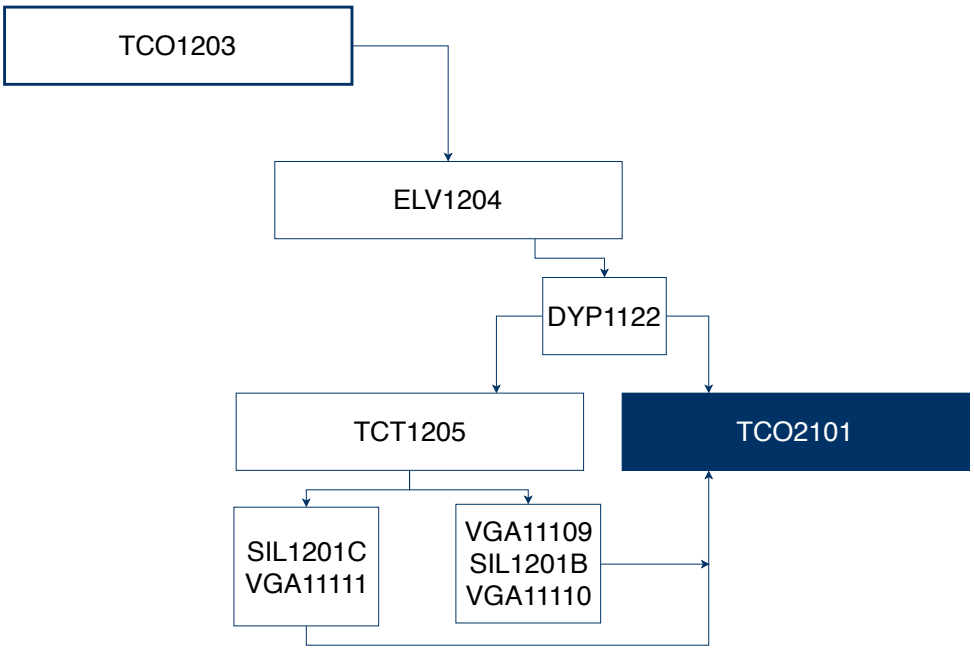


Figura 61 – Fluxograma Envia Preparação. Fonte: Original.

Sendo assim, foram programados *SFC's types*, que fazem o acionamento sequencial dos equipamentos de cada parte do processo. E cada *control strategy* desses *SFC's types*, representa uma das possíveis rotas da sub-área.

Por exemplo na Figura 62, temos o acionamento sequencial das 3 possíveis da rotas e do desligamento do setor Envia Preparação. Essas rotas são divididas de acordo com os possíveis destinos: Silo 1, Silo 2, preparação e equipamentos do setor desligado.

Sendo assim os 4 SFC's mostrados na Figura 62 estão programados dentro do mesmo bloco, Figura 63. A seleção das diferentes *control strategy's* é feita por um sinal do tipo inteiro na porta CS do bloco. O uso de *SFC types* traz mais uma vantagem, que na compilação das telas, a representação deles é gerado automaticamente, basta apenas posicioná-los na tela.

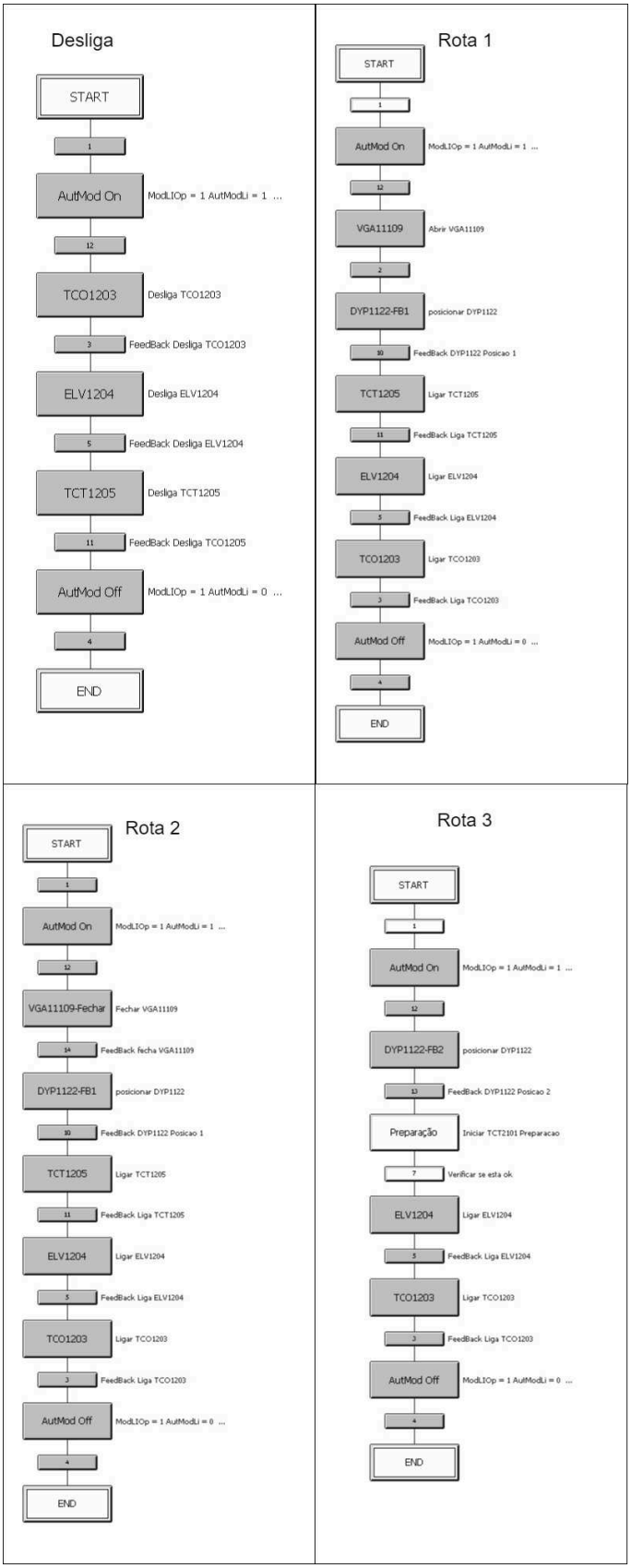


Figura 62 – SFC Envia preparação. Fonte: Original.

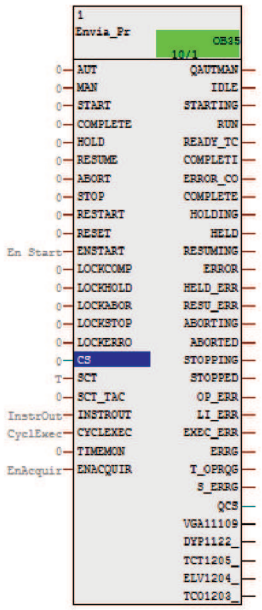


Figura 63 – Bloco do SFC Envia preparação. Fonte: Original.

4.5 Telas

Sobre a revisão das telas do supervisório, pode-se dizer que foi quase a mesma tarefa que foi feita no caso dos P&ID's. Sendo assim, a revisão dos instrumentos na tela foi um dos ponto principais, além de fazer as correções conforme os pedidos do cliente, correção de TAG's e posição de motores.

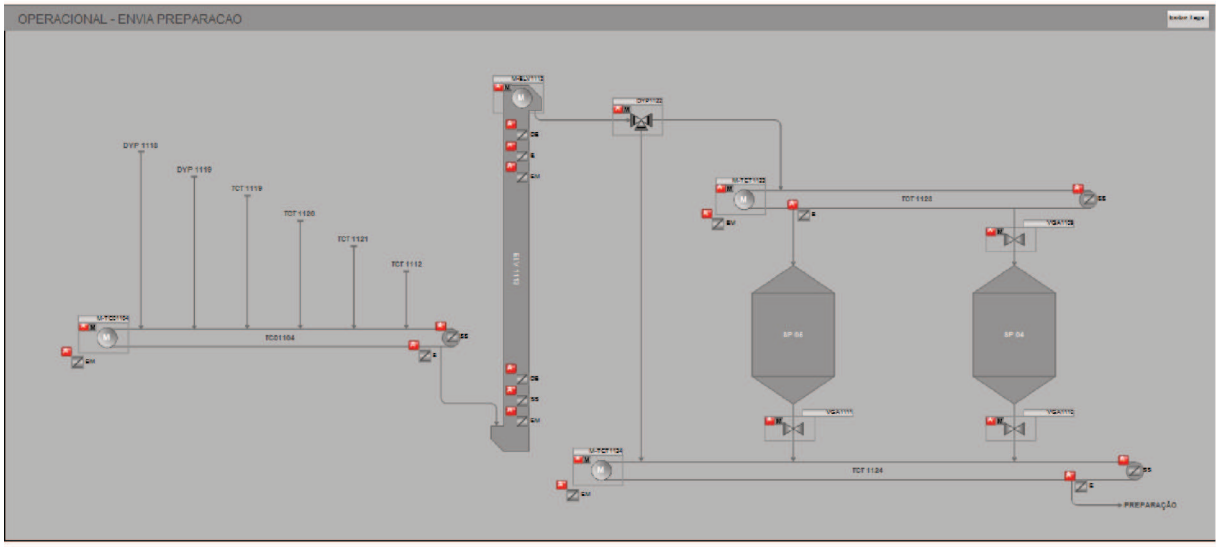


Figura 64 – Tela Envia_Preparação versão preliminar. Fonte: Original.

Ainda sobre essa versão da tela (Figura 64) foi pedido para que a posição dos motores nas esteiras esteja sempre considerando que o motor está puxando o produto, assim o motor deve ficar no lado da saída. A tela com essas correções pode ser observada na Figura 65.

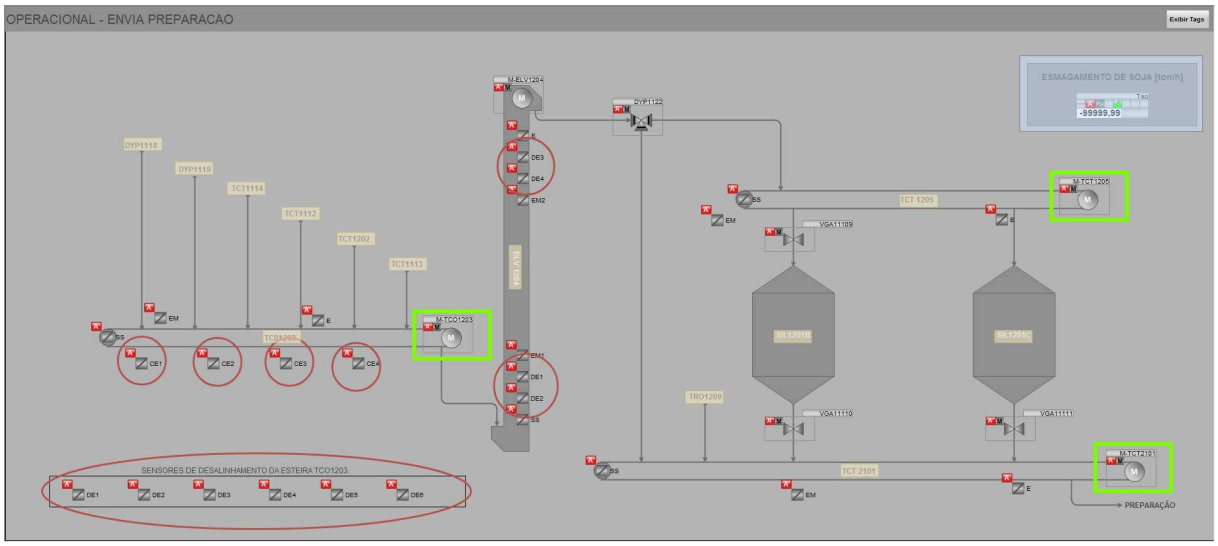


Figura 65 – Tela Envia_Preparação versão corrigida. Fonte: Original.

Na Figura 65, circulado em vermelho estão os instrumentos adicionados, enquadrado em verde estão os motores que mudaram de posição e marcado de amarelo estão as TAG's que foram revisadas.

Outro ponto que foi pedido para acrescentar nessa tela é o indicador de toneladas/hora (marcado de azul), porém até o atual momento, não se sabe como essa medição será feita ou calculada. O formato, as cores e a disposição dos equipamentos e instrumentos, foi feito seguindo o padrão da empresa e usando a técnica de desenvolvimento de telas e IHM's de alta performance. Em resumo essa técnica prega que a tela deve ser o mais simples possível e ter o mínimo de informações possíveis aparecendo na tela.

É importante explicar que, por exemplo, o motor da esteira ou elevador, indicado pela letra *M* com um círculo envolta, tem um janela *pop'up* onde é feito todos os comandos em modo manual.

Álém da tela Envia Preparação, foi necessário criar uma tela interna dos silos graneleiros, contendo os botões de controle dos motores das rosca varredora e do sistema de aeração (Figura 66).

Na direita da Figura 66, existe os ícones de acionamento dos motores separados por tipo, na esquerda está a representação do silo contendo a sua respectiva TAG e válvula de descarga. Como são cinco silos do mesmo tipo foi necessário criar mais quatro telas semelhantes a essa.

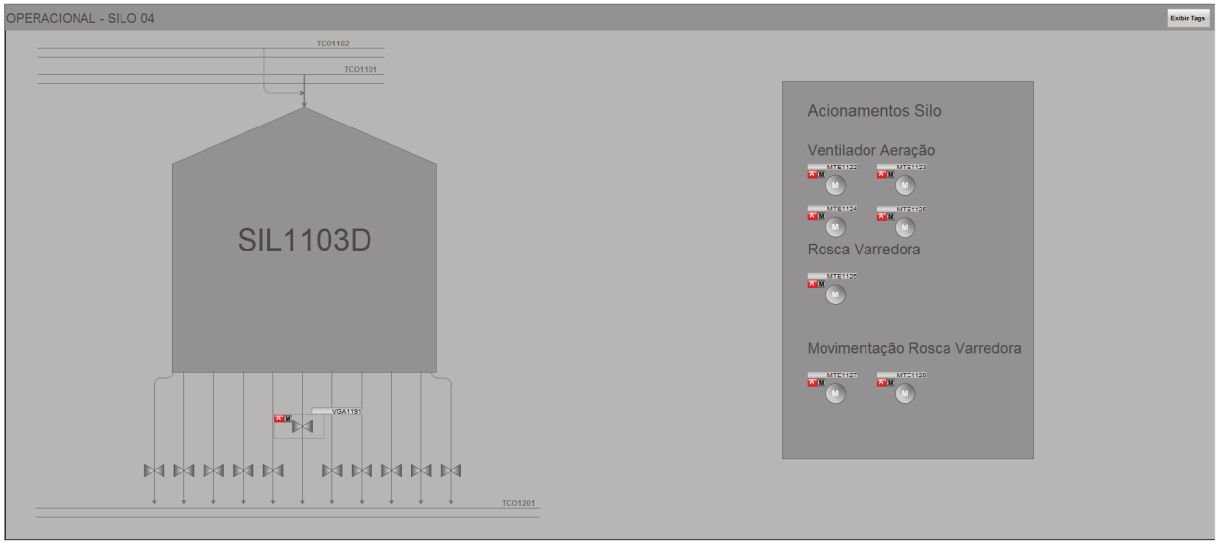


Figura 66 – Tela Armazenamento_Silo_4. Fonte: Original.

4.6 Simulação

Em paralelo com a criação da lógica de controle da planta foi feito uma simulação de todas as áreas do processo dentro do SIMIT. Esta seção tem por objetivo explicar como essas simulações foram feitas. Os testes serão mostrados na Seção 4.7. Cada uma das subseções a seguir tratarão de um nível diferente de simulação, sinal, componente, equipamento e processo. Essa separação pode ser entendida como:

- Sinal: o sinal do sensor ou sinal de comando recebido pelo inversor de frequência que aciona um motor;
- Componente: a dinâmica do acionamento do motor, ou a dinâmica do sensor;
- Equipamento: a simulação da dinâmica da esteira, representando como que ela e o quê ela faz com o produto que entra nela. Ele entra no equipamento e saí depois de um determinado tempo.
- Processo: simulação de todo um setor, ou sub-área. Englobando a ligação entre os diferentes equipamentos.

O diagrama dessa estrutura de simulação pode-ser observado na Figura 67.

Vale ressaltar que a simulação aqui apresentada tem por objetivo principal simular os eventos necessários para checar as lógicas implementadas na programação (Seção 4.4), e não, necessariamente, as dinâmicas exatas dos equipamentos e processos. Isso poderá ser visto mais aprofundadamente nas próximas sub-seções.

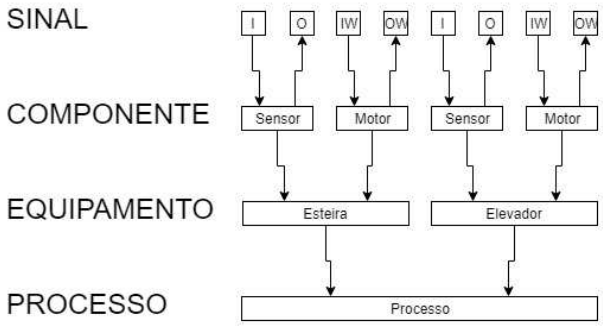
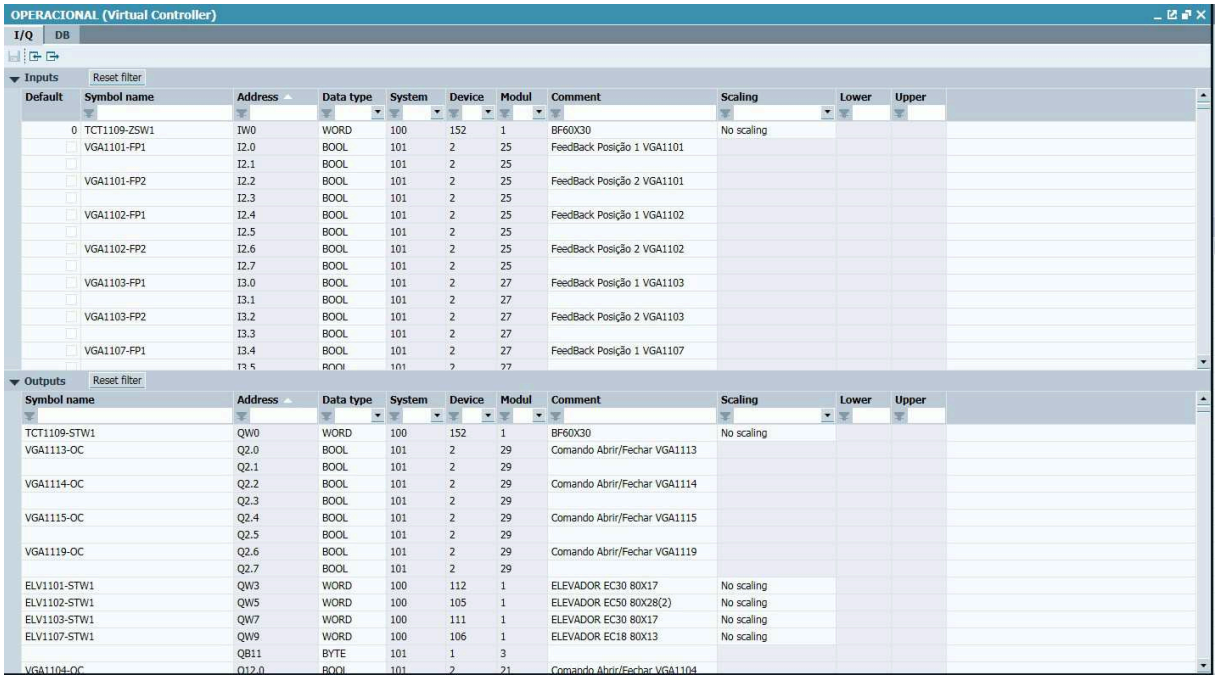


Figura 67 – Diagrama níveis de Simulação SIMIT. Fonte: Original.

4.6.1 Simulação do Nível de Sinal

A simulação do nível de processo é feita, simplesmente, com a comunicação do SIMIT com o PCS7. Sendo assim o primeiro passo da programação da simulação foi iniciar o CLP simualado do SIMIT, criado com a ferramenta Virtual Controller (VC), e fazer a comunicação do mesmo com o PCS7, esse tipo de simulação é chamado de *coupling* (explicado na Seção 3.3), dessa forma montando a simulação no nível de sinal.


OPERACIONAL (Virtual Controller)

I/Q | DB

Inputs — Reset filter

Default	Symbol name	Address	Data type	System	Device	Modul	Comment	Scaling	Lower	Upper
0	TCT1109-ZSW1	IW0	WORD	100	152	1	BF60X30	No scaling		
	VGA1101-FP1	I2.0	BOOL	101	2	25	FeedBack Posição 1 VGA1101			
		I2.1	BOOL	101	2	25				
	VGA1101-FP2	I2.2	BOOL	101	2	25	FeedBack Posição 2 VGA1101			
		I2.3	BOOL	101	2	25				
	VGA1102-FP1	I2.4	BOOL	101	2	25	FeedBack Posição 1 VGA1102			
		I2.5	BOOL	101	2	25				
	VGA1102-FP2	I2.6	BOOL	101	2	25	FeedBack Posição 2 VGA1102			
		I2.7	BOOL	101	2	25				
	VGA1103-FP1	I3.0	BOOL	101	2	27	FeedBack Posição 1 VGA1103			
		I3.1	BOOL	101	2	27				
	VGA1103-FP2	I3.2	BOOL	101	2	27	FeedBack Posição 2 VGA1103			
		I3.3	BOOL	101	2	27				
	VGA1107-FP1	I3.4	BOOL	101	2	27	FeedBack Posição 1 VGA1107			
		I3.5	BOOL	101	2	27				

Outputs — Reset filter

Symbol name	Address	Data type	System	Device	Modul	Comment	Scaling	Lower	Upper
TCT1109-STW1	QW0	WORD	100	152	1	BF60X30	No scaling		
VGA1113-OC	Q2.0	BOOL	101	2	29	Comando Abrir/Fechar VGA1113			
	Q2.1	BOOL	101	2	29				
VGA1114-OC	Q2.2	BOOL	101	2	29	Comando Abrir/Fechar VGA1114			
	Q2.3	BOOL	101	2	29				
VGA1115-OC	Q2.4	BOOL	101	2	29	Comando Abrir/Fechar VGA1115			
	Q2.5	BOOL	101	2	29				
VGA1119-OC	Q2.6	BOOL	101	2	29	Comando Abrir/Fechar VGA1119			
	Q2.7	BOOL	101	2	29				
ELV1101-STW1	QW3	WORD	100	112	1	ELEVADOR EC30 80X17	No scaling		
ELV1102-STW1	QW5	WORD	100	105	1	ELEVADOR EC50 80X28(2)	No scaling		
ELV1103-STW1	QW7	WORD	100	111	1	ELEVADOR EC30 80X17	No scaling		
ELV1107-STW1	QW9	WORD	100	106	1	ELEVADOR EC18 80X13	No scaling		
	QB11	BYTE	101	1	3				
VGA1104-OC	Q12.0	BOOL	101	2	21	Comando Abrir/Fechar VGA1104			

Figura 68 – *Coupling* entre o projeto do PCS7 e o SIMIT. Fonte: Original.

4.6.2 Simulação do Nível de Componente

Em seguida foi colocada a biblioteca do SIMIT quem contém a simulação de cada um dos CMT's. Assim ela foi, posteriormente, usada para criar em larga escala as páginas de simulações de cada um dos elementos, através de um *plugin* de exportação do *charts* do PCS7. Essa biblioteca pode ser observada na Figura 69.

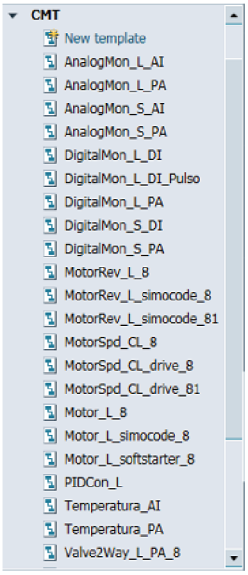


Figura 69 – *Templates* do SIMIT da Biblioteca CMT. Fonte: Original.

Nota-se que os elementos desta biblioteca tem o mesmo nome dos elementos dos CMT's (Figura 45) o nome igual significa que os objetos são relacionados. Dessa forma o *plugin* de exportação identificará qual *chart* corresponde a qual modelo de simulação.

Sobre esses modelos/*templates* do SIMIT eles tem uma janela de atribuição de valor (Figura 70). Quando esses *templates* forem instanciados, a janela de atribuição irá aparecer.

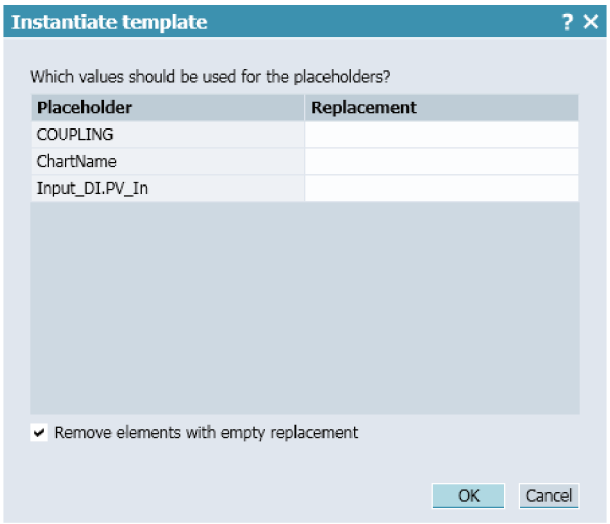


Figura 70 – Atribuição de variáveis na criação a partir de *templates* SIMIT. Fonte: Original.

Inicialmente não existia nesta biblioteca (Figura 69) a simulação da dinâmica do sensor de rotação. Para isso, foi criado o bloco `DigitalMon_L_DI_Pulso`, que faz a simulação do sensor de pulso, já explicada na seção 4.2. Sendo assim, o mesmo precisava ser criado no SIMIT. Na Figura 71 mostra o *template* da simulação do sensor em questão.

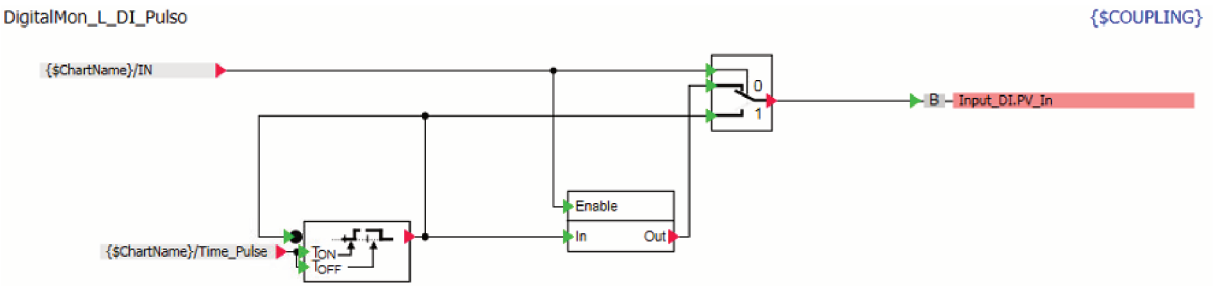


Figura 71 – *Templates* do SIMIT `DigitalMon_L_DI_Pulso`. Fonte: Original.

Em resumo a simulação funciona como um oscilador que tem seu tempo de oscilação definido pela variável Time_Pulse, essa virá posteriormente das simulações de nível de equipamento. Essa oscilação tem uma memória que ao ser desabilitada, grava a sua última saída, assim no momento de parada pode-se manter a saída tanto em 1 quanto em 0. Os níveis superiores da simulação deverão emitir um sinal informando que a equipamento relacionado ao sensor está ligado.

4.6.3 Simulação do Nível de Equipamento

A simulação da esteira transportadora e do elevador podem ser observados nas Figuras 72 e 73.

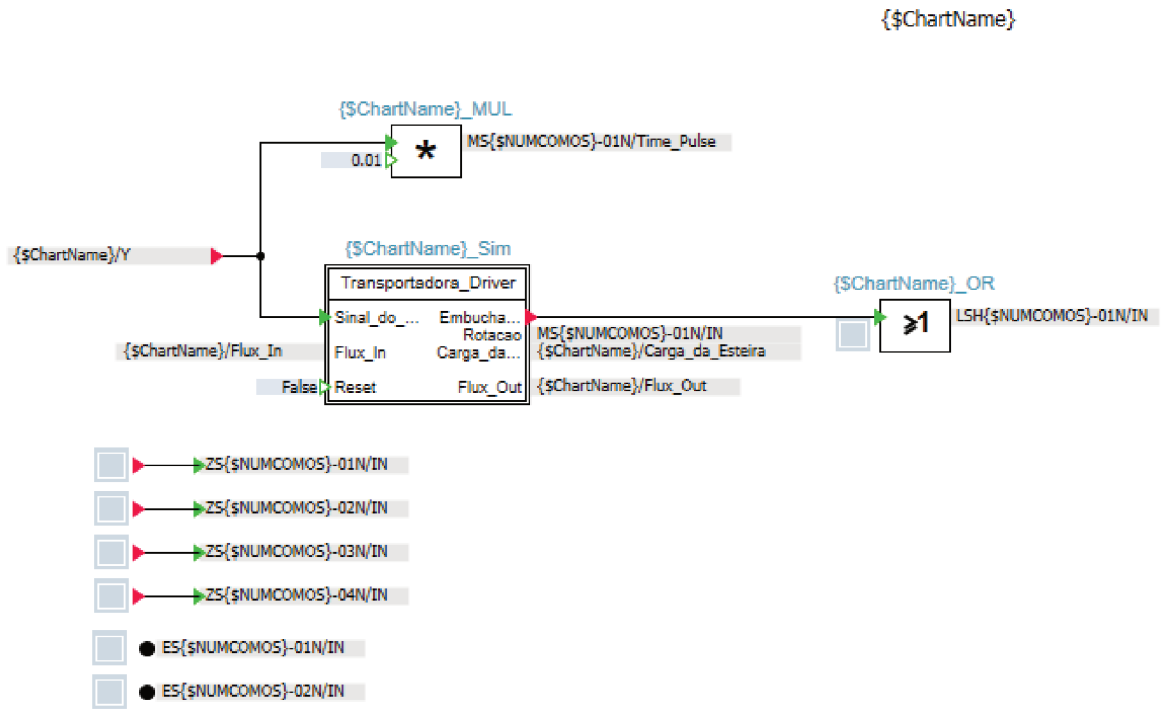


Figura 72 – *Templates* do SIMIT Elevador caneco. Fonte: Original.

Percebe-se que a parte principal da dinâmica simulada nos dois equipamentos está dentro do bloco `Transportadora_Driver`. Um aspecto importante dessa simulação são

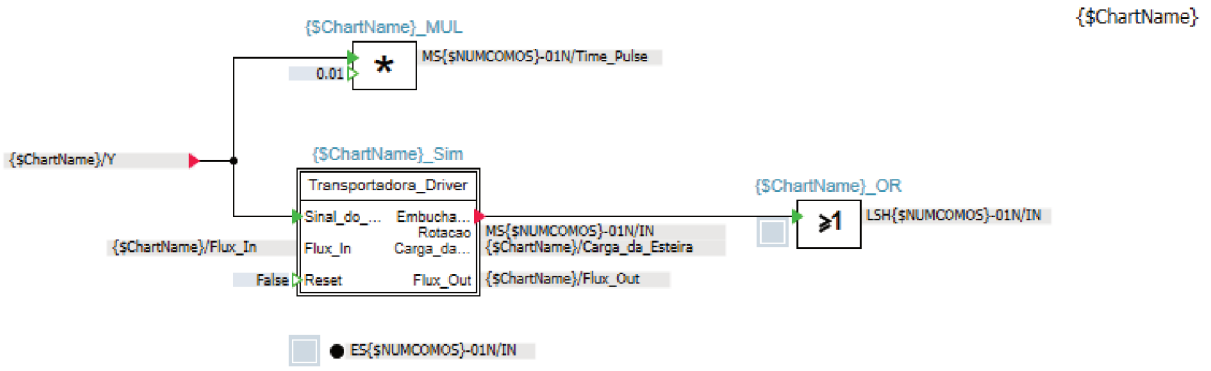


Figura 73 – *Templates* do SIMIT Esteira Transportadora. Fonte: Original.

os botões que simulam, e devem ser comandados manualmente, os *status* dos botões de emergência e sensor de desalinhamento. Esse último não foi encontrado uma forma simples de simulação.

O bloco `Transportadora_Driver` é um tipo especial de bloco, que foi criado para essa aplicação, com o uso da ferramenta de criação de *Macros* do SIMIT. Sendo assim a funcionamento interno desse bloco pode ser observado na Figura 74.

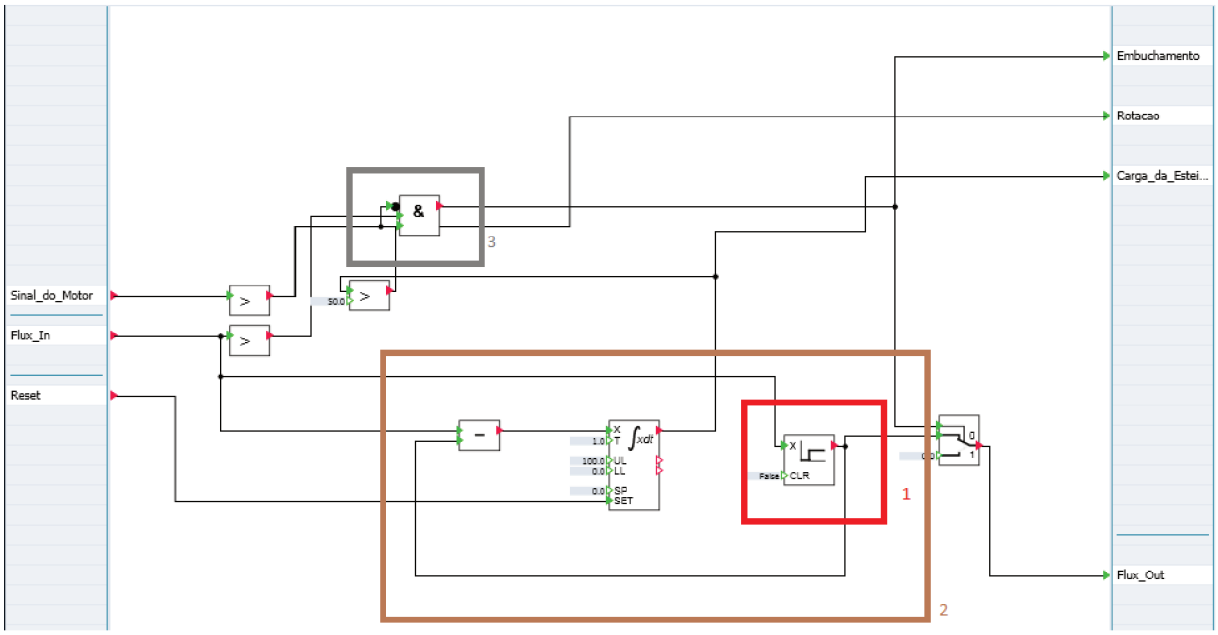


Figura 74 – *Macro* `Transportadora_Driver`. Fonte: Original.

Os itens circulados na Figura 74 são:

1. (Vermelho) Bloco do atraso de transporte, que simula a dinâmica básica das transportadoras;
2. (Marrom) Integrador responsável por calcular a carga da esteira;
3. (Cinza) Lógica booleana que determina se a transportadora está ou não em estado de embuchamento.

Sobre o bloco integrador, vale explicar que a cada ciclo da simulação (Δt) o integrador adiciona o valor de sua entrada a sua saída, ou seja ele irá integrar a entrada a cada instante de tempo.

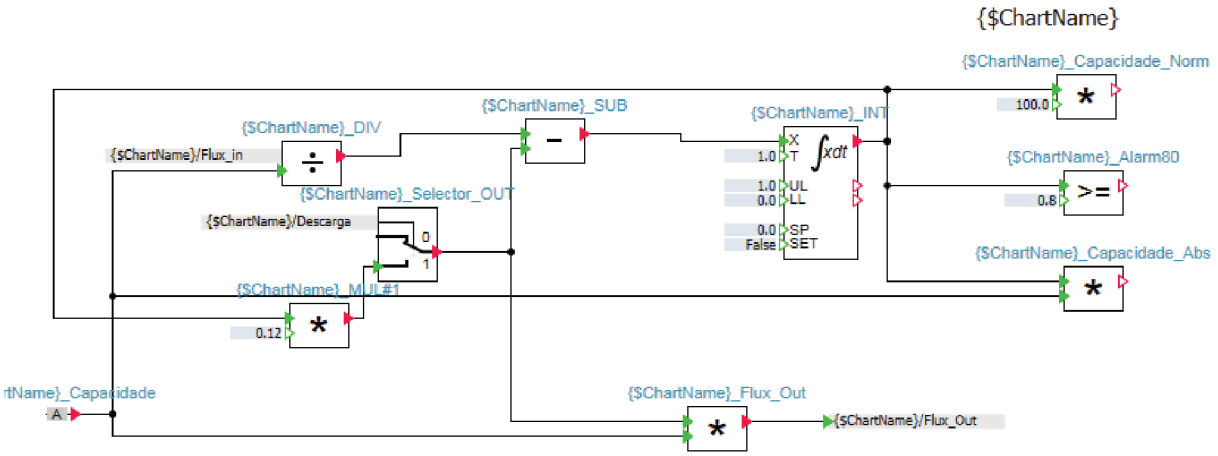


Figura 75 – *Templates* do SIMIT Silo Pulmão. Fonte: Original.

Na Figura 75 mostra-se o *template* de simulação do Silo. Esse, funciona da seguinte forma, o divisor `{$ChartName}_DIV` recebe a o fluxo de entrada do silo e faz a divisão, já que esse sinal de entrada é uma carga em que varia no tempo ($\frac{carga}{\Delta t}$), assim usa-se o divisor para transformar isso em um valor normalizado por unidade de tempo ($\frac{\text{valor de 0 a 1}}{\Delta t}$). Em seguida o `{$ChartName}_SUB` calcula a entrada do integrador (`{$ChartName}_INT`), caso o resultado dessa operação, subtração, for negativo, estará saindo mais produto do que entrando, logo uma dinâmica de descarga, e vice-versa. Mas, para que a dinâmica de descarga ocorra precisa que o sinal booleano de entrada, `{$ChartName}Descarga` esteja em 1. Nota-se que existem três saídas do *template* a primeira, `{$ChartName}_Capacidade_Norm`, será a saída do bloco em porcentagem normalizada, a saída 2 é a saída do bloco `{$ChartName}_ALARM80`, que é um sinal booleano que é 1 caso a capacidade do silo chegue em 80% de sua capacidade, por fim a última saída é a do bloco `{$ChartName}_Capacidade_ABS`, que é a carga absoluta do silo em toneladas.

A simulação das Moegas (Figura 76) tem a seguinte dinâmica: O sinal que vem do conector A, é o sinal de fluxo de entrada de produto, os multiplicadores representam a abertura proporcional das válvulas através do sinal de porcentagem da abertura das mesmas, porém essas válvulas não são comandadas analogicamente. As válvulas também podem controlar para qual das diferentes saídas pode ir o escoamento do produto, se para a saída 1 ou a 2, no caso as três válvulas de final impar direcionam para a saída 1 e as outras três para a saída 2.

Outra função interessante que esse *template* possui é a de fazer o cálculo da proporcionalidade das saídas, exemplo, caso esteja aberto só uma das válvulas, todo o fluxo de saída passará por ela, caso duas válvulas estejam abertas, em cada uma delas passará

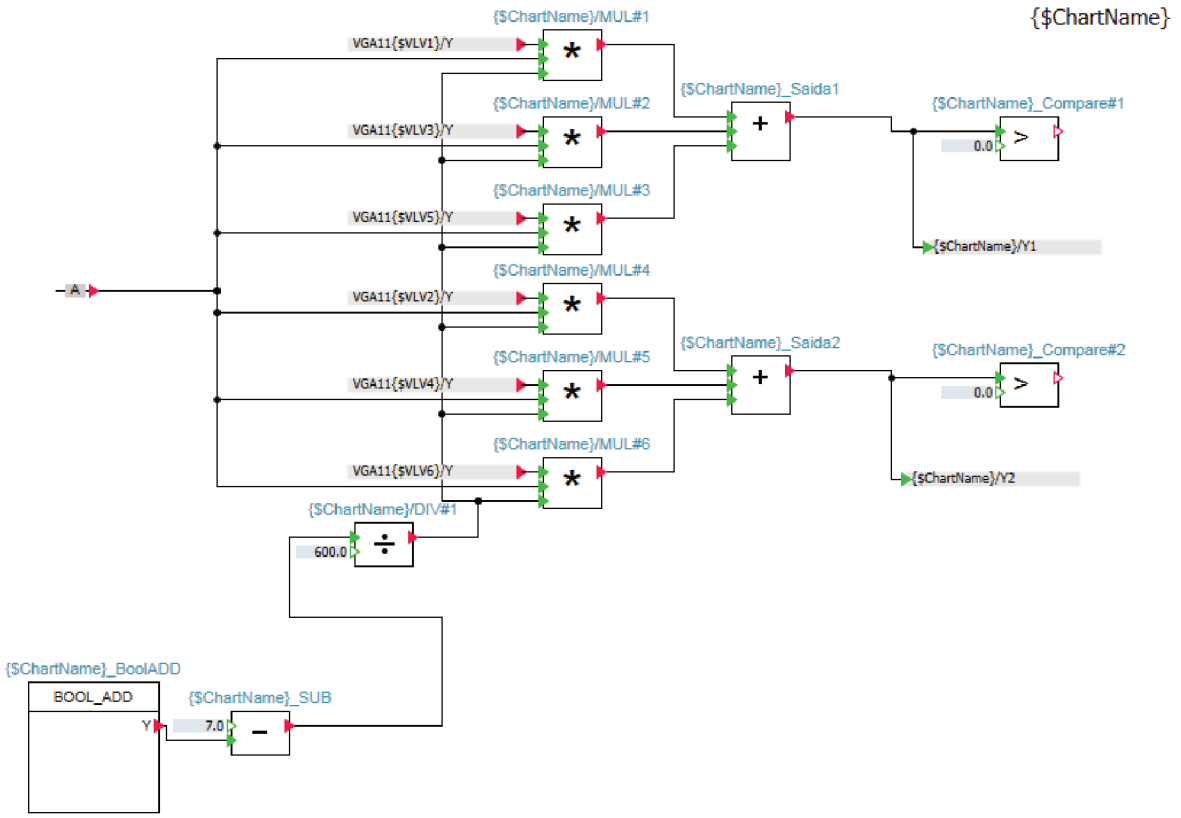


Figura 76 – *Templates* do SIMIT Moegas. Fonte: Original.

metade do fluxo e assim por diante.

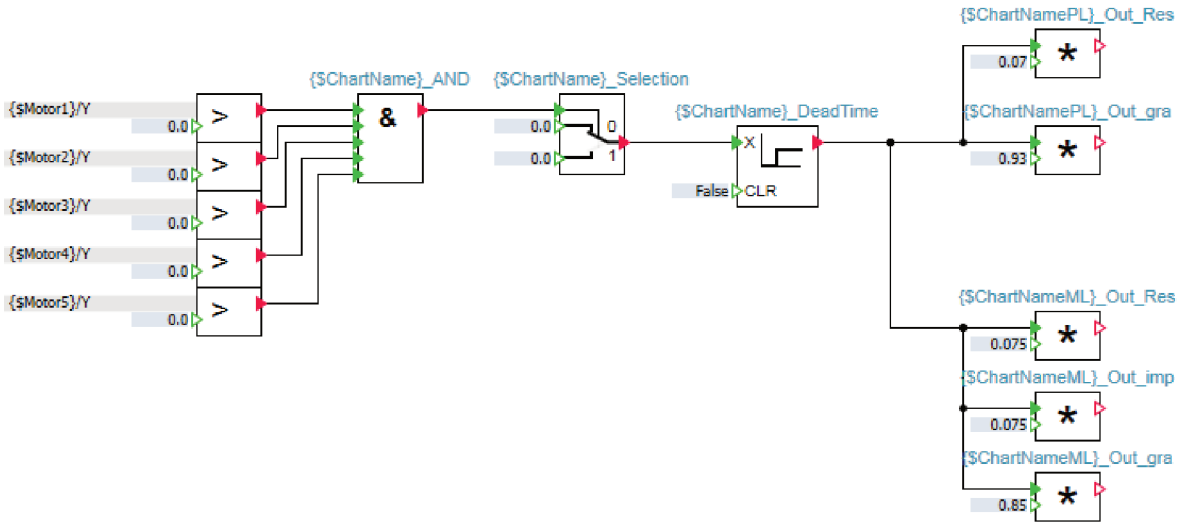


Figura 77 – *Templates* do SIMIT Máquinas de Limpeza e Pré-Limpeza. Fonte: Original.

O *template* das máquinas de limpeza e pré-limpeza é mais simples. A dinâmica é simulada apenas por um atraso, que é habilitado caso todos os motores estejam ligados. Já na saída dessa simulação, temos que o fluxo é divido em várias saídas, grãos, resíduos e impurezas, esses dois últimos tiram uma parcela de (x)% do fluxo de entrada, o restante

é considerado grão. Pelas máquinas terem um comportamento parecido, foi optado por fazer apenas um *template* para as duas. As únicas diferenças entre os equipamentos, no ponto de vista dessa simulação, de eventos, é a quantidade de motores e a quantidade de saídas, as de limpeza tem 5 motores e as três saídas, já as máquinas de pré-limpeza tem 3 motores e saída e grão e resíduos. A escolha e qual máquina será simulada é feita apenas na declaração, caso máquina de pré-limpeza, apenas declara-se 3 motores, e caso máquina de limpeza declara-se os 5 motores. A interface de atribuição de variáveis desses equipamentos podem ser vistos nas Figuras 78 e 79, bem como a diferença na declaração entre uma máquina de limpeza e pré-limpeza.

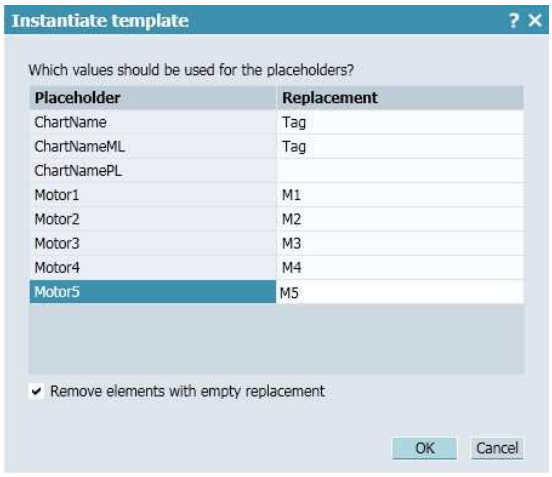


Figura 78 – Instancia das variáveis caso Máquinas de Limpeza. Fonte: Original.

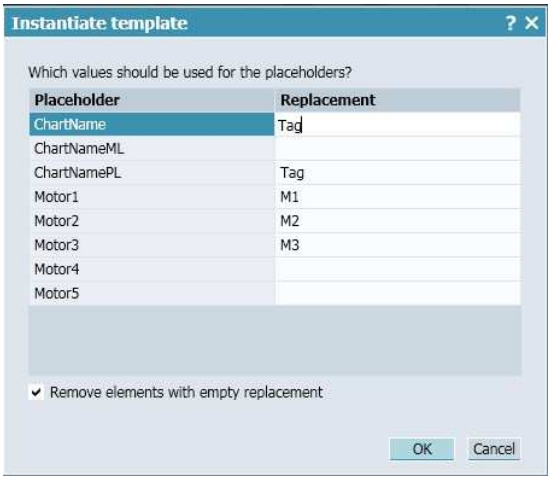


Figura 79 – Instancia das variáveis caso Máquinas de Pré-Limpeza. Fonte: Original.

Os *templates* criados para as Moegas, transportadoras, elevadores e máquinas, citados anteriormente, foram unidos em uma biblioteca, para fazer a criação automatizada da maior parte da simulação. O mecanismo de criação em massa usado funciona através de importação das planilhas em `.xlsx`. Essa tabela, além de criar os elementos dos *templates* faz o preenchimento dos valores das variáveis. Um exemplo dessa tabela pode ser visto na Tabela 2.

Tabela 2 – Tabela exemplo de criação automatizada SIMIT

Nome do Template	Hierarquia	Tag	Nome Variável 1	Valor Variável 1	Nome Variável 2	Valor Variável 2
Elevador	ENVIA_PREPARACAO	ELV1204	NUMCOMOS	1142	ChartName	ELV1204
Transportadora_fun_Saida_Central	ENVIA_PREPARACAO	TCT1205	NUMCOMOS	1143	ChartName	TCT1205
Silo	ENVIA_PREPARACAO	SIL1201C	Capacidade_Silo	1327	ChartName	SIL1201C

Pela planta ter apenas um secador, não foi optado por criá-lo via *template*. A simulação do secador é basicamente um atraso, porém existem integradores na entrada e na saída, para simular uma dinâmica de acumulo de grãos. Como a ideia dessa simulação é considerar a carga, e já que o secador retira cerca de 5% da umidade do grão, foi colocado um multiplicador no antes do atraso, para simular essa perda mássica.

Já a simulação da temperatura do secador, foi uma das mais complexas. A ideia era fazer que a fornalha tivesse uma dinâmica minimamente parecida com a descrita pelo fabricante do secador.
A Figura 80 mostra a resposta da temperatura (^oC) no tempo(min) da fornalha. Vale ressaltar que esse é o único sinal analógico usado em controle no processo, porém, pelo descritivo do fabricante, esse seria controlado é apenas por um controle ON/OFF.

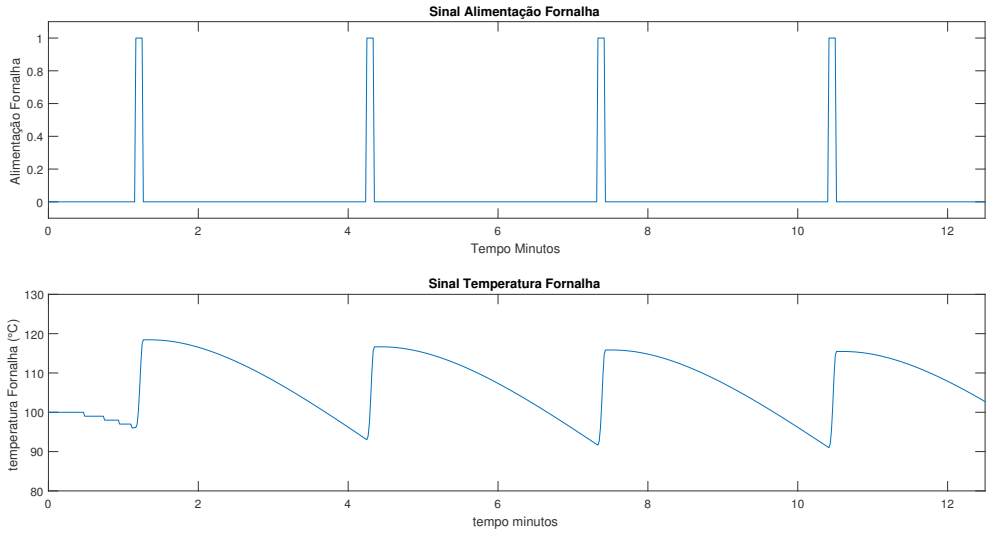


Figura 80 – Dinâmica da temperatura da Fornalha. Fonte: Original.

Sobre o descritivo do comportamento da fornalha, informava que deveria ser alimentado a fornalha por 5 s e mantê-la desligada por no mínimo 1 min. Os 5 s de alimentação deveria ser o suficiente para fazer a fornalha aquecer e passar da temperatura de *setpoint*, no caso da simulação, Figura 80 , o *setpoint* seria 100 . Já o resfriamento, o tem o tempo de no mínimo 1 min para diminuir no máximo 15 em relação ao *setpoint*. A simulação do sinal da fornalha acima, foi feita usando *scripts*, onde a o sinal de ligar a fornalha era mantido por 5 s e desligado durante 1 min ou até que a temperatura baixa-se de 95 . Dessa forma, o *script* simularia a ação do controlador em malha fechada.

4.6.4 Simulação Nível de Processo

As simulações de nível de processo tem muitas informações consideradas sigilosas pelas empresas, assim será apresentado apenas de uma área. Porém antes de mostrar a simulação, irá ser explicado os blocos criados com animações que representam o funcionamento dos equipamentos simulados e descritos na subseção anterior, Seção 4.6.3, bem como os blocos das válvulas direcionais.

Iniciando com o bloco das válvulas bifurcadas e trifurcadas. Esses foram feitos usando o *component type editor* (editor de típicos de componentes, já que eram elementos simples e muito utilizados, nesse setor). Assim na Figura 81 pode ser observado os blocos e

suas animações de posição resultado da animação da válvula bifurcada. Essas animações indicam o fluxo do produto para as saídas.

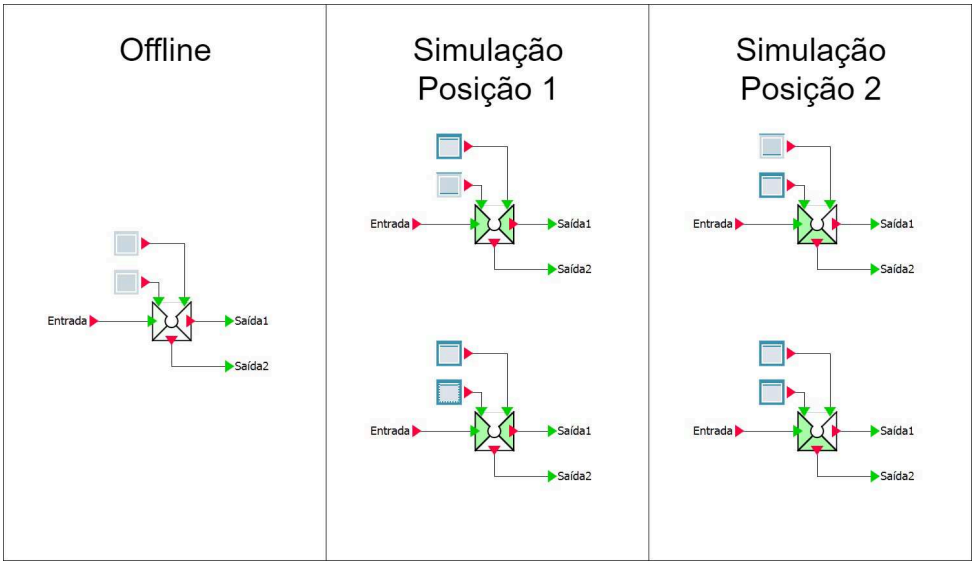


Figura 81 – Típico válvula 2 vias e suas animações. Fonte: Original.

Na Figura 81 nota-se que a válvula está conectada a dois botões (quadrados em cinza na parte superior da válvula), os botões circulados de azul indicam que os acionamentos da válvula estão recebendo o valor lógico 1. Lembrando que o comportamento desses blocos é feito através de programação escrita, baseada em C. Sendo assim a programação delas foram feitas para atender a seguintes dinâmicas: no caso da válvula bifurcada ela recebe dois sinais booleanos de comando, cada um dos sinais representa uma posição, se eles ficaram os dois no mesmo valor lógico a válvula mantêm a última posição.

No caso da válvula trifurcada, ela recebe 4 sinais, que no caso simulam as posições de dois cilindros pneumáticos. Sendo assim, divide-se os 4 sinais em 2 pares, cada par representando os comandos que os cilindros irão receber. Assim se os dois cilindros estiverem em recuo, a válvula direciona a saída central, caso seja acionado um dos cilindros ele direcionará a saída de um dos lados, os dois cilindros não devem estar em avanço ao mesmo tempo. Para esse caso na simulação foi considerada que a válvula mantém a ultima posição.

A animação da esteira transportadora pode ser observada na Figura 82

Assim como a animação do elevador pode ser observado na Figura 83.

Esses animação (Figuras 82 e 83), transportadora e elevador, tem todo o código igual, a única mudança é no formato da animação. Enquanto a simulação está ligada e os equipamentos (elevador e ou transportadora) estão ligados o quadrado verde move-se no sentido que o produto deveria fazer. Percebe-se que essas animações, Figura 82 e 83, tem um indicador de falha, para esse indicador funcionar corretamente a animação recebe os sinais dos sensores do equipamento. Durante a simulação, pode-se dar um duplo click no

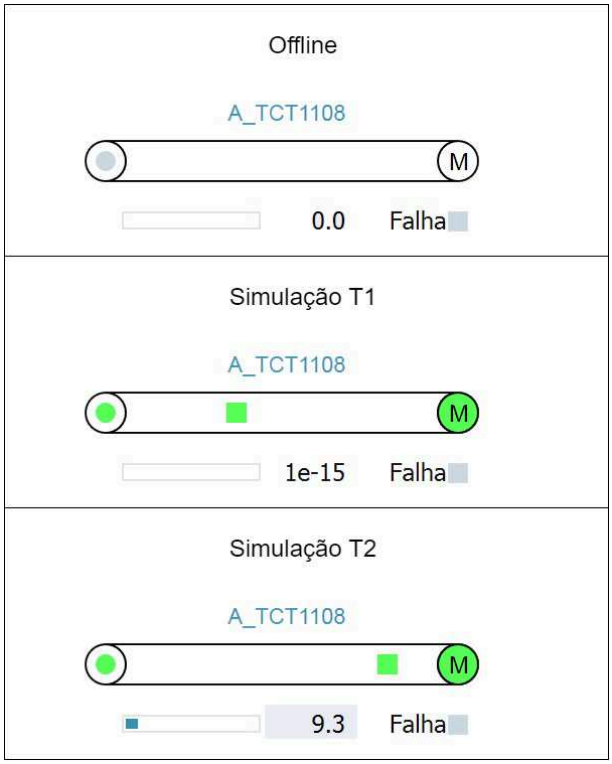


Figura 82 – Animação Esteira e suas animações. Fonte: Original.

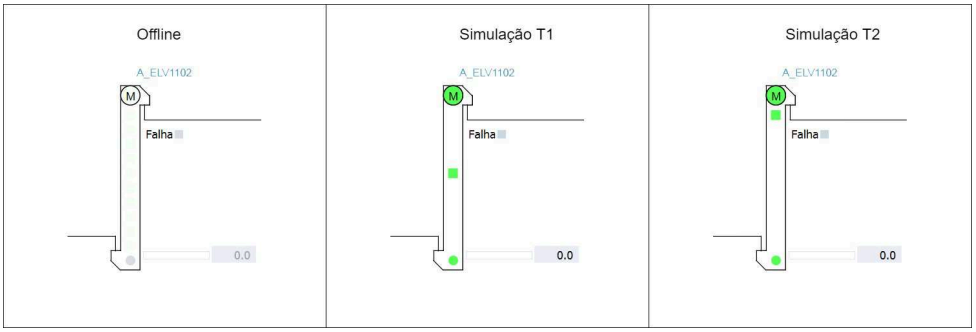


Figura 83 – Animação Elevador e suas animações. Fonte: Original.

ícone do equipamento e observar os status individuais dos sinais (Figura 84) assim, por exemplo, pode-se descobrir os motivos das falhas.

No caso dos silos (Figura 85) tem-se que a animação tem somente três indicadores, o de nível analógico, nível em 80% e um logo em baixo indicando se a válvula de descarga do silo está aberta ou fechada.

A animação das máquinas de pré-limpeza e sua janela de indicação interna podem ser observadas na Figura 86.

Já a animação das máquinas de limpeza pode ser vista na Figura 87, bem com sua janela de indicação interna, que pode ser observada na Figura 87.

Usando essas animações e fazendo os *links* entre os equipamentos pode-se montar as 9 telas de simulação das 9 áreas do setor operacional. Para fazer os testes das simulações dessas áreas, foram criados *scripts*, que iniciam os motores, redirecionam as válvulas

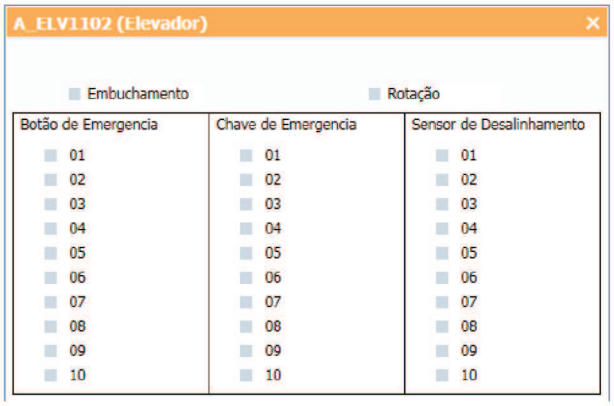


Figura 84 – Indicador de status dos sensores da animação do elevador. Fonte: Original.

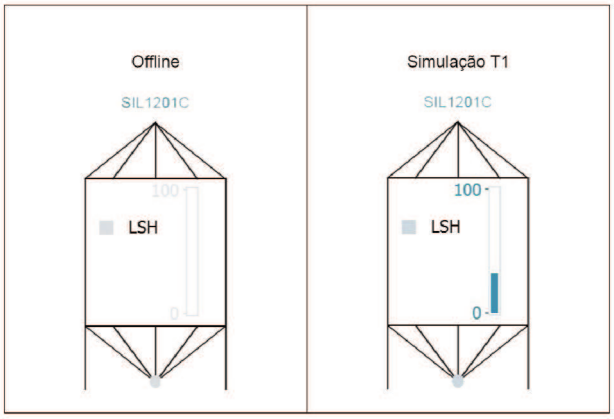


Figura 85 – Animação Silo e suas animações. Fonte: Original.

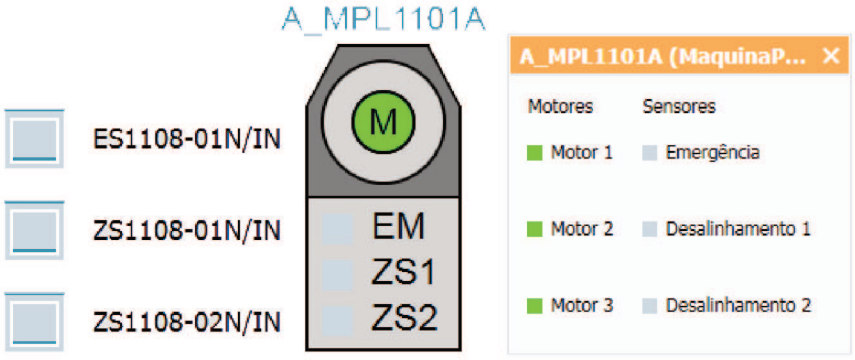


Figura 86 – Animação máquina de pré-limpeza e informações internas. Fonte: Original.

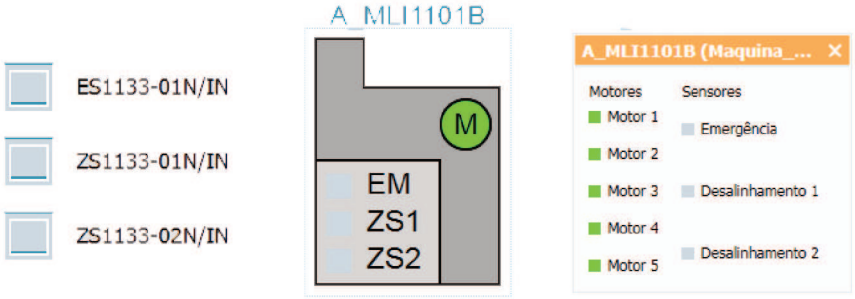


Figura 87 – Animação máquina de limpeza e informações internas. Fonte: Original.

e colocam carga nos equipamentos, ou seja, os *scripts* simulam a operação da planta. Na Figura 88 mostra-se a simulação em nível de processo do setor Envia Preparação, o mesmo mostrado na Seção 4.1, assim pode-se dizer que essa camada de simulação é, de certa forma, o PI&D "simulado" já que possui a mesma disposição física do P&ID, e quase a mesma aparência.

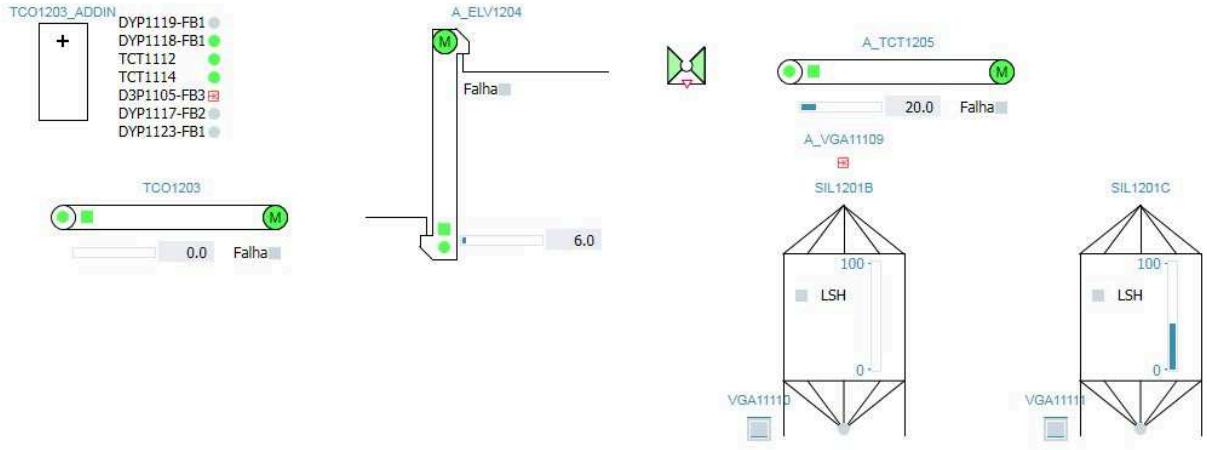


Figura 88 – Simulação da área Envia Preparação. Fonte: Original.

Na Figura 88, o silo SIL1201C, está sendo carregado pela esteira TCT1205. Por sua vez, todas a carga que passou por essa área e foi parar dentro do silo, veio de outras áreas da planta. As áreas que englobam o *script* do setor envia preparação são: Moegas e Transporte, Figura 89 e 90 respectivamente.

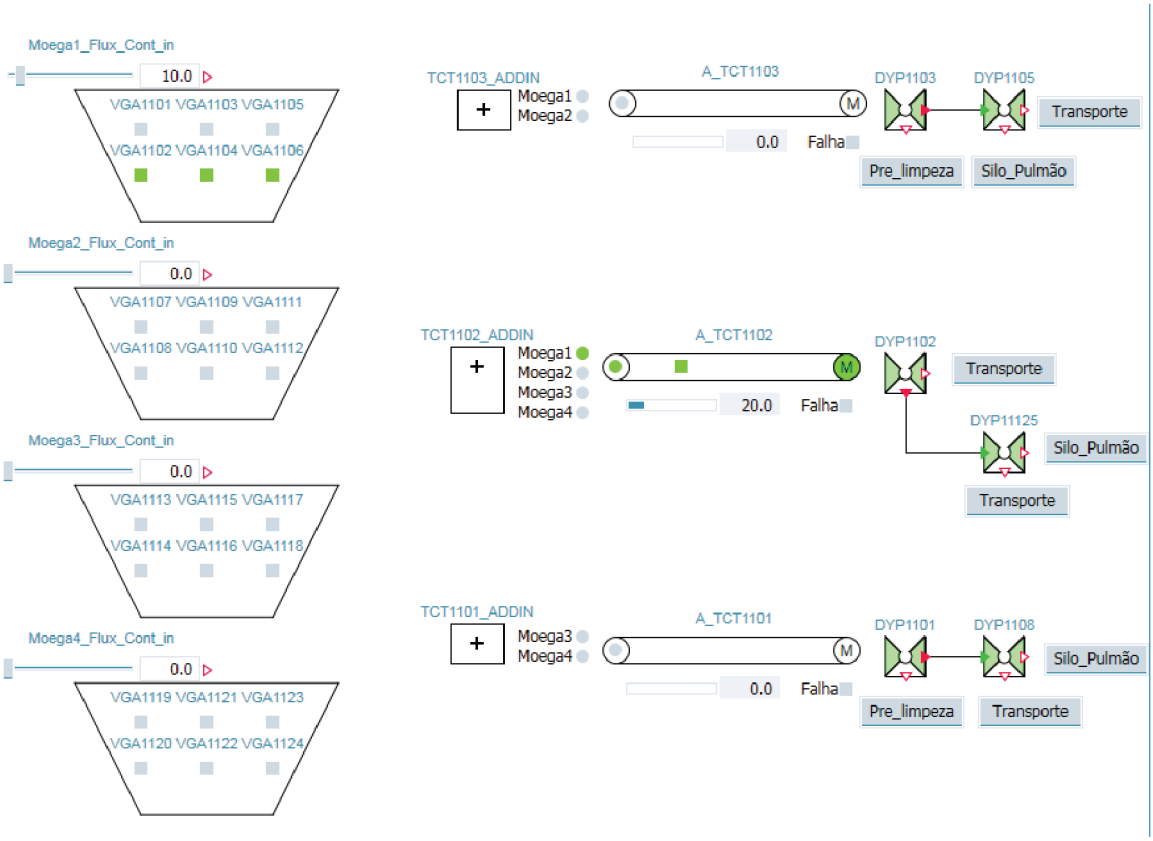


Figura 89 – Simulação da área Moegas. Fonte: Original.

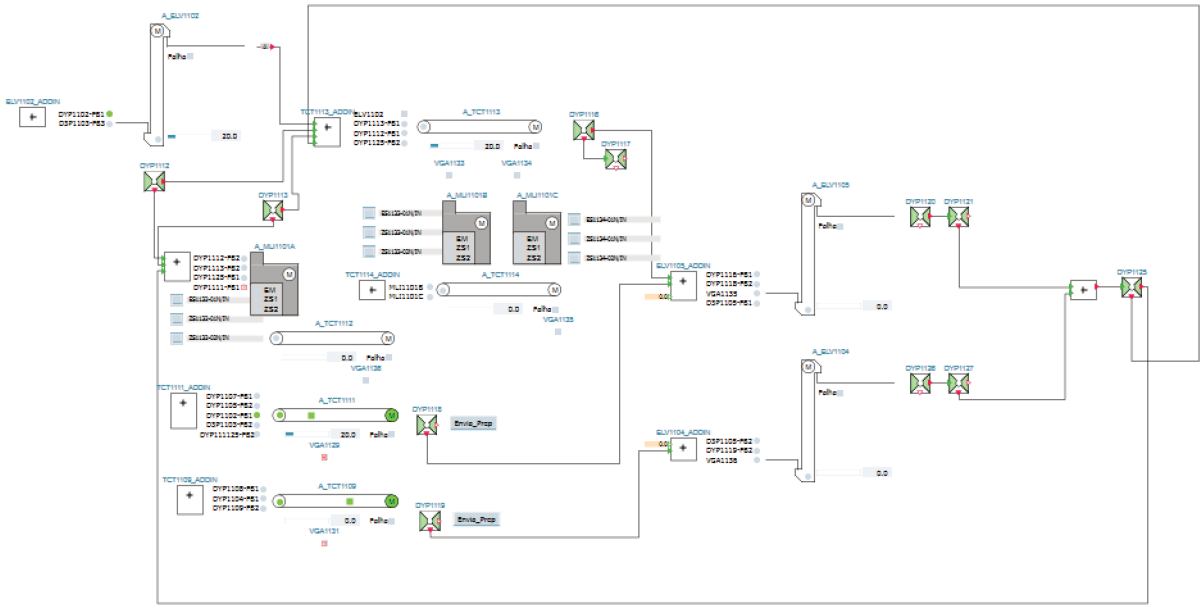


Figura 90 – Simulação da área Transporte. Fonte: Original.

Nota-se, nas Figuras 88, 89 e 90 que existem blocos somadores, `_ADDIN`, esses blocos tem a função de somar todos as possíveis entradas dos equipamentos, e por meio das ligações implícitas (ligações que não são desenhadas, não aparecem graficamente) entre as

simulações, aplicar esse resultado na entrada de fluxo da esteira em questão. Essa soma fica nessa tela, pois ajuda a indicar de onde está vindo o fluxo do processo. Além disso foram colocados botões para ajudar na navegação de uma parte da simulação para outra.

4.7 Testes

Nesta seção serão mostrados alguns testes que juntam a simulação do SIMIT com o software programado no PCS7. O primeiro teste será feito para descobrir se a simulação em nível de sinal está funcionando corretamente. No segundo teste será mostrado o comportamento da simulação e do sistema de controle da Chave de Rotação (nível de componente). O terceiro e o quarto teste mostrarão o funcionamento dos dois tipos de intertravamentos (*interlock* e *protect*) (simulação de nível de equipamento). Por fim o quinto teste mostrará o uso do SFC para ligar uma das rotas da área Envia Preparação para encher um dos silos, em seguida será acionado manualmente o sinal de nível cheio do silo afim de testar o funcionamento do desligamento em cascata (simulação de nível de processo).

4.7.1 Teste da Comunicação (Nível de Sinal)

Primeiramente, na Figura 91 que demonstra a conexão entre os dois softwares PCS7 e SIMIT.

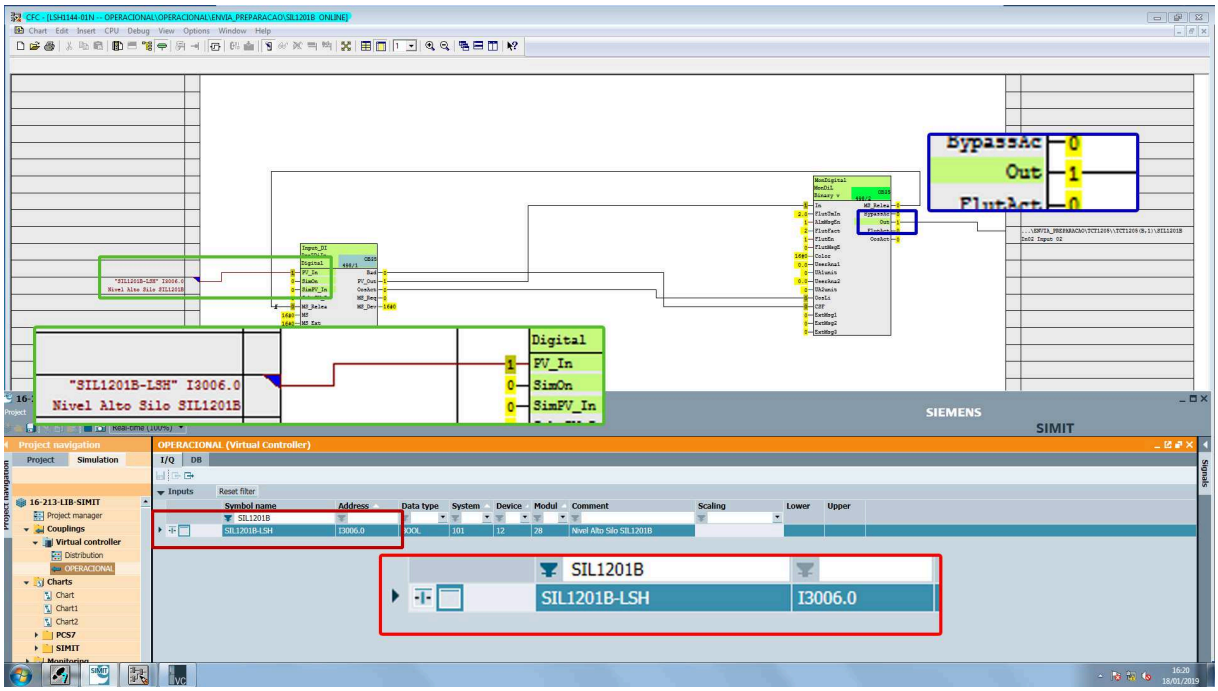


Figura 91 – Simulação Nível de sinal. Fonte: Original.

Na Figura 91, vemos que o PCS7 está recebendo a informação da chave de nível do silo SIL1201B do SIMIT, visto que o sinal aplicado na simulação (vermelho) está chegando

na entrada da lógica do sensor (verde), que por sua vez está replicando o sinal para ser usado nas outras lógicas da programação (azul). Essa simulação é a simulação de nível de sinal, onde os sinais são mudados manualmente.

4.7.2 Teste da Chave de Rotação (Nível de Componente)

No segundo teste, foi feito a simulação no nível de componente, onde foi demonstrado o resultado do CMT e da simulação do `DigitalMon_L_DI_Pulso`, na Figura 92 mostra-se o gráfico dos sinais relevantes ao sensor.

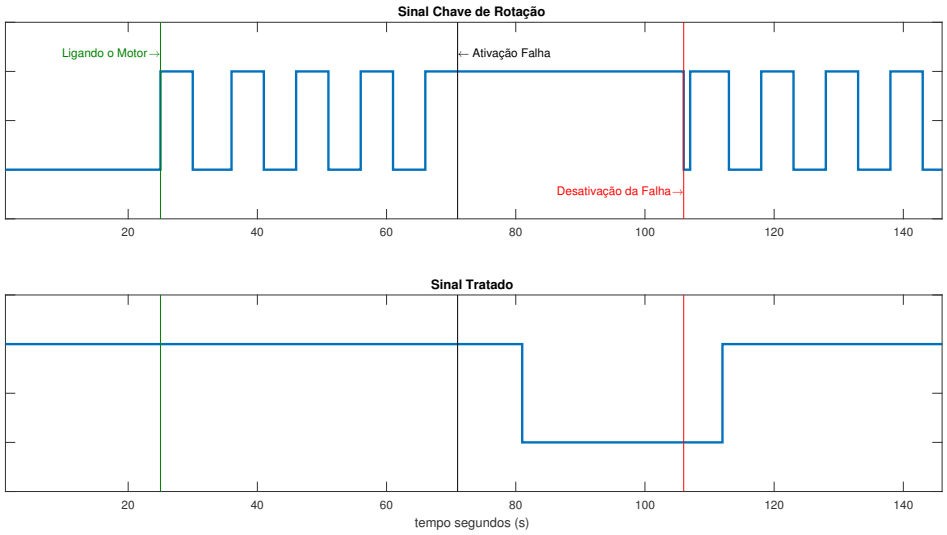


Figura 92 – Sinais da simulação da chave de rotação. Fonte: Original.

Na simulação da chave de rotação, Figura 92, foi iniciado o motor manualmente, pela Simulação da planta e foi observados os sinais dentro do software de controle. Na Figura 92 mostra-se em verde o instante em que o motor é ligado. Após ligar o motor a entrada do sensor fica oscilando e a saída do sinal tratado mantém-se em alto, informando que esta, ok. Nota-se que antes do motor ligar o sensor indica o estado ok, já que com o motor desligado é esperado que o sensor não oscile.

Durante a simulação, foi simulado um erro do sensor, a adição do erro foi feito com um bloco *OR* na saída da simulação do sensor(Figura 93. No caso, o sensor pararia de oscilar enquanto o motor estive-se ligado, e por consequência a saída "tratada" do sensor acusou o erro. A mesma ideia ocorreu ao voltar o sensor para a operação normal, ou seja, depois de uma oscilação o sinal tratado já acusou o sensor em funcionamento correto. Os tempos dos *timers* tanto da simulação quanto do software deverão ser ajustados após testes com a planta real.

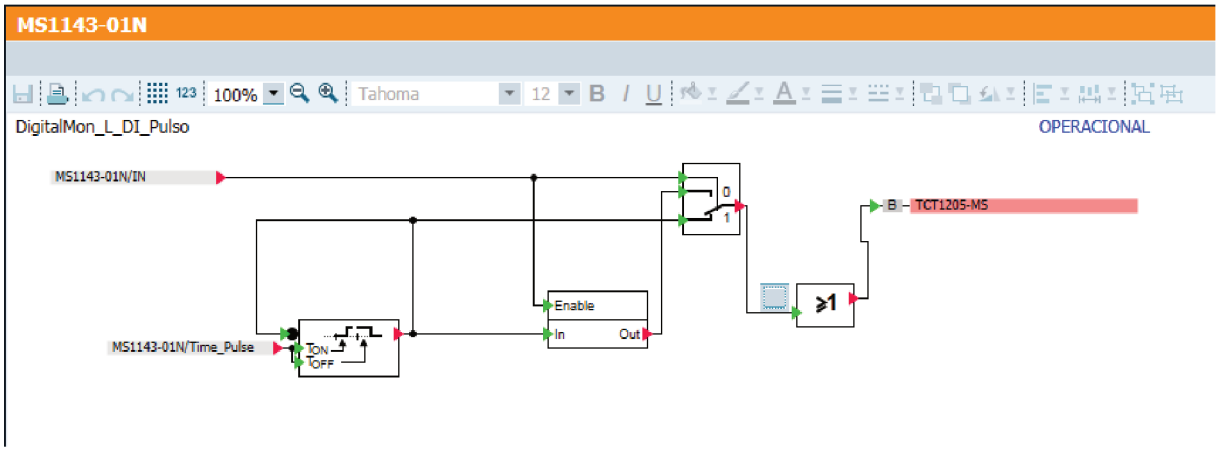


Figura 93 – Modificação da simulação `DigitalMon_L_DI_Pulso` para adicionar erro. Fonte: Original.

4.7.3 Teste dos Intertravamentos (Nível de Equipamento)

Para os testes dos intertravamentos, vale lembrar que esses blocos tem o sinal lógico 1 em correto funcionamento e em sinal lógico 0 caso esteja em falha. Assim, no terceiro teste mostra-se os sinais do blocos relacionados ao bloco *protects*.

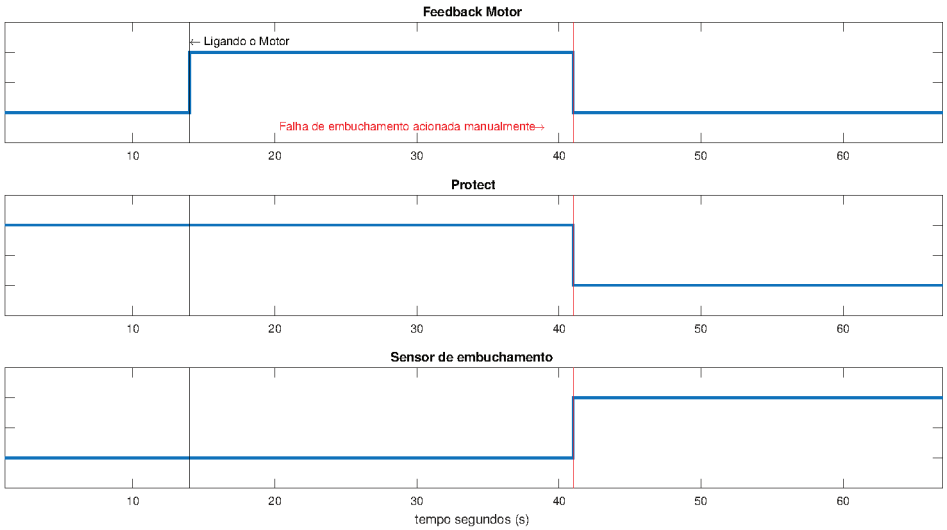


Figura 94 – Sinais da simulação do bloco de *protect*. Fonte: Original.

Na Figura 94, mostra-se que o sinal de ligação do motor ocorreu entre o instante de tempo 10 e 20 s, nesse intervalo nota-se que, tanto para o sinal do sensor quanto para o do intertravamento do tipo *protect*, mantiveram os seus valores. Quando o sensor é manualmente mudado para acusa a falha o motor é automaticamente desligado e o sinal do *protect* varia para 0. Assim, o motor só pode ser ligado novamente após o sensor parar de acusar a falha e o motor ser reiniciado, essa função de *reset* pode ser comandada pelo sistema supervisório, ou pelo bloco. Na Figura 95, nota-se que o bloco está em `RdyToRes`,

que significa pronto para reiniciar.

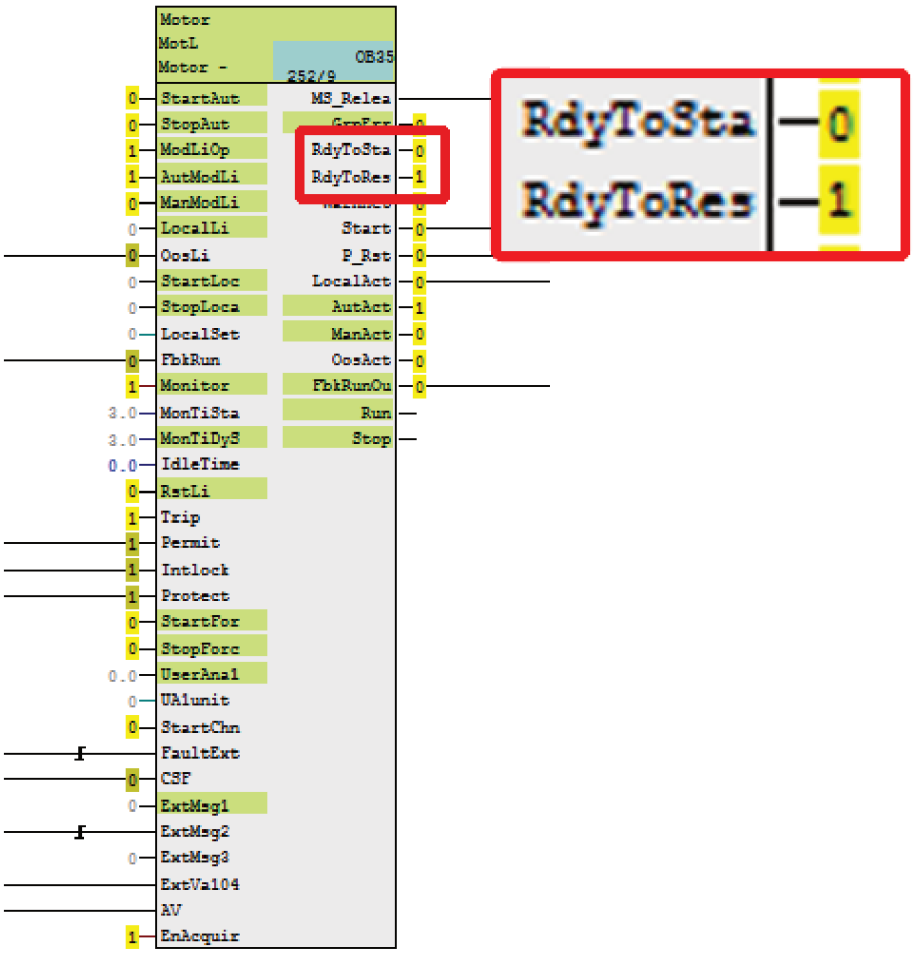


Figura 95 – Bloco do motor após o erro. Fonte: Original.

Quarto teste (Figura 96) mostra a ação do *interlock*. Após o motor da esteira ser ligado, é forçado o sinal do silo para acusar que o silo encheu. Após isso pode-se notar a ação do *interlock* desligando o motor da esteira transportadora.

4.7.4 Rotas e seus Intertravamentos (Nível de Processo)

O quinto teste foi feito para testar os SFC's. No caso foi testado o SFC da rota 3 da Figura 62. Nas Figuras 97, 98 e 99 temos o passo a passo do SFC sendo executado e como isso reflete na simulação da área especifica.

O Primeiro passo depois do início do SFC é colocar os equipamentos em controle automático, e o último é tira-los desse modo. Sendo Assim indo direto para o segundo estado do SFC, nele a válula VGA11109 é fechada para que o silo SIL1201C possa ser enchido. Lembrando que o diagrama de processo dessa área pode ser observado na Figura 43.

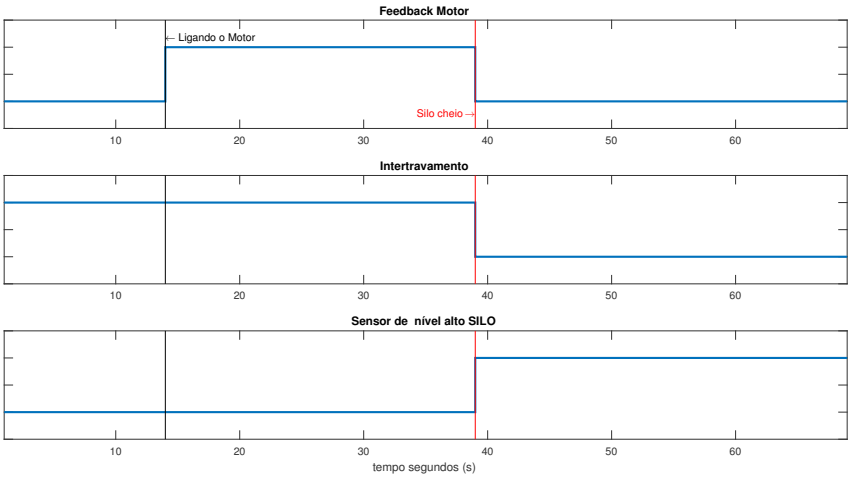


Figura 96 – Sinais da simulação do bloco de *interlock*. Fonte: Original.

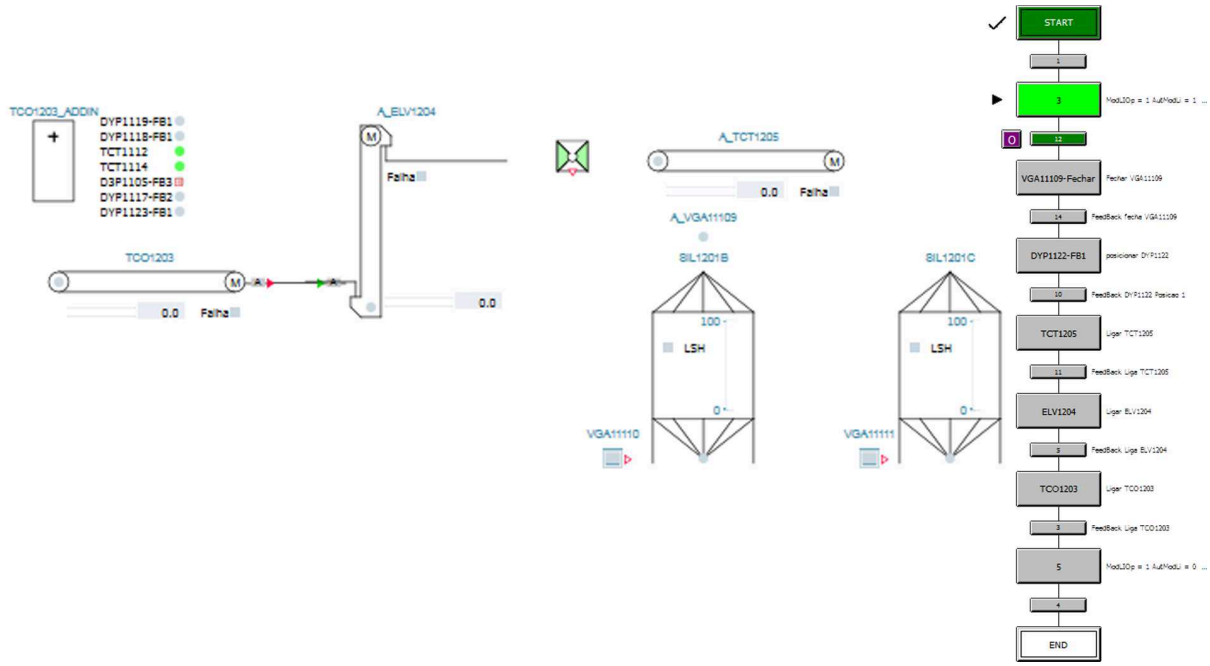


Figura 97 – SFC passo 2. Fonte: Original.

Em seguida a válvula DYP1122 é colocada na posição 1, dessa forma direcionando o fluxo de produto para a esteira TCT1205.

Com todas as válvulas direcionadas os próximos passos são pra ligar os motores das esteiras e do elevador. A sequencia de acionamento dos mesmos pode ser visto na SFC.

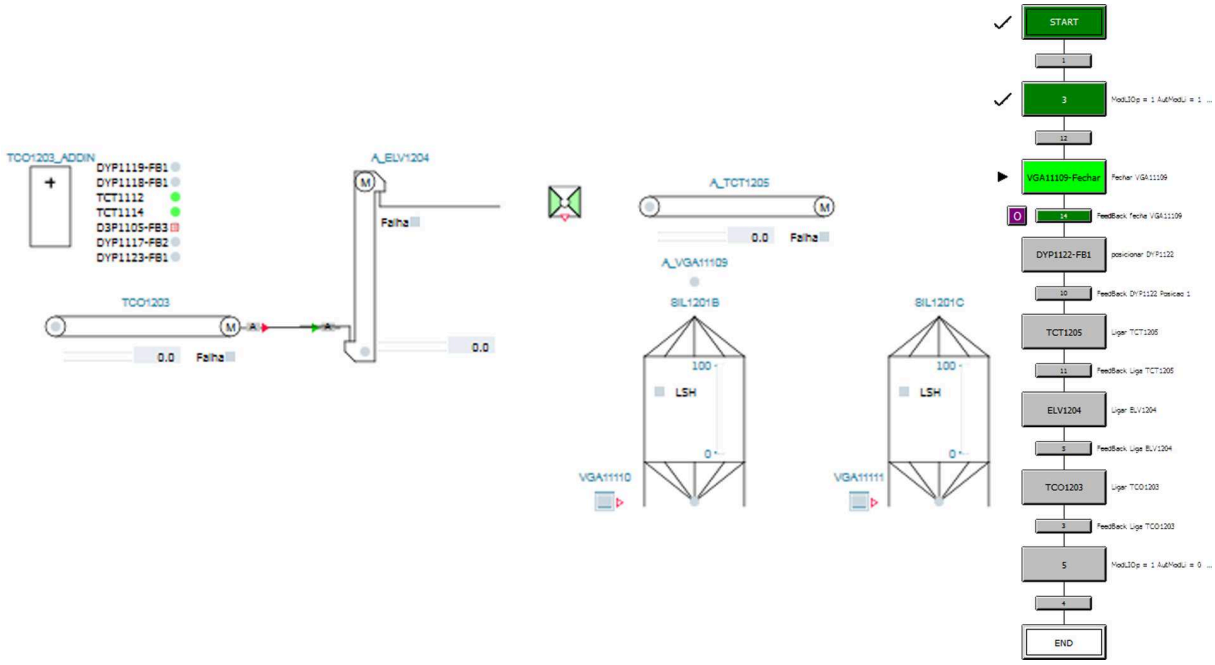


Figura 98 – SFC passo 3. Fonte: Original.

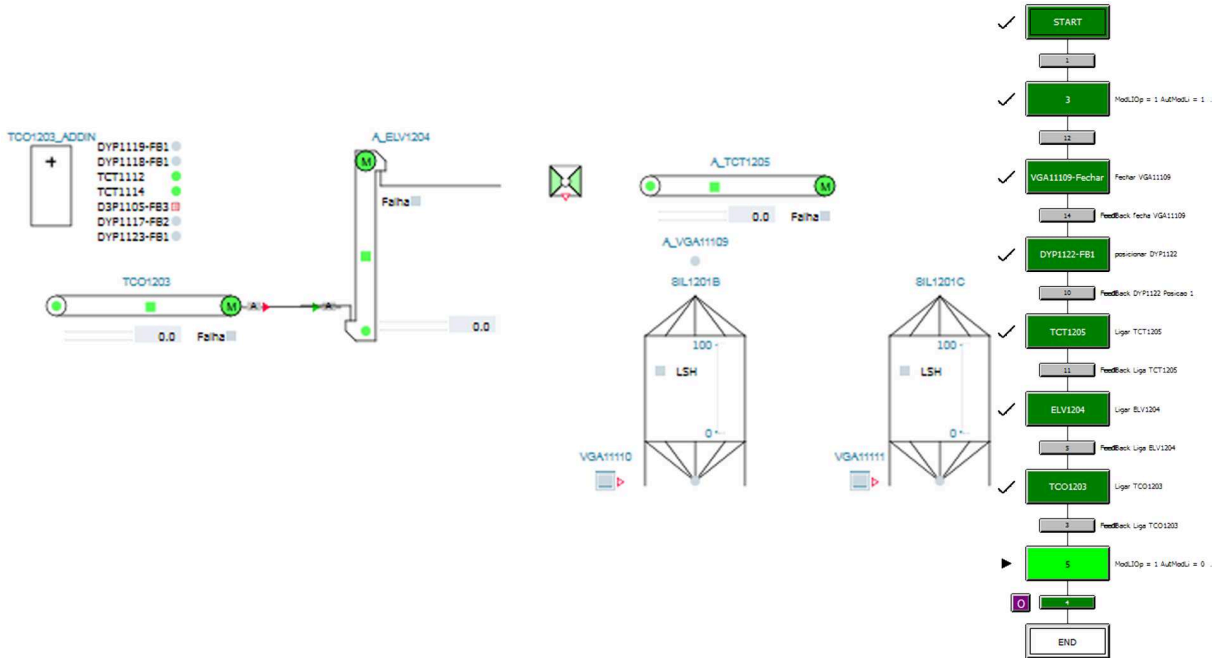


Figura 99 – SFC passo 8. Fonte: Original.

Nota-se que em todos esses passos a simulação indicou, conforme o esperado, o status de equipamento, ligado. Em seguida o sinal de nível máximo do silo foi modificado para testar a sequencia de desligamento do setor. As Figuras 100, 101 e 102 mostram a sequencia de desligamento do *Interlock* de rotas aplicado ao setor todo.

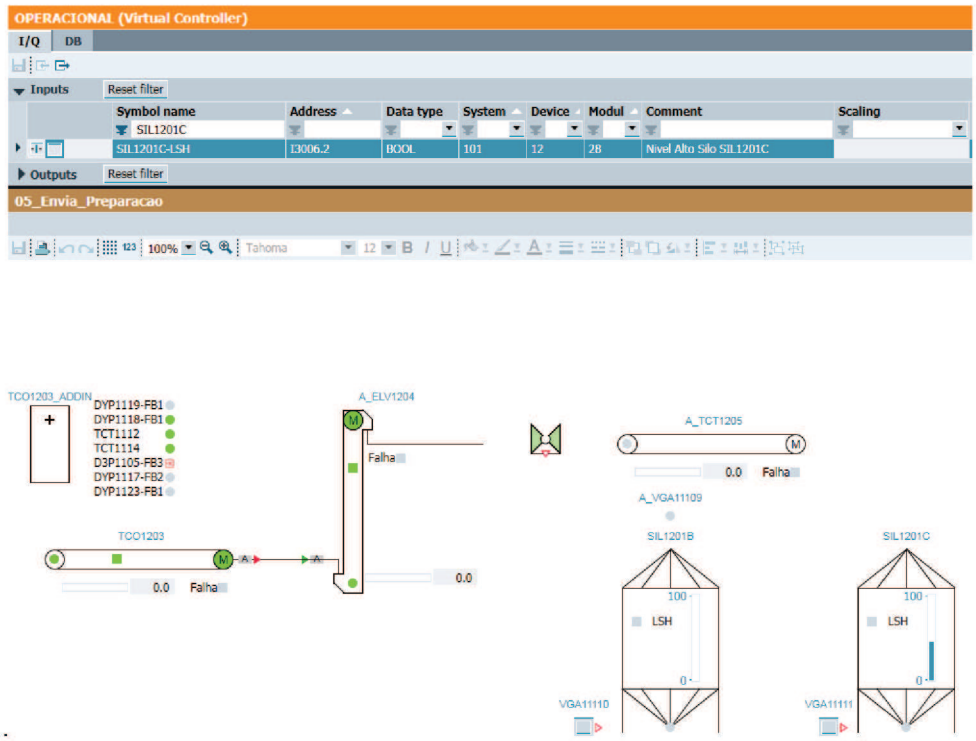


Figura 100 – Sequência de desligamento do *Interlock* Passo 1. Fonte: Original.

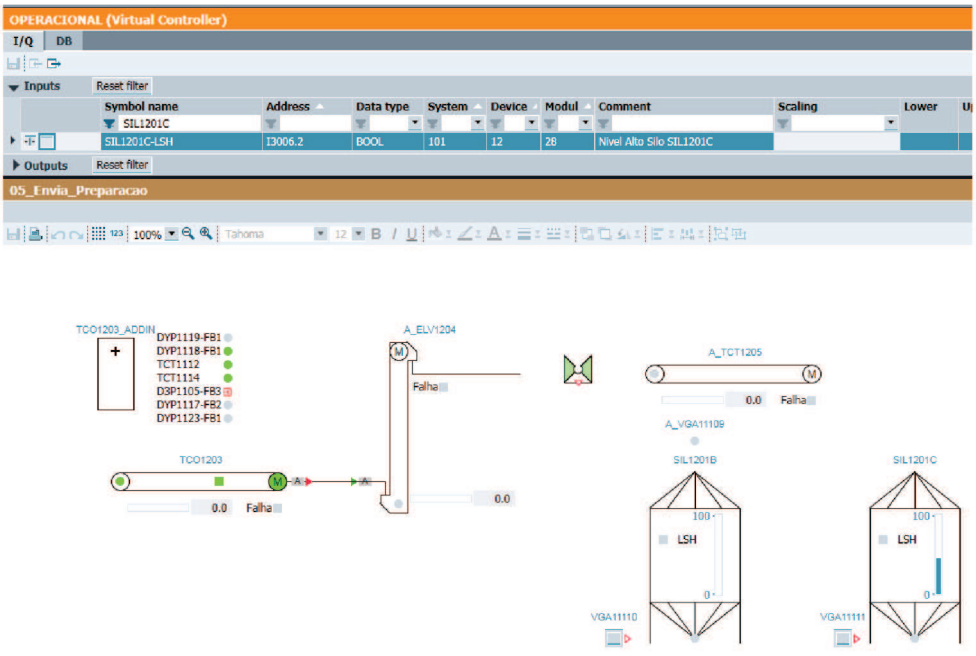


Figura 101 – Sequência de desligamento do *Interlock* Passo 2. Fonte: Original.

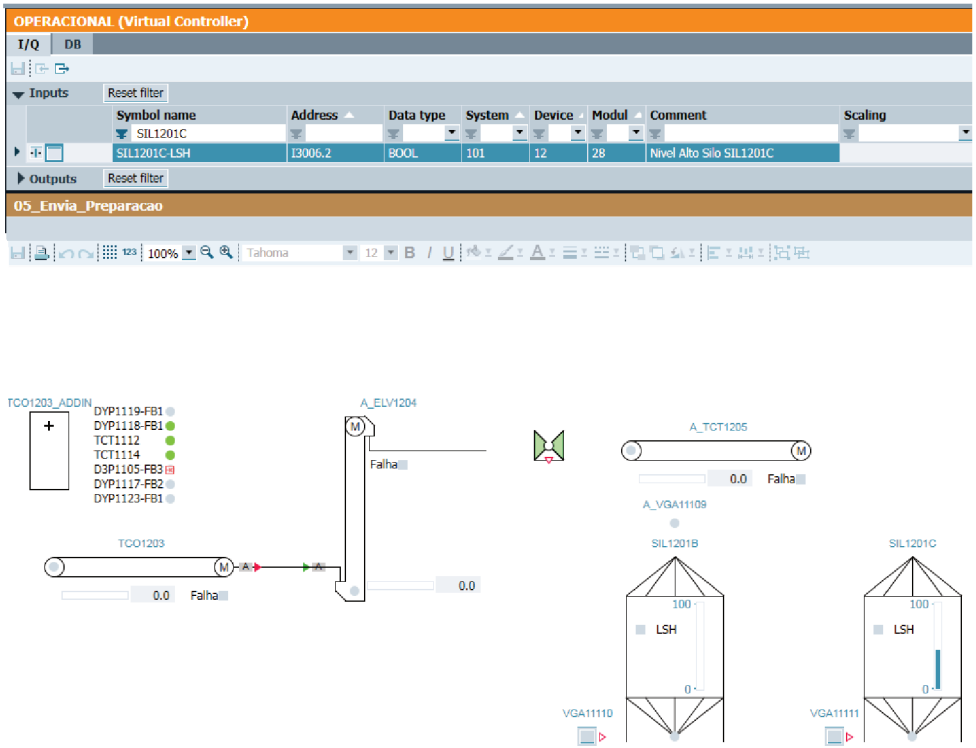


Figura 102 – Sequência de desligamento do *Interlock* Passo 3. Fonte: Original.

Sendo assim a sequência atende a especificação de só permitir ligar um equipamento se, o equipamento frontal estiver ligado, e caso algum desligue deve-se desligar os equipamentos em anteriores em sequência.

4.8 Conclusões

Neste capítulo foi abordado os principais resultados do projeto, dentre eles os dois mais relevantes são os referentes a construção da simulação, e o da programação do sistema de controle. Já os testes realizados juntam esses dois resultados mostrando que em teoria o sistema de controle está funcionado corretamente, pelo menos para a sub-área da Envia Preparação. Já os resultados referentes a revisão do P&ID, modificação da biblioteca do CMT's, construção da arquitetura de rede no PCS7 e atualização e criação das telas do supervisório são relevantes já que, esses, servem de base para atingir os resultados do sistema de controle e da simulação.

5 Conclusões

Nesse trabalho foi apresentado a simulação e a programação do sistema de controle do setor operacional de uma nova unidade industrial de extração de óleo de soja. O primeiro ponto que merece destaque da parte da programação foi a arquitetura de rede, já que no processo de declaração de *hardware* da parte interna da rede AS-I, foi necessário fazer algumas correções, pois notou-se um erro de projeto, no que diz respeito ao número de endereços ocupados pelos módulos de entrada e saída da rede AS-I. Essa correção, não se limitou a apenas a área do operacional e foi aplicada em todas as outras áreas da indústria.

Já com relação a revisão das telas do sistema supervisor e dos diagramas P&ID's, tiveram grande utilidade, primeiramente por que, esses de colaboraram para a programação tanto do sistema de controle quando da simulação. Além disso, as telas foram bem aceitas pelo cliente já que não foi necessário fazer grandes alterações após o retorno do cliente. A revisão do diagrama P&ID serviu, também, para levantar dúvidas referentes de funcionamento dos equipamentos, que posteriormente foram enviadas e respondidas pelas empresas fornecedoras do projeto, isso fora o fato de manter a plataforma COMOS atualizada.

A criação do CMT para a biblioteca da empresa foi relevante já que ajudou, novamente, a programação do sistema de controle, já que tornou automatizado o processo de criação dos CFC's dos sensores do tipo chave de rotação. A mesma modificação foi feita na biblioteca do *software* de simulação, agilizando a programação da simulação. Observando o teste feito com a simulação e a lógica de controle do mesmo, pode-se afirmar que, nesse aspecto, os resultados foram satisfatórios, já que os dois se comportaram como o previsto.

Sobre a programação, no âmbito geral do projeto, tanto no sistema de controle quanto na simulação, afirma-se que os resultados foram satisfatórios, uma vez que, os dois reagiram conforme o esperado. Especificamente dentro da programação das lógicas de controle da planta foi possível atender as especificações, autorizando o operador a somente acionar a planta dentro da sequência específica, e por consequência, o desligamento em caso de falha. Com relação ao requisito da mudança de rotas, ainda não foi implementado nenhuma solução. Outro ponto, que mostra, que o software de controle ainda está incompleto, é que os acionamentos sequencias dos motores dos equipamentos de limpeza, pré-limpeza e trilhagem ainda não foram feitos, por falta das respectivas documentações dos fabricantes.

Sobre a simulação, além dos resultados dos testes, tem-se como resultado, todos os elementos criados para a simulação de transporte, armazenagem, secagem e limpeza. Esses poderão ser usados para fazer a simulação de outras áreas da planta que tenham, principalmente, movimentação de grãos, como é o caso das áreas da preparação e expedição.

Como, principal trabalho futuro está juntar os níveis de simulação de Equipamento e Processo, afim de simplificar a complexidade da simulação e da programação da mesma. E assim, possivelmente, usar o recurso de importação do COMOS → SIMIT, que montaria as simulações automaticamente. Com relação a isso já foi feito o teste do equipamento esteira transportadora, onde foi montado um componente com dinâmica e animação juntas, além de usar um sinal, tipo *struct*, para mandar o fluxo de soja simulada apenas para os componentes da mesma biblioteca. Dessa forma, deve-se implementar a mesma estratégia para os outros elementos como elevador, silo, secador e as maquinas de limpeza e pré-limpeza. Bem como aplicar um ajuste fino nos parâmetro da simulação, como o tempo dos *timers*, ajustes nos tipos de sensores (normalmente aberto ou normalmente fechado) e acrescentar os comandos de aeração dos silos.

Em relação a programação do controle da planta é necessário, dar continuação ao trabalho, após o recebimento das documentações dos equipamentos que ainda não foram programados. Outro trabalho futuro é a importação das lógicas de controle para dentro do COMOS, afim de ter uma documentação inteligente do *software* de controle, expandindo ainda mais o poder de indexamento de documentação do COMOS. Ainda como trabalhos futuros, poderia ser feito um teste com a simulação, o sistema de controle e o sistema supervisor, a fim de testar a operação da planta pelas telas desenvolvidas.

Finalmente, no geral, o projeto foi bem sucedido em cumprir a maioria de seus objetivos propostos já que foi feita a revisão da documentação, das telas e de arquitetura de rede, bem com foi feito a programação do sistema de controle, dentro do possível com as documentações apresentadas. E, por fim, foi montada a primeira versão da simulação, que serviu em seu propósito de auxiliar na programação. Porém ainda há espaço para trabalho em todas essas etapas, principalmente, na melhoria da simulação e conclusão da programação do *software* de controle. Além disso, só sera possível finalizar o código de controle após o *startup* da planta. Bem como, só poderá ser garantido a eficiência das telas, após testes e treinamentos com os futuros operadores da planta. Lembrando que esses treinamentos poderão ser feitos com a versão final da simulação da planta, que será revisada após o inicio das operações da planta.

Referências Bibliográficas

1 PRODUTOS. [S.l.]: Bianchini. <http://bianchinisa.com.br/produtos/#>. [Acesso em 09 de dezembro de 2018].

2 SILVA, L. C. da. Estruturas para armazenagem de grãos a granel. *Boletim Técnico: AG*, f. 43, 02 2010. [Departamento de Engenharia de Alimentos].

3 Carpan | Produtos. [S.l.]: Carpan. <http://www.carpan.com.br/site/produtos/>. [Acesso em 13 de Janeiro de 2019].

4 FERNANDEZ, J. O. S. Processo de extração e refino de Oleo de soja. *Greylogix Brasil*, f. 43, 10 2018. [Revisão: 1.0].

5 Kepler Weber. [S.l.]: Kepler Weber. <http://www.kepler.com.br/>. [Acesso em 04 de dezembro de 2018].

6 Silo Pressure Measurements - Background. [S.l.]: silopressuremeasurements. <https://silopressuremeasurements.weebly.com/background.html>. [Acesso em 13 de Janeiro de 2019].

7 Mechanical Engineering: Bucket Elevator. [S.l.]: mechanic-info. <http://mechanic-info.blogspot.com/2011/04/bucket-elevator.html>. [Acesso em 24 de Janeiro de 2019].

8 PRODUTOS. [S.l.]: Agriaço. <http://www.agriaco.com.br/produtos/tipos/4>. [Acesso em 13 de Janeiro de 2019].

9 ANSI. ISA 5.1:2009, Instrumentation Symbols and Identification. 2009.

10 KIRUBASHANKAR, R.; KRISHNAMURTHY, K.; INDRA, J. Remote monitoring system for distributed control of industrial plant process. CSIR, 2009.

11 Industrial Networks - Part 1. [S.l.]: SMAR. <http://www.smar.com/en/technical-article/industrial-networks-part-1>. [Acesso em 09 de dezembro de 2018].

12 Tudo sobre a Tecnologia ASi. [S.l.]: SMAR. <http://www.smar.com/brasil/asi>. [Acesso em 02 de novembro de 2018].

13 AG, S. System manual as-interface / asisafe. *Siemens*, 2008.

14 Virtual Academy of AS-International Association. [S.l.]: AS-I foundation. *Download at* "<https://www.as-interface.net/en/technology/education/>". [Acesso em 28 de dezembro de 2018].

15 Industrial Networks Part 1. [S.l.]: SMAR. <http://www.smar.com/en/technical-article/industrial-networks-part-1>. [Acesso em 20 de Janeiro de 2019].

16 PROFINET Real-Time Communication. [S.l.]: Profibus. <http://www.profibus.org.pl/index.php?option=com_docman&task=doc_view&gid=28>. [Acesso em 20 de Janeiro de 2019].

17 PROFINET for Network Geeks (and Those Who Want to Be). [S.l.]: PROFINET. <https://us.profinet.com/profinet-network-geeks-want/>. [Acesso em 20 de Janeiro de 2019].

18 Structured industrial cabling and connection under PROFINET. [S.l.]: The industrial ethernet book. <https://iebmedia.com/index.php?id=6023&parentid=63&themeid=255&hft=50&showdetail=true&bb=1&PHPSESSID=dehbgsvbj01dviv2vn71mfsbk6>. [Acesso em 20 de Janeiro de 2019].

19 PERIN, R. Implementação de um sistema de controle distribuído em uma linha de produção de margarina. Florianópolis, f. 94, 2018.

20 IEC, I. 61131-3: Programmable controllers–part 3: Programming languages. *International Standard, Second Edition, International Electrotechnical Commission, Geneva*, v. 1, p. 2003, 2003.

21 Praticamente todo o crescimento do país vem do agronegócio, diz Maggi. [S.l.]: Globo Rural| Economia. <https://revistagloborural.globo.com/Noticias/Economia/noticia/2018/03/praticamente-todo-o-crescimento-do-pais-vem-do-agronegocio-diz-maggi.html>. [Acesso em 11 de novembro de 2018].

22 HIRAKURI, M. H.; LAZZAROTTO, J. J. O agronegócio da soja nos contextos mundial e brasileiro. *Londrina: Embrapa Soja*, p. 9–15, 2014.

23 XSTORY - Bilfinger Greylogix GmbH. [S.l.]: Bilfinger Greylogix Gmbh. <https://greylogix.com/xstory/>. [Acesso em 06 de janeiro de 2019].

24 GLx Brasil. [S.l.]: GreyLogix Brasil. <http://www.greylogix.com.br/>. [Acesso em 11 de novembro de 2018].

25 Nossa História. [S.l.]: COAMO. <http://www.coamo.com.br/site/institucional/>. [Acesso em 11 de Novembro de 2018].

26 O, R. S. . a. Anuário do agronegócio. In: _____. 1. ed. [S.l.]: Editora Globo, 2018. (1, v. 1), cap. 1, p. 80–82.

27 Noticias COAMO - O estágio das obras das industrias da COAMO em Dourados. [S.l.]: COAMO. <http://www.coamo.com.br/site/noticia/1418/o-estagio-das-obras-das-industrias-da-coamo-em-dourados>. [Acesso em 11 de novembro de 2018].

28 SLONGO, C. S. W. e L. H. Modelo de controle para o processo de beneficiamento de sementes baseado em eventos discretos e teoria de controle supervisÓrio. Toledo, f. 71, 2014.

29 PETRUZELLA, F. *Controladores Lógicos Programáveis - 4ed.* [S.l.]: AMGH Editora, 2014. ISBN 9788580552836.

30 MAZIERO, C. A. Sistemas operacionais: Conceitos e mecanismos. *Livro aberto. Acessível em: "<http://wiki.inf.ufpr.br/maziero/lib/exe/fetch.php?media=so:so-livro.pdf>*, 2014.

31 SIEMENS ENERGY AUTOMATION. DCS or PLC? Seven Questions to Help You Select the Best Solution. [S.l.]: Siemens Energy Automation, 2007. <https://w3.siemens.com/mcms/process-control-systems/SiteCollectionDocuments/efiles/pcs7/support/marktstudien/PLC_or_DCS.pdf>. Alpharetta, GA.

32 HANSSEN, D. H. *Programmable Logic Controllers: A Practical Approach to IEC 61131-3 Using CODESYS*. [S.l.]: John Wiley & Sons, 2015.

33 JOHN, K.; TIEGELKAMP, M. *IEC 61131–3: Programming Industrial Automation Systems: Concepts And Programming Languages, Requirements for Programming Systems, AIDS to Decision-making Tools*. [S.l.]: Springer, 2001. ISBN 9783540677529.

34 TANENBAUM, A. S. a d. wetherall. *Computer networks. 5th ed. Boston: Pearson*, 2011.

35 PIRÂMIDE de automação industrial: O que é? [S.l.]: logiquesistemas. <ttps://www.logiquesistemas.com.br/blog/piramide-de-automacao-industrial/>. [Acesso em 27 de dezembro de 2018].

36 WINKEL, L. Real-time ethernet in iec 61784-2 and iec 61158 series. In: IEEE. *Industrial Informatics, 2006 IEEE International Conference on*. [S.l.], 2006. p. 246–250.

37 Uma visão do Protocolo PROFINET e suas aplicações. [S.l.]: Inatel. <https://www.inatel.br/biblioteca/artigos-cientificos/2008/3813-uma-visao-do-protocolo-industrial-profinet-e-suas-aplicacoes/file>. [Acesso em 20 de Janeiro de 2019].

38 LUGLI, A. B.; SANTOS, M. M. D. Redes industriais para automação industrial: As-i, profibus e profinet. *São Paulo: Érica*, 2010.