U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Controlo Distribuído e Heterogéneo de Linhas de Produção Industriais

José Jácome Felgueiras
Aluno nº 201207161

Dissertação realizada no âmbito do
Mestrado Integrado em Engenharia Mecânica
Ramo de Automação

Orientador: Prof. António J. Pessoa de Magalhães
Professor Auxiliar do Departamento de Engenharia Mecânica da
Faculdade de Engenharia da Universidade do Porto

Junho de 2017

Resumo

A notável evolução das áreas de automação e comunicação industrial observada desde meados do século passado teve como consequência o aparecimento de inúmeras soluções de controlo lógico, protocolos de comunicação e suportes físicos de redes industriais, originadas em diferentes eras e desenvolvidas por inúmeros fabricantes. Devido à sua diversidade e por motivos de sigilo das organizações que as criaram, estas tecnologias nem sempre são compatíveis entre si, o que resulta num obstáculo à sua integração em ambientes industriais.

Numa tentativa de abordar esta questão, a presente dissertação descreve a realização de sistemas de controlo distribuído e cooperativo de ambientes virtuais automatizados. O objetivo deste projeto consiste na elaboração de soluções de controlo lógico distribuído, baseadas na coordenação de PLCs de diversos fabricantes e gerações, com recurso a vários protocolos e suportes físicos de redes de comunicação industrial. Aquando disto, pretendeu-se acostumar o autor nesta mesma prática.

As soluções de controlo e de comunicação utilizadas partiram desde PLCs e protocolos atuais, para controladores mais antigos e protocolos menos sofisticados, o que resultou numa crescente dificuldade de implementação destas soluções. Os alvos do controlo implementado consistiram em ambientes industriais virtuais criados com o *software Factory I/O* da empresa portuguesa *Real Games Lda*. A apresentação e especificação dos casos de estudo concebidos são complementadas pela utilização de diagramas baseados nas normas UML e *SysML* e de redes de Petri.

A dissertação inicia com uma explicação mais detalhada do problema abordado, as suas implicações e alguma contextualização histórica que está nas suas origens. Daí parte para um levantamento dos recursos tecnológicos utilizados no trabalho desenvolvido. Depois, são apresentadas as normas de modelação e especificação de sistemas destinadas à representação dos casos de estudo elaborados. Reunidos estes conceitos, são reveladas as soluções de controlo desenvolvidas, constituídas pelos cenários virtuais, controladores lógicos, redes de comunicação e os algoritmos de controlo, sendo cada um destes aspetos ilustrados com modelos adequados.

Considera -se que cada caso de estudo foi executado com sucesso, tendo-se observado o comportamento pretendido para cada cenário, assim como um correto desempenho a nível das comunicações.

Palavras-Chave: Treino de competências em automação industrial, Controlo lógico distribuído, Controladores lógicos programáveis, Comunicações e redes industriais, Modelação de sistemas de eventos discretos, Modelação de sistemas controlados, Cenários Industriais Virtuais, Simulação de Ambientes Fabris.

Abstract

The perceptible evolution in the fields of automation and industrial communication, which has been observed since mid-20th Century, resulted in the emergence of numerous logical control solutions, communication protocols and physical media of industrial networks, originated in different eras and developed by numerous manufacturers. Due to its diversity and for reasons of sigil of the organizations which created them, these technologies are not always compatible between each other, resulting in an obstacle to their integration in industrial environments.

In an attempt to approach this issue, the present dissertation describes the concretization of distributed and cooperative control systems of virtual automated environments. The objective of this project consists of the elaboration of distributed logical control solutions, based on the coordination of PLCs from different manufacturers and technological periods, while resorting to several protocol and physical media of industrial communication networks. In the meanwhile, it was intended that its author would be become accustomed to this practice.

The used control and communication solutions ranged from present day protocols and PLCs, to more outdated controllers and less sophisticated protocols, resulting in a growing difficulty in implementing these solutions. The targets of the implemented control consisted of virtual industrial environments created using the *Factory I/O* software from Portuguese company *Real Games Lda*. The presentation and specification of the conceived study cases are complemented by the utilization of diagrams based on the UML and *SysML* norms and of Petri Nets.

The dissertation begins with a more detailed explanation of the approached issue, its implications and some historical contextualization which lies in its origins. From there, it moves to a summary of the technological resources used in the developed work. Afterwards, the systems modelling and specification norms, directed at the representation of the created study cases, are presented. With these concepts gathered, the developed control solutions, consisting of the virtual scenarios, logical controllers, communication networks and control algorithms, are displayed, with each one of these concepts being illustrated with the adequate models.

It is considered that each study case was successfully executed, having observed the desired behavior for each scenario, as well as a correct performance at the communications level.

Keywords: Proficiencies training on industrial automation, Distributed logical control, Programmable logical controllers, Communications and industrial networks, Discrete Event Systems Modelling, Virtual industrial Scenarios, Simulation of production environments.

Agradecimentos

Ofereço os meus honestos agradecimentos ao Professor António Pessoa de Magalhães pela proposta dum tema que estimulou em mim um grande interesse, assim como pelo apoio e disponibilidade que demonstrou ao longo deste trabalho.

Também agradeço ao Engenheiro Bruno Teófilo Vigário, gestor da *Real Games Lda.*, por prestar auxílio em questões relacionadas com o *Factory I/O* e outros *software*, e por ser um exemplo profissional.

Expresso a minha gratidão aos vários colegas e professores que me acompanharam ao longo do curso, aos meus amigos por me apoiarem e especialmente à minha família por terem feito de mim a pessoa que sou hoje.

Índice

Índice de Ilustrações

Índice de Tabelas

Abreviaturas, Acrónimos e Siglas

- CAN - *Controller Area Network*
- CPU – *Central Processing Unit*
- EIA - *Electronic Industry Association*
- DES - *Discrete Event Systems*
- FBD – *Function Block Diagram*
- FIFO – *First In First Out*
- HMI - *Human Machine Interface*
- I/O – *Inputs/Outputs*
- IEC - *International Electrotechnical Commission*
- IEEE - *Institute of Electrical and Electronics Engineers*
- IL – *Instruction List*
- ISO - *International Organization for Standardization*
- LD – *Ladder Diagram*
- MMI - *Man Machine Interface*
- MPI - *Multi Point Interface*
- OLE - *Object Linking and Embedding*
- OMG - *Object Management Group*
- OPC - OLE *for Process Control*
- P&P – *Pick and Place*
- PC – *Personal Computer*
- PLC – *Programmable Logical Controller*
- PPI - *Point to Point Interface*
- PtP - *Point-to-Point*
- RS(232/422/485) – *Recommended Standard*
- SCADA - *Supervisory Control And Data Acquisition*
- SFC – *Sequential Function Chart*
- *SysML - Systems Modelling Language*
- TIA - *Telecommunications Industry Association*
- TCP/IP - *Transmission Control Protocol/Internet Protocol*
- UDP - *User Datagram Protocol*
- UML - *Unified Modelling Language*

Capítulo 1 - Introdução

Conquanto a imparável evolução da tecnologia e da engenharia se reflita numa constante introdução de produtos e sistemas novos e tecnologicamente relevantes, nem sempre é possível, nem tão pouco faz sentido, adotar tais propostas, por mais tentadoras e promissoras que posam parecer.

Um cenário em que esta situação se verifica com especial relevância, são os equipamentos e máquinas associados aos sistemas flexíveis de produção. Neste contexto, as propostas de inovações tecnológicas são constantes e aliciantes, com novos equipamentos produtivos controlados por sofisticados automatismos lançados quase diariamente no mercado por múltiplos fabricantes. Contudo, são em geral equipamentos particularmente dispendiosos que envolvem compromissos financeiros por parte dos empresários industriais, que, por esse motivo e pela necessidade de amortizar investimentos anteriores, não estão muitas vezes em condições de as adotar.

Este facto é sem dúvida um fator que contribui para a característica mescla de tecnologias e soluções produtivas observáveis em instalações industriais de alguma dimensão e idade. Efetivamente, após um período de grande investimento e modernização – feito eventualmente aquando da entrada em funcionamento da fábrica ou por altura da introdução de uma nova linha de produção – é vulgar, e mesmo inevitável, moderar os investimentos, limitando-os a "pequenas rubricas" que visam manter a atualidade da instalação e, sempre que possível, melhorar a sua produtividade, dentro das contingências impostas pelo investimento financeiro e tecnológico inicial.

A heterogeneidade tecnológica tão típica da generalidade das instalações fabris coloca problemas diversos, mas sem dúvida, curiosos e aliciantes, aos engenheiros que as exploram. Concretamente, no caso dos sistemas flexíveis de fabricação, não é raro os engenheiros de automação serem confrontados com a necessidade de alterar um programa de PLC desenvolvido num *software* concebido para um sistema operativo já em desuso – e, não raras vezes, a partir de um ficheiro mantido num suporte obsoleto. Por outro lado, a integração de pequenos equipamentos de última geração – como sensores mais sofisticados, ou sistemas de identificação – requerem, em geral, uma familiarização com novas tecnologias e terminologias. Por fim, e apenas para que a lista não se estenda indefinidamente, a adoção de programas de manutenção mais modernos e eficazes é, não raramente, um objetivo que obriga os engenheiros a estudar, dominar e implementar uma série de conceitos na área das tecnologias da informação que, até aí, só conheciam na perspetiva de utilizadores.

A presente dissertação emana precisamente da necessidade, ou pelo menos, do grande interesse, em treinar minimamente um estudante de engenharia de automação para enfrentar a inventável heterogeneidade dos ambientes fabris. A proposta lançada é um trabalho de desenvolvimento de soluções de controlo de máquinas e equipamentos industriais que corporizam exigências e topologias frequentemente encontradas na indústria. A particularidade, inovação, rigor e utilidade do desenvolvimento tem múltiplas vertentes:

Em primeiro lugar, a utilização de um *software* de emulação de ambientes industriais altamente realista, flexível e abrangente que, por um lado, estimula a criatividade e, por outro, possibilita um desenvolvimento quase ilimitado dos cenários industriais mais típicos e interessantes para alicerçar esta dissertação.

Em segundo, a utilização de um leque alargado de PLCs e HMIs de diferentes gerações e fabricantes, todos com a particularidade de serem já ou ainda extremamente populares, uma vez que se trata de controladores surgidos nas últimas décadas.

Em terceiro, o recurso a redes de comunicação diversificadas, todas elas populares e maioritariamente abertas, fazendo assim uma "demonstração prática" daquilo que tem sido o estado da arte das redes de comunicação industrial nos últimos tempos, salientando a transição das redes série para as TCP/IP.

Em quarto lugar, a adoção de técnicas de modelação de sistemas de eventos discretos, como redes de Petri e *SysML*, na definição dos problemas a resolver. Usando estas técnicas é possível encontrar padrões de problemas e provar a eficiência das respetivas soluções que se tornam desse modo adaptáveis a cenários duais dos aqui abordados.

Por último, a utilização de técnicas estruturadas de programação que conduzem à geração de código que implementa de forma fidedigna os modelos atrás citados, tornando assim natural e compreensível a transposição dos requisitos funcionais para o código de controlo.

Face ao exposto, é lícito dizer-se o trabalho apresentado tem pois propósitos ambiciosos, assenta num desenvolvimento metódico e sólido, incorpora tecnologias e metodologias atuais e relevantes, e propõe-se ser uma importante plataforma de aprendizagem para o seu autor.

1.1. Objetivos

O objetivo fundamental do presente trabalho é acostumar o seu autor à prática do desenvolvimento e integração de soluções de controlo lógico baseado em redes de PLCs de diferentes fabricantes e gerações. Os cenários a considerados refletem casos de estudo tipicamente encontrados no mundo industrial e considerados relevantes na literatura especializada.

Nesse sentido, e como primeira meta, importa definir e recriar cenários industriais de interesse, semelhantes aos encontrados em instalações reais de produção flexível, onde equipamentos e máquinas coabitam e interagem de diversas formas. Duas facetas devem ser tidas em conta nesta atividade: por um lado, encontrar um ambiente visual e realista que possa ser "alvo de ações de controlo", imitando com exatidão o efeito das mesmas em equipamentos industrias; por outro, encontrar modelos minimamente formais, com uma base matemática, portanto, que salientem e equacionem os aspetos essenciais do controlo discreto exigido em cada um dos casos tratados. A escolha aqui recaiu no *software Factory I/O*, desenvolvido pela empresa *Real Games*, para resolver a primeira questão, e nas redes de Petri e outras formas de modelação de sistemas de eventos discretos para resolver o segundo.

Definidos os sistemas alvo, o objetivo seguinte é desenvolver e integrar os respetivos controladores. A "prata da casa" e as soluções gratuitas fornecerão a generalidade dos meios necessários. Empregam-se, no primeiro caso, PLCs e consolas adquiridos nos últimos vinte anos e que coincidem com escolhas industriais vulgares - tais como controladores das marcas *Siemens*, *Omron, Schneider*, *Moeller*, etc. No segundo caso, soluções baseadas em *SoftPLCs Codesys* V2.3 e V3.5 são os recursos a destacar.

O vasto leque de controladores considerado potencia também a heterogeneidade das redes de comunicação utilizadas. A definição, configuração e exploração de uma diversidade de redes e formas de comunicação – *Modbus* série e TCP/IP, *Profinet*, *Ethernet/IP*, OPC, entre outras - é outro objetivo deste trabalho.

O objetivo final é a organização e a documentação do trabalho realizado. Para isso cada caso de estudo é devidamente registado e documentado de forma adequada, o que inclui uma descrição textual do mesmo e a lista dos equipamentos de controlo e diálogo utilizados, o ambiente virtual concebido, um modelo formal do problema tratado, as linhas de orientação do *software* criado e, naturalmente, os ficheiros relativos ao código desenvolvido e às configurações dos dispositivos e das redes.

1.2. Organização da Dissertação

A dissertação está organizada em 6 capítulos, servindo o presente para a introduzir.

O Capítulo 2 – Justificação da Dissertação – justifica e define o trabalho a tratar na dissertação. Concretamente, aborda os sistemas distribuídos no contexto do controlo industrial e justifica a heterogeneidade dos mesmos no contexto dos equipamentos produtivos, das marcas e idades dos controladores e das respetivas formas de comunicação. Uma breve resenha da evolução tecnológica registada nesta matéria nas duas últimas décadas encerra o capítulo.

O Capítulo 3 – Recursos Tecnológicos de Suporte – elege e reúne as tecnologias necessárias ao desenvolvimento. Ou seja, define e apresenta por um lado, a ferramenta de recriação dos ambientes industriais e, por outro, os equipamentos que os irão controlar. As hipóteses de comunicação entre os diversos atores são também analisadas, experimentadas e reportadas neste capítulo.

O Capítulo 4 – Conceção de Modelos Fundamentais de Produção Flexível – introduz e define com o detalhe adequado as ferramentas de modelação à luz das quais serão representados os casos de estudo. Segue-se uma apresentação, auxiliada pelas normas de modelação referidas, de alguns blocos de controlo direto de elementos interativos presentes nos cenários virtuais realizados, assim como algumas soluções de controlo cooperativos entre PLCs. Estes elementos servirão de base à elaboração dos ambientes virtuais de maior complexidade a serem tratados no capítulo seguinte.

O Capítulo 5 – Modelação e Programação do Controlo Distribuído – é o mais longo e revela o desenvolvimento laboratorial efetuado. O ponto de partida são os aspetos teóricos obtidos no capítulo anterior. O caminho percorrido é o desenvolvimento de *software* de controlo e a implementação das comunicações entre componentes, e o ponto de chegada é a validação dos resultados. Uma vez que os condicionalismos inerentes à escrita de uma dissertação assim o exigem, diversas questões de pormenor são omissas neste capítulo. Porém, em anexo, é feita uma descrição mais pormenorizada de um dos casos de estudo desenvolvidos.

O Capítulo 6 – Conclusões e Trabalhos Futuros – encerra a dissertação. Nele se resumem as principais conclusões, se salientam os ensinamentos mais relevantes e se sugerem formas de dar continuidade ao trabalho realizado.

Capítulo 2 – Justificação da Dissertação

Este capítulo inicia-se com uma apresentação e justificação do trabalho a realizar nesta dissertação, sendo explicado em detalhe os objetivos e o plano do exercício proposto. Este ponto é acompanhado por uma comparação com dissertações similares de anos anteriores de modo a aferir acerca da originalidade do tema.

Será depois feita uma curta introdução teórica, complementada com alguma contextualização histórica dos tópicos científicos e tecnológicos centrais à dissertação. Estes tópicos são os sistemas de fabricação flexíveis, os dispositivos de controlo lógico e as redes de comunicação industrial.

O capítulo termina com um sumário das conclusões mais relevantes retiradas da análise das dissertações e da pesquisa bibliográfica efetuada no ponto anterior.

2.1. Problema Proposto

A motivação por detrás da realização deste trabalho de dissertação advém do interesse, por parte do autor, em reunir experiência e conhecimentos relacionados com a desmedida variedade de soluções de *hardware* e *software* de automação, de linguagens de programação e de redes e protocolos de comunicação industrial atualmente em uso em ambientes de fabricação.

Os desafios lançados na proposta desta dissertação foram, em primeiro lugar, a utilização de um *software* de criação, desenvolvimento e simulação de ambientes industriais virtuais como base para criar casos de estudo úteis para o trabalho a efetuar; em segundo lugar, a implementação dos algoritmos de controlo necessários para se observar o comportamento desejado nos cenários virtuais; em terceiro lugar, a distribuição do controlo de cada cenário sobre um conjunto de PLCs e a realização das comunicações entre eles; em quarto e último lugar, a especificação e representação dos casos de estudo e do controlo lógico com recurso a técnicas de modelação de sistemas discretos.

O *software* de simulação, chamado *Factory I/O* (1) e da autoria da *Real Games* (2), foi desenvolvido com o intuito de constituir um ambiente prático, acessível e seguro para o treino de controlo de instalações fabris através de controladores lógicos programáveis, ou PLCs. Para facilitar tal objetivo, o *Factory I/O* é compatível com uma multitude de protocolos de comunicação industrial comuns, possuindo ainda facilidades de comunicação direta entre os cenários virtuais e os controladores de alguns fabricantes.

Para responder ao desafio lançado, o autor teria à sua disposição uma significativa gama de equipamento de controlo industrial – PLCs e HMIs – duma grande variedade de idades e fabricantes. Como tal, seria possível implementar comunicações entre controladores sob diversos tipos de protocolos industriais e suportes físicos, desde comunicações série, com arquiteturas mestre-escravo, até às TCP/IP, norma que suporta o uso concorrente de múltiplos protocolos de comunicação dentro duma mesma rede *Ethernet*. Estas comunicações potenciariam o controlo distribuído dos casos de estado, constituídos pelos ambientes virtuais, através da repartição dos sensores e atuadores de cada cenário por dois ou mais controladores.

Outro requisito, era o de que o desenvolvimento do código dos controladores dos ambientes virtuais seria feito de modo a obter um desempenho fluido e fiável em cada cenário, e de forma a implementar fiel e naturalmente os algoritmos de controlo especificados nos modelos concebidos.

De modo a realizar a tarefa da especificação e documentação dos casos de estudo de forma consistente e sistemática, proceder-se-ia a uma seleção de ferramentas de modelação e análise de sistemas baseadas em redes de Petri e nas normas UML e *SysML*. A sua adoção permitiu a criação de um conjunto de modelos que permitiu especificar, apresentar e resumir cada um dos vários casos de estudo numa forma compacta, facilmente legível e abrangente. Deste modo, mostrar-se-ia a execução do trabalho sob várias vistas, incluindo a estrutura física dos ambientes virtuais, a implementação dos algoritmos de controlo e o estabelecimento das comunicações.

Uma busca de trabalhos anteriores relacionados com este tema destacou duas dissertações. A primeira (3) teve como objetivo a integração e interligação, numa rede de telemetria constituída por microcontroladores, de dois protocolos industriais, CAN e *FlexRay*. Esta atividade terá sido seguida por uma análise do desempenho e da fiabilidade da rede concebida. A segunda dissertação (4) focou-se na criação de um sistema de controlo e aquisição de dados de ensaios realizados num laboratório. Esse sistema terá sido constituído por módulos de I/O e consolas táteis distribuídas sobre uma rede CAN, supervisionados por um controlador central que tinha também por missão armazenar a informação dos ensaios numa base de dados.

A primeira dissertação referida aproxima-se um tanto ao trabalho aqui proposto, na medida em que envolveu a distribuição de controlo sobre uma rede constituída por um conjunto de protocolos industriais não compatíveis entre si. A segunda também se enquadra no trabalho de dissertação aqui proposto, em termos da prática do controlo distribuído, mas aborda mais a questão do controlo vertical, através da implementação de uma base de dados, tema que é tratado com menos pormenor neste trabalho.

2.2. Contextualização Histórica das Tecnologias da Automação

A atividade descrita nesta dissertação inicia-se com uma revisão bibliográfica acerca das tecnologias associadas à automação e controlo, nomeadamente, sistemas de fabricação, controladores lógicos e redes de comunicação industrial, assim como a sua evolução histórica e como esta conduziu ao presente estado da arte dos sistemas de fabricação.

2.2.1. Sistemas de Fabricação Flexíveis

Ao longo do último século, observou-se uma evolução progressiva, não só na área da automação, mas também na indústria como um todo. Este desenvolvimento iniciou-se nas décadas de 1950 e 1960 com a automatização da maquinaria e dos equipamentos de fabrico (5), que veio a diminuir a intervenção humana direta, substituindo operações manuais por operações parcial ou totalmente automáticas.

O desenvolvimento das tecnologias informáticas, que principiou na década de 1970 e que ainda decorre atualmente, permitiu uma utilização proeminente da informação, como dados de fabrico, vendas, procura, etc. Isto resultou numa revolução das áreas de engenharia e de gestão (5).

Isto tudo deu azo a ganhos de produtividade dos processos e qualidade dos produtos, mas trouxe também um crescente aumento da competitividade empresarial, agravando a necessidade de tirar aproveitamento de investimentos feitos no passado e o interesse em maximizar a produtividade dos sistemas automatizados.

Resultou isto numa crescente exigência da parte das tecnologias de automação. As soluções de automação, que se tratavam inicialmente de soluções rígidas, começaram a incluir sistemas de controlo reconfiguráveis ou reprogramáveis que potenciaram uma gama alargada de possibilidades de operações – e de sequências de operações - para um mesmo sistema ou célula de fabrico. Deu-se assim a introdução dos sistemas de fabricação programáveis e dos sistemas de fabricação flexíveis (6).

De forma concorrente, os sistemas de fabrico automático evoluíram de “ilhas de automação”, ou máquinas autónomas, para sistemas altamente integrados e flexíveis, compostos por máquinas alinhadas em série ou agrupadas em células com o controlo distribuído por conjuntos de controladores, controladores estes que são alvo da supervisão de um sistema de monitorização e aquisição de dados (7). Para estes sistemas de supervisão adotou-se a designação SCADA (*Supervisory Control And Data Acquisition*) (8).

Atualmente, discute-se sobre “Indústria 4.0” (9), termo que se refere à ideia de uma quarta revolução industrial, focada no desenvolvimento a nível mundial das tecnologias de sistemas ciber-físicos, computação embebida, comunicações e *Internet of Things*.

Os seus objetivos são a aplicação da produção em massa a baixo custo através da alavancagem de processamento e comunicações embebidas, a criação de sistemas de produção esbelta, eficiente, flexível e em tempo-real, facilitar o encadeamento das fases do ciclo de vida do produto, deste a ideia inicial até ao abate ou reciclagem, entre outros. Estes objetivos implicam o aumento da complexidade dos sistemas de produção e dos desafios enfrentados nas áreas industriais (9).

2.2.2. Controladores Lógicos Programáveis

O conceito de PLC (*Programmable Logical Controller*) (10) (11) refere-se a controladores com base em microprocessadores que utilizam memória programável para armazenar instruções programadas e posteriormente implementá-las com o fim de controlar máquinas e processos.

Os PLCs são similares aos computadores. Contudo, enquanto estes são otimizados para funções de cálculo e de interação com o utilizador, os PLCs são idealizados para tarefas de controlo e a ambientes industriais. Para satisfazer tais necessidades, os PLCs:

- São robustos e concebidos para suportar várias condições ambientais adversas como vibrações, ruído, humidade e temperatura;
- Têm interfaces, designadas por pontos de I/O (*inputs* e *outputs*), para entrada e saída de dados, binários ou analógicos, que permitem o controlo e monitorização do sistema controlado;
- São facilmente programáveis e exibem linguagens de programação fáceis de compreender e baseadas em operações lógicas (11).

A nível de *hardware* (Ilustração 2.2.1), um PLC típico apresenta os componentes básicos:

- Unidade central de processamento (CPU);
- Memória;
- Fonte de alimentação;
- Interface de I/O;
- Interface de comunicações;
- Dispositivo de programação.

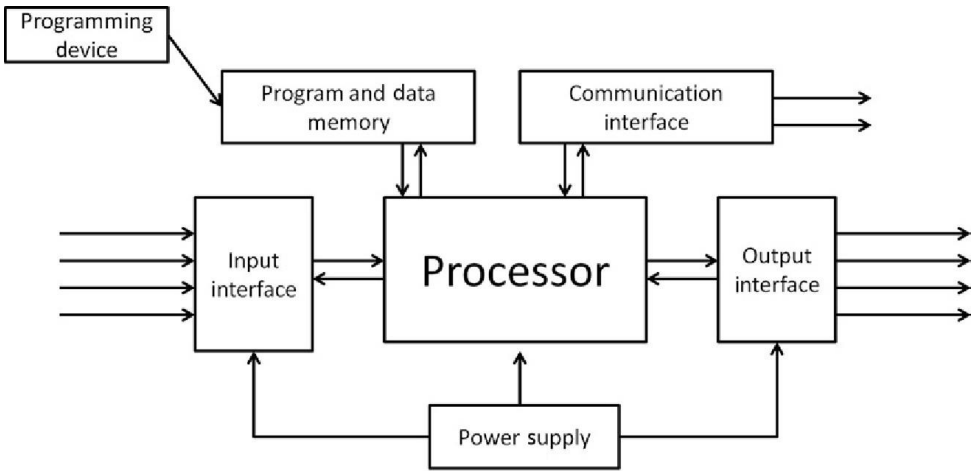

Ilustração 2.2.1: Componentes de um PLC (11)

Apesar da estrutura elementar dos PLCs se ter mantido quase inalterada desde a sua introdução, a sua complexidade, sofisticação e capacidades de desempenho evoluíram notavelmente. Inicialmente, os PLCs eram programados por consolas remotas ou neles encastráveis, que recorriam a uma simples linguagem gráfica, normalmente baseada em diagramas de contatos (*ladder diagram*), para permitir a um técnico exercer a programação do controlador. A constante evolução das exigências da indústria de automação e o progresso tecnológico obrigou a uma sofisticação das capacidades de programação dos PLCs e das capacidades de controlo dos mesmos.

Como resultado disso, os controladores lógicos são, hoje em dia, quase totalmente programados a partir de aplicações de *software* instaladas em computadores, aplicações essas que são cada vez mais complexas e possuem cada vez mais funções, como implementação de controladores PID e serviço de páginas *Web*.

Isto teve como consequência um aumento das competências exigidas dos programadores de PLCs, de modo que essa tarefa cada vez mais deixou de ser uma função de técnicos de automação, passando a recair nas mãos dos engenheiros de automação. Por sua vez, os PCs industriais tomaram lugar enquanto recursos de *hardware* e *software* de suporte aos PLCs.

Com o intuito de normalizar a programação de controladores lógicos, foi desenvolvida uma norma internacional, norma IEC 61131-3 (*International Electrotechnical Commission*), publicada em 1993 (11) (12), que define várias linguagens de programação. A norma define desde as linguagens gráficas de diagrama de contactos (LD), diagrama de blocos funcionais (FBD) e diagrama de funções sequenciais (SFC) às linguagens textuais de texto estruturado (ST) e lista de instruções (IL) (11) (12). Exemplos dalgumas destas linguagens estão presentes na Ilustração 2.2.2.

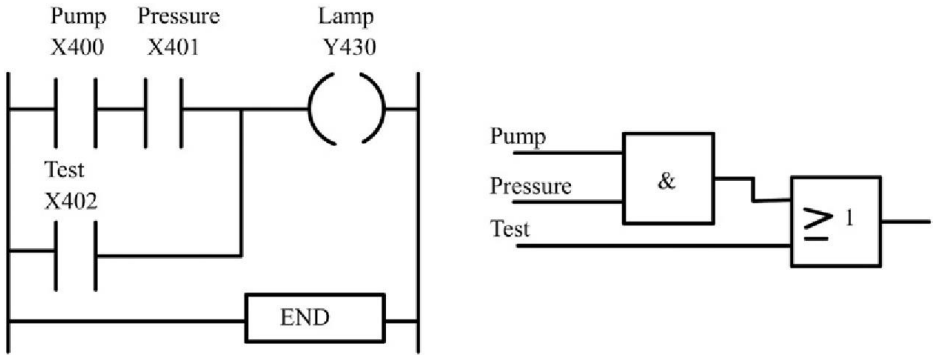

Ilustração 2.2.2: Um programa de PLC representado em *Ladder* (esquerda) e em blocos funcionais (direita) (11)

No entanto, a adesão a esta norma for concretizada apenas parcial e ultimamente. Devido à existência de tradições específicas a cada fabricante, estes continuaram a desenvolver os seus produtos à margem das normas existentes.

Associados aos PLCs estão os conceitos de monitorização e supervisão, daí surgindo o conceito de HMI (Ilustração 2.2.3). HMI (*Human Machine Interface*), ou MMI (*Man Machine Interface*), é um dispositivo que constitui uma interface entre utilizador e máquina (13).

No contexto de sistemas de controlo de processo, uma HMI oferece uma representação visual dum sistema, recorrendo à aquisição de dados em tempo real. As HMIs podem constituir parte de um centro de controlo centralizado que permita tanto aumentar a produtividade como ser altamente acessível aos utilizadores.

A utilização de HMIs e consolas apresenta múltiplas vantagens:

- Introduzem uma interface gráfica;
- O uso de cores e imagens facilita a leitura por parte do utilizador;
- Podem comunicar com PLCs em tempo real, adicionar funções de alarmes e modificar um sistema sem reprogramar o PLC;
- Podem gerir ou monitorizar os dados e a sua evolução;
- Podem aumentar a produtividade;
- São capazes de executar funções elaboradas através de, por exemplo, macros. (13)

Ilustração 2.2.3: Exemplos de consolas HMI (13)

Para que decorram as trocas de informações entre controladores, HMIs, sistemas de aquisição e supervisão de dados e outros dispositivos, são necessários recursos tecnológicos que as concretizem. Esses recursos consistem nas redes de comunicação industrial.

2.2.3. Redes de Comunicação Industrial

Com a evolução da tecnologia e com o aumento da competitividade dos mercados ao longo das décadas, foi surgindo uma grande heterogeneidade de dispositivos – como HMIs, máquinas de CNC, robôs, etc. – ao mesmo tempo que os sistemas de automação começaram a cobrir áreas geográficas consideráveis. Ainda mais, as redes industriais foram adquirindo uma notável importância e organização piramidal, abrangendo desde as redes de sensores e atuadores, nível mais baixo da pirâmide de automação, até ao nível de gestão e supervisão de setores inteiros duma fábrica ou até mesmo duma empresa (Ilustração 2.2.4). Apareceu ainda a necessidade de interligar redes industriais a redes convencionais, encontradas em ambientes empresariais (Tabela 2.2.1).

Tabela 2.2.1: Diferenças típicas entre redes industriais e convencionais (14)

	Industrial	Conventional
Primary Function	Control of physical equipment	Data processing and transfer
Applicable Domain	Manufacturing, processing and utility distribution	Corporate and home environments
Hierarchy	Deep, functionally separated hierarchies with many protocols and physical standards	Shallow, integrated hierarchies with uniform protocol and physical standard utilisation
Failure Severity	High	Low
Reliability Required	High	Moderate
Round Trip Times	250 μs - 10 ms	50+ ms
Determinism	High	Low
Data Composition	Small packets of periodic and aperiodic traffic	Large, aperiodic packets
Temporal Consistency	Required	Not required
Operating Environment	Hostile conditions, often featuring high levels of dust, heat and vibration	Clean environments, often specifically intended for sensitive equipment

Em consequência desta variedade de dispositivos, meios de comunicação e organizações hierárquicas, foi necessário o desenvolvimento progressivo de diversos protocolos, suportes físicos e topologias de rede adequados para satisfazer um conjunto de critérios conforme a aplicação (15). Critérios esses que são, por exemplo, tempo de resposta, taxa de transmissão de dados, tipo de dispositivos envolvidos e compatibilidade entre eles, tolerância a falhas e a fatores ambientais, etc.

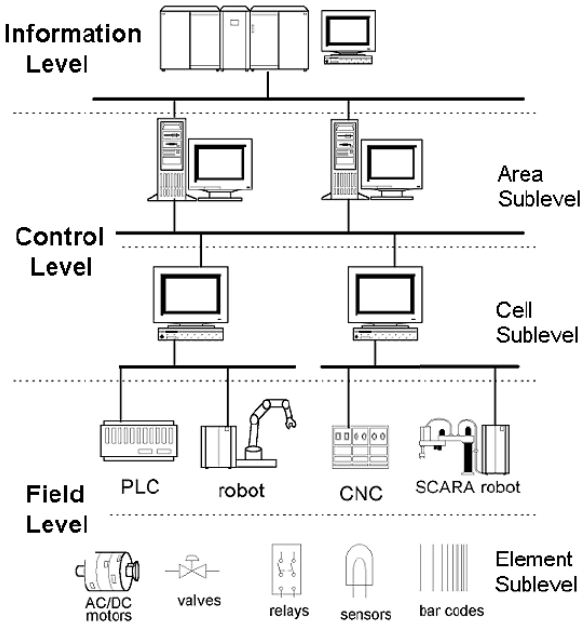

Ilustração 2.2.4: Organização hierárquica dum sistema de automação industrial (16)

De igual modo, esses protocolos e redes são altamente diversificados em termos de características funcionais e também a nível de acessibilidade. No princípio estes protocolos eram fechados (16), ou seja, constituíam propriedade das organizações que os criaram, mas, com a crescente necessidade de interligar recursos de diferentes fabricantes de equipamento industrial e informático, foram surgindo protocolos abertos que, por sua vez, não são diretamente compatíveis entre si.

Em 1969, foi desenvolvida, nos Estados Unidos, a norma RS-232, também conhecida como EIA/TIA-232, pela *Electronic Industry Association* e pela *Telecommunications Industry Association* em 1969. É a norma de comunicação série mais popular (11). Sobre ela foram mais tarde desenvolvidas as normas RS-422 e RS-485 que apresentam melhores características de desempenho, como a capacidade de poder interligar vários dispositivos na mesma rede série.

No início da década de 1970 deu-se início à conceção da *Ethernet*, quando investigadores da Universidade do Havai tentaram montar uma rede de comunicações entre ilhas utilizando transmissões de rádio (11). Mais tarde, a *Xerox Corporation* levou a iniciativa para a frente, tendo-a patenteado em 1975 (11). No início da década de 1980, o IEEE (*Institute of Electrical and Electronics Engineers*) tomou conta do desenvolvimento das normas *Ethernet* (11).

A *Ethernet* é largamente utilizada em redes de escritório. O seu desenvolvimento ao longo dos anos melhorou o seu desempenho temporal, tornando-a mais adequada a aplicações industriais. Isto permitiu a ligação entre as redes *Ethernet* industriais e de escritório, resultando numa mais fácil cooperação entre os níveis de produção e de gestão.

Em 1974 surgiu a norma TCP/IP (*Transmission Control Protocol* / *Internet Protocol*) (11). Originalmente desenvolvida pelo Departamento de Defesa Norte-Americano em 1974 e atualmente defendida pelo *Internet Architecture Board* e pela *Internet Engineering Task Force*, esta norma é responsável pela monitorização de trocas de dados entre o computador e a rede e define como os computadores acedem ao nível físico (11).

Em 1979 foi publicado pela *Modicon* o serviço de comunicação *Modbus*, destinado à troca de dados entre controladores lógicos programáveis. Este protocolo possui variantes, nomeadamente *Modbus* RTU e *Modbus* ASCII, para redes que utilizam série, e *Modbus* TCP, adaptado ser transmitido sobre o protocolo TCP/IP. As variantes série utilizam arquiteturas mestre/escravo enquanto *Modbus* TCP utiliza uma arquitetura cliente/servidor.

Em 1983, começou o desenvolvimento da norma CAN (*Controller Area Network*) pela *Robert Bosch GmbH*. Esta norma foi oficialmente publicado pela SAE (*Society of Automotive Engineers*) em 1986 e mais tarde como a norma ISO 11898 em 1993 (17). Foi originalmente desenvolvida para a indústria automóvel com o intuito de substituir a cablagem complexa de uma rede a dois fios. Esta especificação é bastante popular em áreas ou indústrias tais como automação, medicina e fabrico (18). Existem várias redes baseadas no protocolo CAN, tais como a rede *CANopen*, gerida pela CiA (*CAN in Automation*) ou a rede *DeviceNet*, criada pela *Allen-Bradley* e que é de momento propriedade da *Rockwell Automation* (18).

A norma tecnológica OPC (*Object linking and embedding for Process Control*) foi desenvolvida em 1996, numa tentativa de enfrentar o desafio de conferir interoperabilidade entre os vários fornecedores de soluções de automação (11). Funciona de acordo com um modelo cliente/servidor. Um fabricante pode desenvolver um servidor OPC para comunicar com uma fonte de dados ou estabelecer um conjunto de interfaces tal como especificadas na norma. Existem diversas especificações geridas pela fundação OPC:

- OPC DA (*Data Access*);
- OPC UA (*Unified Architecture*);
- OPC HDA (*Historical Data Access*);
- Etc.

A fundação OPC continua a atualizar as normas existentes para atingir os requisitos emergentes de interoperabilidade entre as tecnologias de automação (11).

Alguns dos protocolos utilizados nesta dissertação, em particular os protocolos fechados omissos neste capítulo, serão apresentados com melhor detalhe mais adiante neste relatório, no Capítulo 3.

Existem inúmeros outros protocolos e serviços de comunicação industrial desenvolvidos ao longo das últimas décadas, alguns sendo protocolos abertos e outros sendo proprietários. Cada serviço de comunicação é adequado a um determinado tipo de dispositivos ou um certo conjunto de requisitos funcionais e temporais. A necessidade de adotar novos protocolos, inseri-los dentro de sistemas de fabricação existentes e compatibilizá-los com suportes de comunicação antiquados é uma realidade com que os engenheiros de automação terão de viver nos próximos tempos.

2.3. Conclusões do Capítulo

À luz da análise e comparação de trabalhos de anos anteriores, aprecia-se que a dissertação proposta constitui um tema original e potencia um estudo bastante diversificado a nível de equipamento, maioritariamente graças à existência duma gama vasta – e tecnologicamente atual - de *software* e *hardware* de automação presentes no laboratório, enquanto as dissertações acima referidas foram mais restritas em termos de equipamento e protocolos usados. Não foi encontrado nenhum trabalho que tivesse como objetivo primário o treino de um estudante no âmbito do controlo distribuído e da coordenada de protocolos de comunicação industrial com a diversidade que esta dissertação apresenta.

A área da automação industrial enfrenta o constante desafio de criar protocolos e interfaces que permitam a coordenação e compatibilidade entre as várias tecnologias que lhe estão associadas, que frequentemente provêm de eras diferentes e fabricantes diversos e que envolvem diferentes níveis de aplicação, desde o controlo discreto à supervisão.

Quer pela necessidade de amortizar investimentos passados, quer pelo interesse em maximizar a produtividade dos sistemas de fabricação flexíveis, é pertinente a profissionais da área alcançar um certo domínio sobre o vasto leque de normas e meios de comunicação existentes.

Apesar dos esforços feitos para normalizar as tecnologias desta área, este objetivo está longe de ser alcançado, pelo que a heterogeneidade de soluções vivida nas últimas décadas mantem-se em grande parte.

Perante estas observações, prevê-se que o exercício proposto para esta dissertação justifica-se, constituindo uma forte e pertinente experiência de aprendizagem para o seu autor.

Capítulo 3 - Recursos Tecnológicos de Suporte

Neste capítulo são reunidos e estudados os recursos tecnológicos a utilizar no desenvolvimento da dissertação. Estes são constituídos, em primeiro lugar, pelo *software* de recriação de ambientes industriais, em segundo lugar, pelos controladores lógicos cujo fim será o controlo dos cenários e, em último lugar, pelos suportes físicos e protocolos de comunicação industrial.

A exposição do *software* de simulação, designado *Factory I/O*, será feita com o devido detalhe, sendo mostradas as suas funcionalidades, capacidades de comunicação e os vários elementos de construção de cenários virtuais a usar no trabalho.

A descrição de cada controlador será focada nas suas capacidades funcionais e de comunicação. Serão também referidos os equipamentos acessórios (módulos de expansão, placas de aquisição, etc.) que potenciaram uma mais flexível utilização dos dispositivos de controlo. Juntamente com os controladores serão apresentadas algumas consolas HMI a cuja utilização o autor desta dissertação teve acesso.

Como já foi mencionado, serão concorrentemente analisadas e comparadas as opções de comunicação realizáveis entre os diversos elementos, hipóteses essas que constituirão um dos principais fatores de seleção dos controladores. Após essa análise é apresentada uma tabela a resumir as hipóteses de comunicação industrial realizáveis entre os dispositivos estudados e os cenários virtuais.

O capítulo termina com algumas observações relativas ao estudo destes elementos tecnológicos. Dentre os controladores analisados, é feita uma seleção para utilizar na implementação a realizar no Capítulo 5. Esta seleção tem como fator dominante a variedade de formas de comunicação realizáveis de modo que estas constituam ferramentas de estudo aliciantes.

3.1. *Software Factory I/O*

Factory I/O (1) é um *software* de criação, desenvolvimento e simulação de ambientes industriais virtuais tridimensionais, produzido pela empresa portuguesa *Real Games* (2). Foi desenvolvido com o intuito de constituir um ambiente prático, acessível e seguro para o treino de controlo de instalações fabris através de controladores lógicos programáveis, ou PLCs.

3.1.1. Funcionalidades

Esta aplicação tem como principal objetivo constituir um ambiente de treino acessível para o projeto de instalações industriais, assim como para a programação de PLCs ou de outros controladores. Este programa apresenta dois modos de funcionamento: o modo de edição, no qual se construem e se editam os cenários virtuais; e o modo de corrida, onde os componentes do cenário entram em funcionamento, permitindo testar as soluções de controlo e de implementação.

O modo de corrida permite ainda ser colocar o ambiente virtual em pausa a qualquer momento, suspendendo-o, assim como reiniciá-lo de volta ao seu estado inicial.

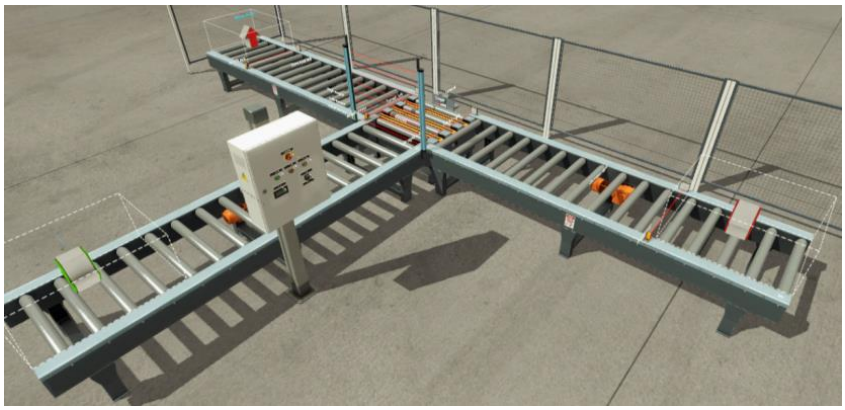

Ilustração 3.1.1: Exemplo de um cenário *Factory I/O*

Para tal, os cenários virtuais (Ilustração 3.1.1) podem ser construídos a partir de diversos componentes, tipicamente encontrados no âmbito industrial, que podem ser sensores, atuadores, ou sistemas de reduzida dimensão que combinam vários sensores e atuadores. O estado de cada sensor define o valor de uma variável transmitida ao controlador, enquanto os sinais vindos deste conduzem à operação de cada atuador virtual.

Ilustração 3.1.2: Variáveis *Factory I/O* em vários estados de operação

Para além da monitorização das variáveis, o *Factory I/O* permite forçar os seus valores ou impor avarias nos sensores e atuadores (Ilustração 3.1.2), permitindo assim testar a reação dum programa de PLC a diversas situações.

O *Factory I/O* é capaz de realizar comunicações com um controlador externo através do protocolo *Modbus* TCP, como cliente ou como servidor, e consegue estabelecer ligação a um servidor OPC DA. Para facilitar a ligação aos dispositivos de controlo, este *software* possui *drivers* de comunicação com uma multitude de controladores, PLCs, *SoftPLCs* e outros dispositivos de diversos fabricantes, como controladores *Siemens* e placas de I/O *Advantech*.

Configurada a ligação com um controlador, este poderá ler os valores de cada um dos sensores no ambiente *Factory I/O*, e comandar os atuadores disponíveis conforme a lógica segundo a qual foi programado. No entanto, por defeito, o *Factory I/O* apenas permite realizar a ligação do cenário com um controlador de cada vez ou, no caso da ligação por OPC, a um servidor de cada vez.

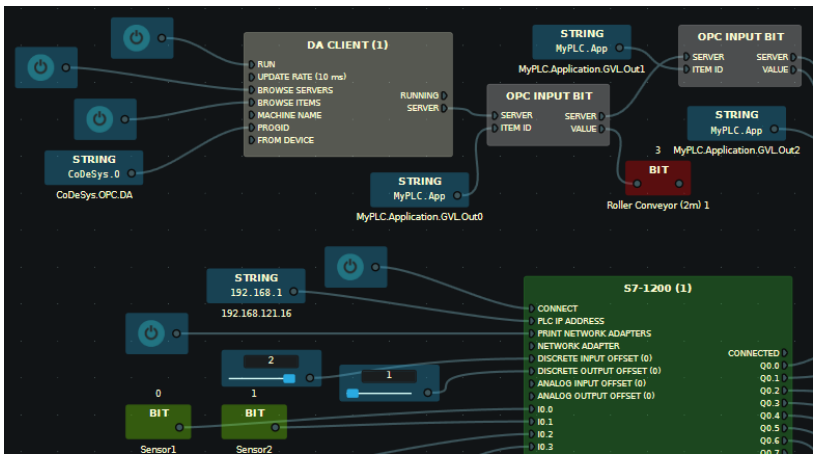

Ilustração 3.1.3: Excerto de ligações a PLCs estabelecidas no Connect I/O

Recorrendo ao *software Connect I/O* (19), da mesma empresa, é possível estabelecer ligações entre um cenário *Factory I/O* e vários dispositivos simultaneamente. A Ilustração 3.1.3 apresenta ligações do cenário a dois controladores lógicos, um por servidor OPC, o outro através de um *driver Siemens*.

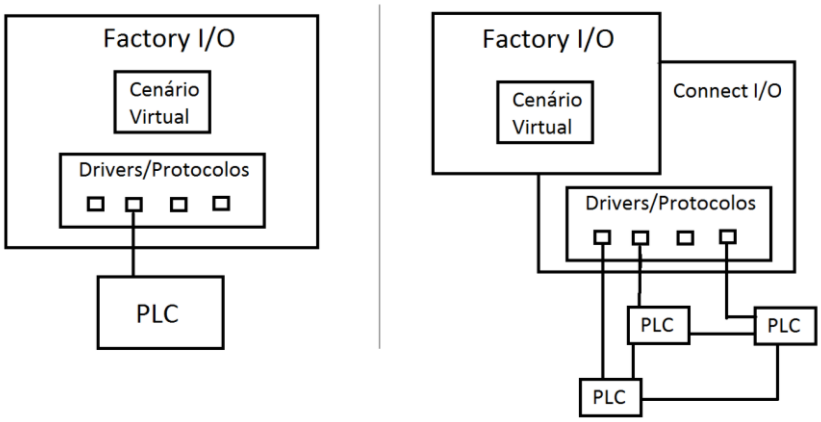

Ilustração 3.1.4: Ligação do cenário *Factory I/O* a um controlador sem o *Connect I/O*, à esquerda, e a vários controladores com o *Connect I/O*, à direita

A Ilustração 3.1.4 exemplifica as duas situações de comunicação acima descritas. Na primeira, representada à esquerda, o cenário está ligado a um único controlador por intermédio do *driver* ou protocolo selecionado. Na segunda, à direita, o *Connect I/O* permite a utilização de vários *drivers* ou protocolos simultaneamente, permitindo a ligação do cenário a vários controladores que, por sua vez, comunicam entre si.

3.1.2. Componentes

O *Factory I/O* apresenta uma multitude de componentes usados para construir diversos ambientes industriais automatizados. Nesta secção, introduzir-se-ão os componentes (20) usados na construção dos cenários virtuais.

Os componentes vão aqui ser introduzidos em ordem crescente de complexidade. Começar-se-á, portanto, por apresentar os itens transportáveis, depois os dispositivos de diálogo com os operadores, em seguida os sensores e atuadores, terminando com equipamentos dedicados que combinam múltiplos elementos de atuação e de sensorização.

3.1.2.1. Itens Transportáveis

Num processo industrial estão presentes materiais, produtos ou bens que devem ser movidos, manipulados ou processados para que tal atividade se desenrole. Nos ambientes virtuais gerados, foram utilizados itens, nomeadamente caixas e peças (Ilustração 3.1.5), para serem alvo de diversas operações de movimentação ou separação.

Ilustração 3.1.5: Itens

As caixas são distinguidas nos tamanhos "*palletizing*", pequeno, médio e grande. Quanto às peças, estas encontram-se disponíveis em duas cores, verde e azul, e em três tipos: matérias-primas, tampas e bases.

Ilustração 3.1.6: Paletes

Alguns dos atuadores existentes requerem que estes itens se encontrem pousados sobre paletes (Ilustração 3.1.6) para devidamente exercer a sua movimentação. Estas paletes estão disponíveis em duas variantes: quadradas e retangulares.

3.1.2.2. Emissor/eliminador

Estes dois componentes (Ilustração 3.1.7) têm como função a geração e eliminação de itens transportáveis. São úteis para a construção de casos de estudo do tipo "mover objeto de A até B", permitindo a emissão e eliminação contínua de itens.

O emissor tem como função gerar objetos para serem manipulados pelos outros componentes do ambiente virtual. Pode ser configurado para emitir itens de forma contínua dentro de intervalos especificados pelo utilizador. Permite também escolher quais os objetos a emitir e se estes se encontram ou não pousados sobre paletes e ainda se a sua posição e orientação é fixa ou arbitrária. O eliminador faz desaparecer quaisquer objetos transportáveis que entrem dentro do volume por ele delimitado. Ambos estes componentes são controláveis; exercem a sua função conforme o valor de uma variável de entrada inibidora.

Ilustração 3.1.7: Um emissor e um eliminador sobre um tapete de rolos

3.1.2.3. Dispositivos de Operação e Visualização

Trata-se de pequenos dispositivos que permitem ao utilizador enviar ordens ou dados conforme entender ao ambiente virtual, assim como obter informações sobre o estado atual dos componentes ou de ações que tenham sido ou venham a ser realizadas. Servem para emular botões ou sinalizadores que estão normalmente presentes nalgum painel de operador ao lado de uma instalação real.

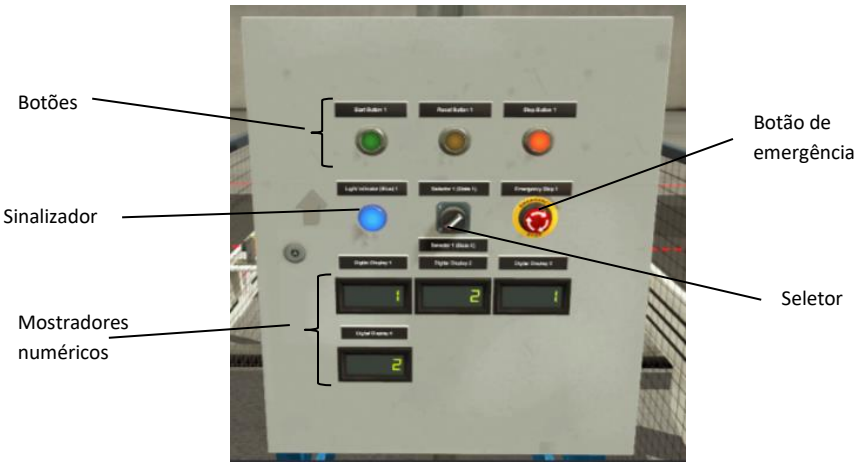

Ilustração 3.1.8 Painel de operador com diversos dispositivos de operação e visualização

O seu propósito pode também ser realizado através das funções de monotorização presentes em *software* de programação de PLCs ou, mais popularmente, através de uma HMI. Como é observável na Ilustração 3.1.8, os dispositivos utilizados incluem sinalizadores luminosos, botões, de atuação momentânea ou alternada, inclusive de emergência, e mostradores numéricos.

3.1.2.4. Sensores

Os sensores permitem a deteção ou identificação de itens, objetos e situações relevantes.

Sensores de posição

Existem vários tipos de sensores de posição. Estes são, por ordem crescente de alcance máximo, indutivo, capacitivo, difusivo e retrorrefletivo (Ilustração 3.1.9). O alcance de deteção de um sensor pode ser regulado dentro de uma gama específica para cada tipo. O sensor retrorrefletivo funciona com contato normalmente fechado e exige a presença de uma superfície refletora no extremo da linha de deteção.

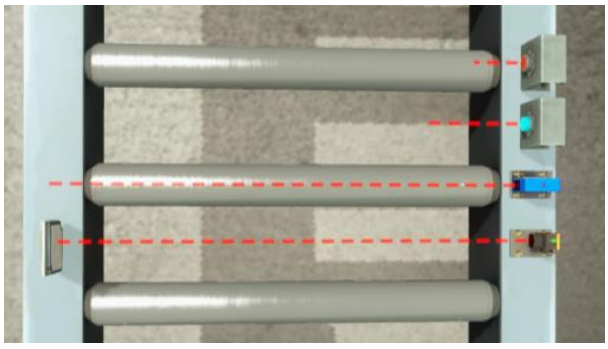

Ilustração 3.1.9: Sensores de posição, de cima para baixo, indutivo, capacitivo, difusivo e retrorrefletivo

Os sensores indutivo e capacitivo possuem um modo analógico no qual emitem um sinal compreendido no intervalo [0, 10] e com característica linear. O sensor retorna o valor máximo quando nenhum objeto é detetado e o valor mínimo quando um objeto se encontra encostado a ele.

Alguns equipamentos, tais como as mesas rotativas e os manipuladores *Pick and Place*, que serão apresentados mais adiante, incluem já os sensores necessários ao seu controlo.

Sensores de Identificação de Peças

Estes sensores permitem detetar a presença de peças verdes e azuis (Ilustração 3.1.10), assim como fazer a distinção entre as mesmas. Podem ser configurados para detetarem apenas um tipo de peça, retornando um sinal binário, ou para detetarem qualquer tipo de peça, fazendo a distinção entre elas.

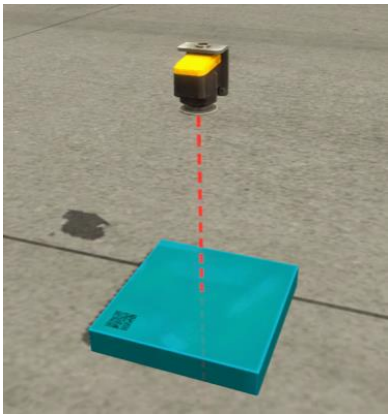

Ilustração 3.1.10 Sensor de visão a identificar uma peça

3.1.2.5. Tapetes Transportadores

Nos cenários desenvolvidos, estes tapetes serão proeminentemente utilizados. São atuadores simples cujo propósito é deslocar objetos na direção desejada. O Factory I/O disponibiliza três tipos de tapetes: tapetes de rolos, de tela (Ilustração 3.1.11) e angular de tela (Ilustração 3.1.12). Os tapetes de rolos são adequados para mover paletes, enquanto os de tela destinam-se a mover itens sem palete. Os tapetes angulares de tela permitem uma maior liberdade a nível da configuração física do cenário desenvolvido.

Ilustração 3.1.11: Tapete de rolos (esquerda) e tapete de tela (direita)

Existem três configurações possíveis; a digital e a analógica. Com a configuração digital, os tapetes ou estão parados, ou a mover-se a velocidade constante, conforme o valor dos sinais binários de entrada. Pode-se optar pela configuração digital unidirecional ou bidirecional.

Na configuração analógica, cada tapete regula a velocidade em resposta a uma variável real compreendida no intervalo [-10, 10]. A velocidade máxima disponível neste modo é superior à da configuração digital.

Ilustração 3.1.12: Tapete angular de tela

3.1.2.6. Mesa de transferência

Uma mesa de transferência (*chain transfer*) (Ilustração 3.1.13) tem como função deslocar peças segundo duas direções concorrentes. Numa direção, o movimento é feito por rolos, similarmente aos tapetes do mesmo tipo; na outra, a deslocação é feita por correias, que são elevadas acima dos rolos ao serem ativadas.

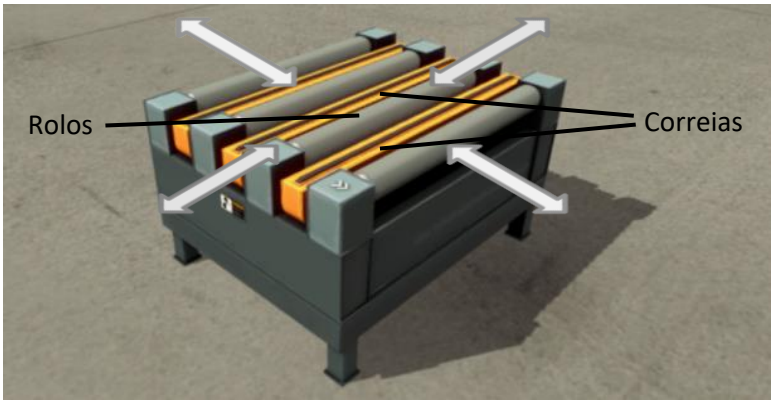

Ilustração 3.1.13: Mesa de transferência

3.1.2.7. Mesa rotativa

Uma mesa rotativa (*turntable*) (Ilustração 3.1.14) exerce a mesma função que as mesas de transferência com melhor fiabilidade mas com menor rapidez. A estrutura no seu centro pode rodar sobre si própria, permitindo escolherem qual das duas direções as paletes poderão ser movidas.

Estas mesas possuem dois sensores de posição, normalmente abertos, que permitem, através da sua observação, um correto posicionamento de uma palete, imediatamente antes de se efetuar uma rotação.

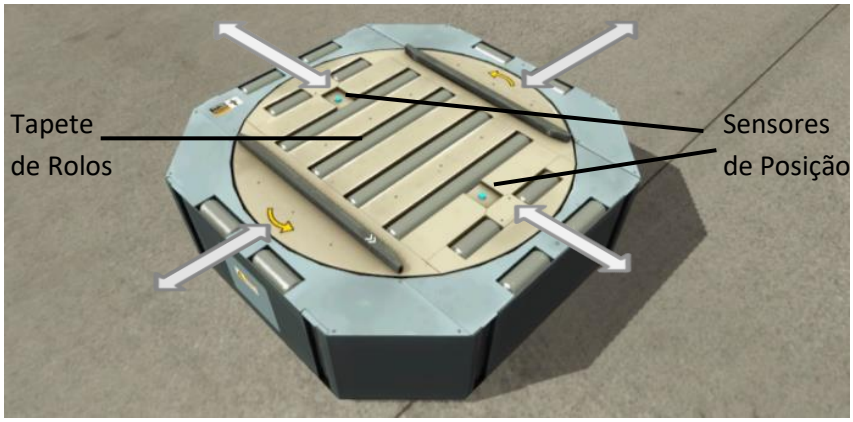

Ilustração 3.1.14: Mesa rotativa

Pode-se escolher entre duas configurações para a mesa: a monoestável e a biestável. Na primeira, a rotação é comandada por um só sinal binário. Quando este é nulo, a mesa roda até à sua posição inicial e mantém-se aí. Quando o sinal é positivo, a mesa roda até à posição de 90 graus relativamente à inicial.

Na configuração biestável, a ordem de rotação é gerida através de dois sinais binários (um para cada posição), podendo a mesa parar em qualquer orientação intermédia quando ambos os sinais são nulos.

3.1.2.8. Separador a rodízios

Um separador a rodízios (*pop up wheel sorter*) (Ilustração 3.1.15) permite, de modo similar às mesas anteriormente mencionadas, receber itens de e encaminhá-los para várias direções. Quando atuados, os rodízios sobem e giram de modo a deslocar qualquer objeto que esteja sobre elas pousado. Esses mesmos rodízios podem também reorientar-se 45 graus para a esquerda ou para a direita. Esta função torna os separadores adequados a serem anexados a tapetes angulares (Ilustração 3.1.16).

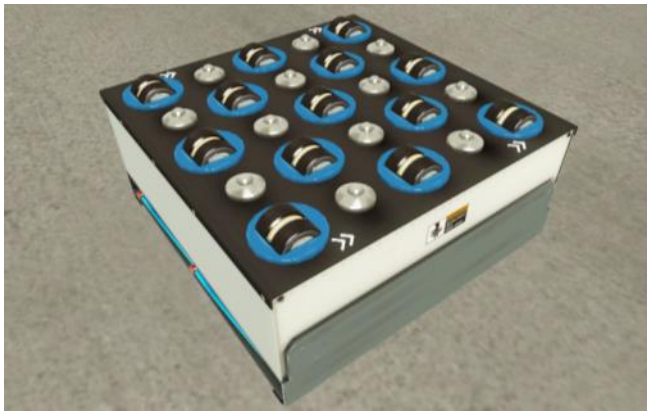

Ilustração 3.1.15: Separador a rodízios

Estes separadores são atuados apenas por sinais binários. É possível escolher entre os rodízios poderem girar num só sentido ou em ambos os sentidos.

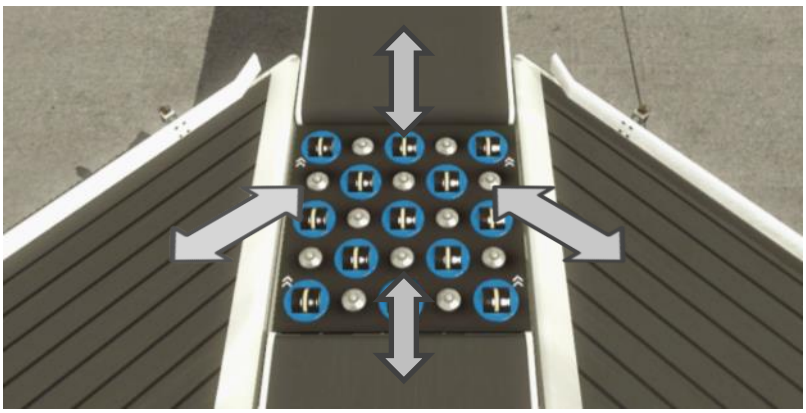

Ilustração 3.1.16: Separador a rodas com tapetes anexados

3.1.2.9. Manipuladores Pick and Place

Este último componente aqui discutido (Ilustração 3.1.17) assemelha-se aos manipuladores robóticos, não raramente encontrados na indústria, cuja função é pegar em objetos e pousá-los (*pick and place*) com apreciável precisão.

O manipulador utilizado é de estrutura cartesiana, com dois eixos, um horizontal e o outro vertical. Os manipuladores possuem ainda uma ferramenta para levantar peças, semelhante a uma ventosa pneumática, e, no extremo desta, um sensor de posição, do tipo normalmente aberto.

Ilustração 3.1.17: Manipulador *Pick and Place* de 2 eixos

Existem duas configurações possíveis no que diz respeito ao controlo dos eixos dos manipuladores. Na configuração digital, cada eixo está a estender-se ou a retrair-se conforme o valor do respetivo sinal binário de entrada, devolvendo outro sinal binário que é positivo quando o eixo se desloca, caso contrário é nulo. Na configuração analógica, cada eixo desloca-se até um ponto específico do seu curso, indicado por uma variável de entrada real contida no intervalo [0, 10], devolvendo outra variável real, contida no mesmo intervalo, o valor atual da posição do eixo, que é continuamente atualizado.

Das duas configurações disponíveis, considera-se que a analógica é a mais fiável e a mais segura, pois permite uma observação mais precisa do estado atual do robô, possuindo também a opção de parar um eixo em qualquer ponto do seu curso.

Existe ainda uma variante dos manipuladores Pick and Place com três eixos ortogonais de movimentação, à qual não se recorreu nesta dissertação.

3.1.2.10. Outros Componentes

No *Factory I/O*, existem diversos outros componentes que não serão utilizados no trabalho descrito nesta dissertação e que são dedicados a determinadas funções. Estes incluem equipamentos que permitem organizar ou distribuir itens transportáveis, tais como máquinas paletizadoras e armazéns automatizáveis. Estão também disponíveis equipamentos mais complexos, tais como tanques de água ou centrais de maquinagem.

É possível obter mais informações acerca dos componentes referidos e outros no manual *online* do *Factory I/O* (20), disponível no site da *Real Games* (2). Podem ainda ser visualizadas inúmeras demonstrações das aplicações do *Factory I/O* em formato vídeo no canal *Youtube* da *Real Games* (21).

3.2. Análise dos Controladores

Nesta secção introduzem-se os dispositivos de controlo industrial a utilizar na implementação das soluções de controlo e monitorização dos cenários virtuais.

A apresentação de cada controlador começa por resumir as características funcionais do modelo utilizado, inclusive o *hardware* adicional a ele instalável e as potencialidades do respetivo *software* de programação. Depois, são sumariadas as capacidades de comunicação do controlador. Algumas redes ou protocolos de comunicação específicos ao controlador são também brevemente explicadas. A exposição de cada controlador termina com observações relativas à sua aplicabilidade no âmbito da dissertação, feitas à luz das suas potencialidades de comunicação industrial, assim como da sua atualidade tecnológica.

O ponto 3.2.1 (*SoftPLCs Codesys*) faz pouca menção às características funcionais dos respetivos controladores, dado que estas estão condicionadas ao desempenho dos PCs que os executam, pelo que este ponto foca-se principalmente no *software* de programação e nas opções de comunicação. No final desta secção, é descrita uma consola HMI que será aplicada no controlo dalguns dos cenários.

3.2.1. *SoftPLCs* Codesys

Um *SoftPLC* é uma aplicação que permite que um computador emule o comportamento e funcionalidades de um controlador lógico programável. Neste trabalho destacam-se os *SoftPLCs Control Win* V3 e *SP PLCWinNT* V2.4, desenvolvidos pela *3S-Smart Software Solutions GmbH* (22).

Estes dois controladores são programáveis através das aplicações de *software Codesys* V3.5 e *Codesys* V2.3, respetivamente. São ferramentas de programação altamente compatíveis com a norma IEC 61131-3, permitindo o uso de todas as linguagens definidas pela norma.

3.2.1.1. Capacidades de Comunicação

Estes *SoftPLCs* são capazes de utilizar as portas de comunicação do computador no qual estão a ser corridos de modo a desempenhar trocas de dados com o exterior. Isto permite a comunicação série ou por protocolo TCP/IP sobre *Ethernet*. No entanto, comunicações que não usem portas série ou *Ethernet* exigem a instalação de *hardware* adicional no computador, como é o caso das redes CAN.

Ambos os *SoftPLCs* conseguem realizar comunicação série por protocolo *Modbus* RTU. Podem utilizar este protocolo como mestres, como escravos, ou ambos em simultâneo. Podem também utilizar o protocolo *Modbus* TCP/IP (como clientes ou como servidores ou ambos simultaneamente) ou por *Profinet* através de uma rede Ethernet, suportando ainda variáveis de rede sob o protocolo UDP/IP.

Podem comunicar através de redes CAN tais como *CANopen* ou *DeviceNet* . O *Codesys Control Win* V3 pode ainda comunicar como mestre ou como escravo numa rede *Profibus* DP.

Ambas as versões *Codesys* mencionadas possuem os seus próprios servidores OPC. Neste trabalho, utilizou-se o mais recente dos dois, o *Codesys* OPC *Server* V3. Esta aplicação é compatível com os dois *softPLCs* descritos. A ligação é permitida através das interfaces *Gateway* V3 e *Gateway* V2.3 para as versões *Codesys* V3.5 e V2.3, respetivamente (23). Na

Ilustração 3.2.1 apresenta-se um exemplo duma possível configuração de um servidor OPC *Codesys* V3.

No entanto, os dois controladores não possuem funcionalidade enquanto clientes OPC, pelo que não podem alterar variáveis ou itens pertencentes a PLCs diferentes, tendo apenas controlo sobre as variáveis a serem lidas pelo servidor. Estes dois *SoftPLCs* podem partilhar o mesmo servidor mas não as mesmas variáveis, sendo necessário um cliente OPC, como por exemplo o *Connect I/O*, para os interligar.

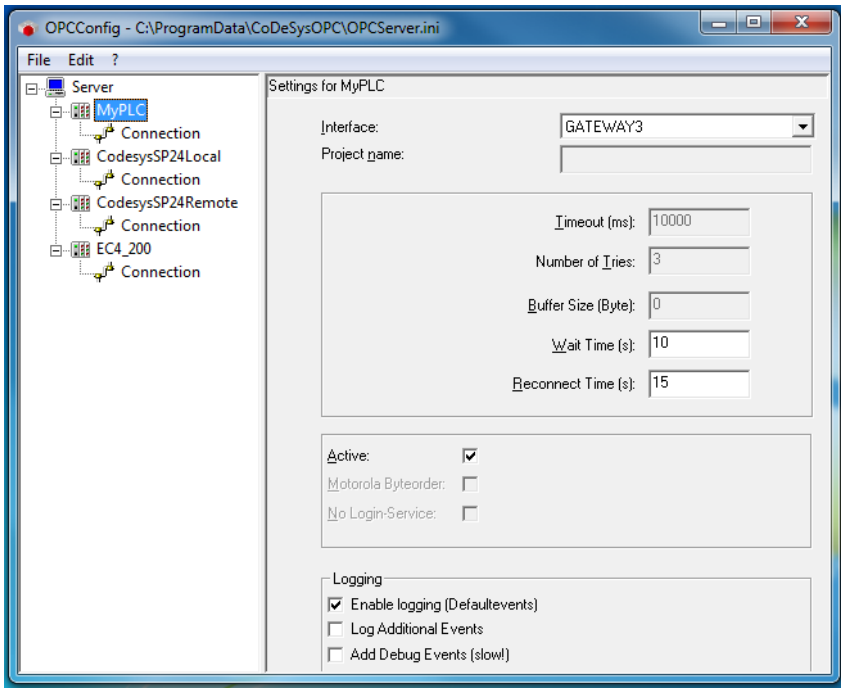

Ilustração 3.2.1: Exemplo da configuração de um servidor *Codesys OPC Server* V3

3.2.1.2. Potencialidades no Contexto da Dissertação

Estas aplicações podem ser facilmente corridas num PC e possuem acesso a uma diversidade de métodos de comunicação, das quais se destacam *Modbus* TCP e variáveis de rede por TCP. A sua ligação ao *Factory I/O* é facilmente realizável via servidor OPC, ou, alternativamente, por *Modbus* TCP.

É possível ligar dois ou mais destes *SoftPLCs* ao mesmo servidor OPC, mas não é possível estabelecer ligação direta por este meio.

3.2.2. *Siemens* SIMATIC S7-1200

Os controladores Siemens das séries S7-1200/1500 foram introduzidos nos anos 2010 e 2013, respetivamente, em substituição aos das séries S7-200/300/400, mais antigas.

Neste trabalho foi utilizado um PLC Siemens S7-1200, modelo CPU 1214C AC/DC/Rly (Ilustração 3.2.2) que ao qual foram instalados os módulos de comunicação:

- CM1241 (RS485) 6ES7 241-1CH30-0XB0;
- CM1241 (RS232) 6ES7 241-1AH30-0XB0;

3.2.2.1. Características funcionais

Estes PLCs são programados com o *software* STEP 7 – TIA Portal BASIC V13, da *Siemens*. Trata-se de um *software* sofisticado, que apresenta uma multitude de funções de controlo lógico e de estruturação do programa. Permite programar em LD, FBD e em linguagem de controlo estruturado (SCL). Apresenta ainda a funcionalidade de permitir programar vários dispositivos, inclusive HMIs e I/O distribuído, dentro do mesmo projeto.

Ilustração 3.2.2: Controlador S7-1200 utilizado com módulos de comunicação série

A Tabela 3.2.1 apresenta algumas características funcionais do modelo controlador utilizado.

Tabela 3.2.1: Características funcionais do CPU 1214C AC/DC/Rly (24)

Tensão de Alimentação	120 a 240 V AC
Entradas físicas digitais	14, 24 V DC
Saídas físicas digitais	10, por relé
Entradas físicas analógicas	2
Porta de comunicação	Ethernet
Expansível por:	
Módulos de sinal, até	8
Placas de sinal, bateria ou comunicação, até	1
Módulos de comunicação, até	3

Aos controladores da série S7-1200 podem ser instalados módulos de sinal, para adicionar pontos de I/O, ou módulos de comunicação, para adicionar portas de comunicação (Ilustração 3.2.3). Neste último caso, podem ser adicionadas uma variedade de interfaces, incluindo portas do tipo série, que vão ser especificadas adiante.

3.2.2.2. Capacidades de Comunicação

Os controladores das séries S7-1200 e S7-1500 são altamente versáteis quanto aos modos de comunicação realizáveis. Isso permite a sua interligação a e cooperação com uma diversidade de dispositivos, em particular outros produtos Siemens (24).

O PLC pode ser ligado a redes série RS232/422/285 através da adição de portas série por módulos específicos (24). As portas série permitem comunicar através dos protocolos:

- PtP;
- USS;
- *Modbus* (24).

PtP permite efetuar comunicação sem protocolo através de uma porta série. É um tipo de comunicação à base de caracteres ASCII, que permite uma grande liberdade em como os dados são transmitidos mas requer uma extensa e cuidadosa implementação no próprio programa dos controladores envolvidos. É equivalente ao modo *Freeport* presente nos S7-200, também da *Siemens*, detalhado mais adiante nesta dissertação (24) (25).

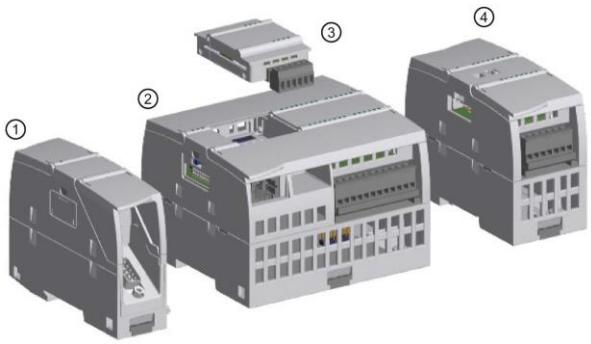

Ilustração 3.2.3: S7-1200 com módulos e placas de expansão (24)

USS é um protocolo especializado para o controlo de *drives* de motores, pelo que não é relevante a este trabalho.

Todos estes protocolos série podem ainda ser utilizados sobre redes *Profibus* ou *Profinet*, desde que os dispositivos a interligar sejam compatíveis com esta funcionalidade (24).

Os S7-1200 conseguem comunicar com o protocolo *Modbus*, variantes RTU e TCP, como mestres ou como escravos. No caso de *Modbus* RTU, como foi mencionado, requerem-se módulos para instalar as portas série necessárias (24). Para *Modbus* TCP, é utilizada a porta *Ethernet* do controlador.

Os S7-1200 podem utilizar as redes *Profibus* e *Profinet*, através das quais podem realizar trocas de dados com outros dispositivos compatíveis. O *software* de programação também possui instruções específicas para permitir comunicações entre dispositivos da série S7 (tais como HMIs ou outros PLCs) de forma fácil e eficaz (24). A comunicação por *Profinet* recorre à porta Ethernet do controlador, enquanto a comunicação por *Profibus* requer módulos específicos para a sua utilização.

3.2.2.3. HMI: Siemens SIMATIC HMI

Para esta dissertação foi disponibilizada uma HMI SIMATIC KTP600 Basic Color.

Trata-se de uma consola programável concebida para apresentar opções de comunicação facilitada com produtos Siemens, em particular controladores das séries S7-200/300/400/1200/1500. Esta ligação é realizada através de protocolos proprietários sobre redes *Profibus* ou *Profinet* (26).

São programadas com o *Step* 7 - TIA *Portal* V13, possibilitando a sua programação em conjunto com a dos controladores a interligar, pertencendo estes às séries S7-1200/1500.

Ilustração 3.2.4: Consola *Siemens* SIMATIC KTP600 Basic (26)

3.2.2.4. Potencialidades no Contexto da Dissertação

Esta série de controladores consegue comunicar por uma diversidade de meios e protocolos, sendo capaz de exercer ligação com praticamente todos os restantes controladores mencionados neste capítulo.

A comunicação com PLCs *Siemens* da série S7-200/300/400 pode ser efetuada sobre *Profibus*, *Profinet*, ou por comunicação série através de *Modbus* RTU ou por PtP.

O *Factory I/O* dispõe de um *driver* pré-instalado para estabelecer comunicação com PLCs *Siemens* S7-1200 ou S7-1500. Este programa pode também comunicar com o S7-1200 por *Modbus* TCP/IP sobre *Ethernet*.

3.2.3. *Siemens* SIMATIC S7-200

Trata-se da série de controladores mais antiga da *Siemens*, vendida desde o final da década de 1980 e atualizada continuamente até ao início deste século, que foi substituída pelas séries S7-300/400 e, mais tarde, pelas mais modernas S7-1200/1500. Neste trabalho, foi utilizado um S7-200, modelo CPU 244 DC/DC/DC (Ilustração 3.2.5), a cujos pontos de I/O esteve ligada uma placa de aquisição *Advantech* USB-4750.

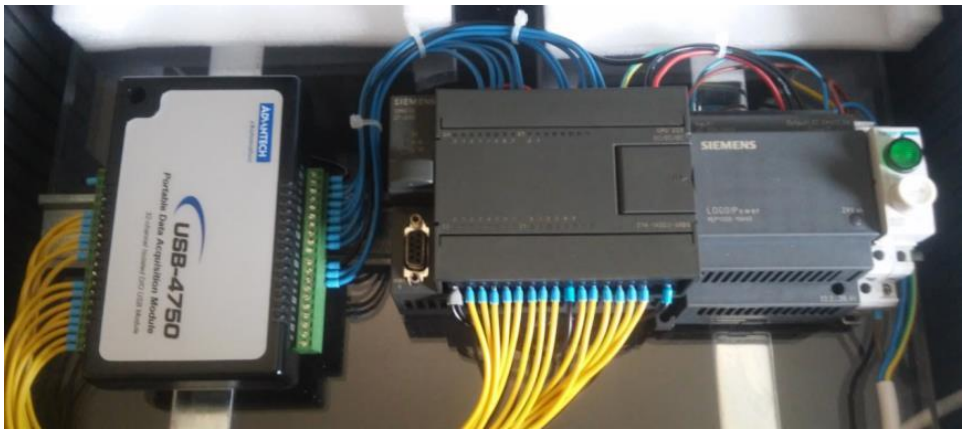

Ilustração 3.2.5: Controlador *Siemens* CPU 244 DC/DC/DC ligado a placa *Advantech*

3.2.3.1. Características funcionais

Os S7-200 são programáveis a partir do *software STEP 7 – Micro/WIN* (25). É menos flexível a nível de funcionalidades, comparativamente ao *software* de programação das séries S7-1200/1500. Permite programar nas linguagens LD, FBD e ST, não sendo, no entanto, compatível com a norma IEC 61131.

Outras características funcionais do modelo do PLC utilizado são apresentadas na Tabela 3.2.2.

Tabela 3.2.2: Características funcionais do *Siemens* CPU 244 DC/DC/DC (25)

Tensão de alimentação	24 V DC
Entradas/Saídas digitais físicas	14/10, 24 V DC
Porta de comunicação	RS-485
Expansibilidade por módulos:	
De I/O ou de comunicação	Até 7 módulos

O controlador S7-200 utilizado possui uma porta de comunicação série (Ilustração 3.2.6). As suas capacidades de sinal e de comunicação podem ser expandidas através de módulos:

- Discretos, que aumentam o número de pontos de I/O digitais físicos;
- Analógicos, que adicionam pontos de I/O analógicos;
- Inteligentes, que permitem estabelecer comunicação através de uma variedade de redes.

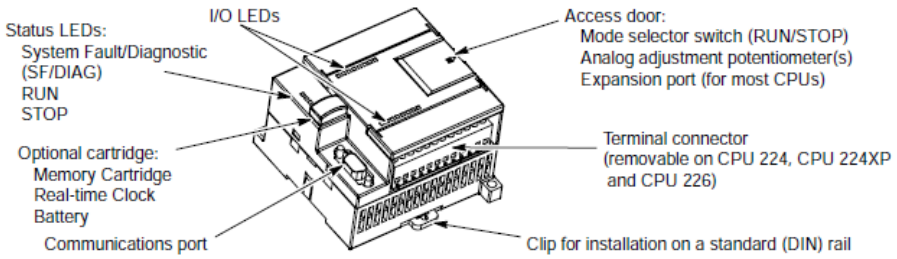

Ilustração 3.2.6: Componentes de um controlador S7-200 (25)

Utilizando módulos inteligentes, o controlador pode comunicar através das redes *Ethernet*, Profibus DP e AS-i, podendo também estabelecer ligação com *Modems* (25).

3.2.3.2. Capacidades de comunicação

Os S7-200 suportam os protocolos de comunicação através da porta série:

- *Point-to-Point* Interface (PPI);
- *Multi-Point* Interface (MPI);
- *Modbus*;
- *Freeport*;
- *Profibus*.

Protocolo PPI (*Point to Point Interface*):

Este protocolo do qual a *Siemens* é proprietária é utilizado estabelecer ligação entre dispositivos Siemens tais como PLCs da série S7-200/300/400 e HMIs, permitindo também a comunicação desses dispositivos com PCs.

Trata-se de um protocolo *master-slave* com possibilidade de configurar vários mestres. Permite a instalação de até 32 mestres numa só rede. A comunicação entre mestres e escravos é feita sobre uma ligação partilhada por eles que é gerida pelo protocolo PPI. Contudo, nem todos os dispositivos *Siemens* podem funcionar como mestres (25).

Protocolo MPI (*Multi Point Interface*):

Consiste noutro protocolo da *Sie*mens que é similar ao PPI. Permite tanto comunicação de mestre para escravo como entre mestres (25).

Existe uma opção de comunicação série sem protocolo idêntica ao PtP das séries S7-1200 e S7-1500, designada por *Freeport*. Esta funcionalidade permite realizar comunicações série entre PLCs desta série e outros dispositivos na ausência de um protocolo de comunicação comum entre os dois.

É possível realizar comunicações através dos protocolos *Profibus*, *Profinet*, TCP/IP sobre *Ethernet* e AS-i (*Actuator Sensor Interface*), sendo para isso necessária a instalação de módulos de comunicação específicos (25).

3.2.3.3. Potencialidades no Contexto da Dissertação

A porta série pode ser utilizada para comunicar por *Modbus* série ou, através do modo Freeport, estabelecer trocas de dados sem protocolo.

Através das redes *Profibus* ou *Profinet*, a comunicação com PLCs das séries de controladores S7-1200/1500 pode ser realizada utilizando instruções neles existentes. Este modo permite apenas as estas séries de controladores dar instruções de leitura e escrita aos S7-200, desempenhando estes o papel de escravos na rede (24).

O *Factory I/O* inclui um *driver* para comunicar com os PLC S7-200, podendo também estabelecer ligações por *Modbus* TCP. Ambos estes meios requerem uma porta *Ethernet* instalada no controlador, à qual não se teve acesso na realização deste trabalho. Optou-se, portanto, por ligar o S7-200 ao Factory I/O por intermédio da placa *Advantech*.

3.2.4. PLC Moeller Easy Control EC4-200

Esta série de PLCs foi em 2006 fabricada pela *Moeller*, que foi mais tarde adquirida pela *Eaton*. Esta série contém o *hardware* de um modelo anterior, que foi adaptado para ser utilizado com o *software Codesys*. Neste trabalho, utilizou-se um controlador *Moeller* EC4P-222-MRAD1 (Ilustração 3.2.7), ao qual estiveram ligados os dispositivos:

- Unidade de expansão de I/O EASY620-DC-TE;
- Placa de aquisição *Advantech* USB-4750 ligada aos pontos de I/O do controlador.

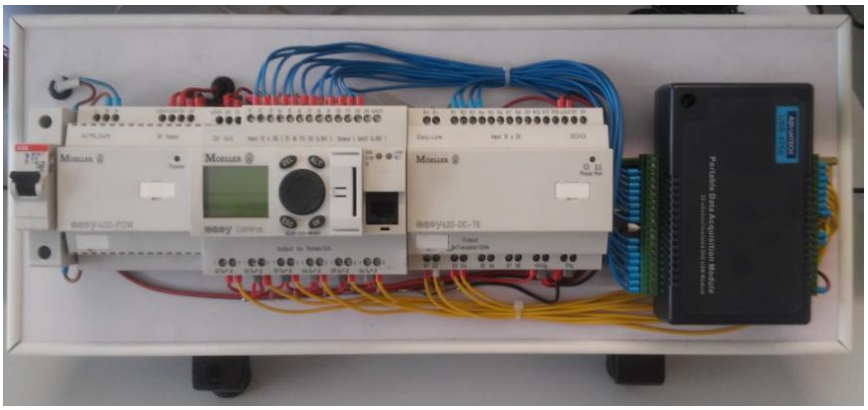

Ilustração 3.2.7: *Moeller* EC4-222-MRAD1 com unidade de expansão e placa *Advantech*

3.2.4.1. Características Funcionais

A programação deste controlador é feita através do *software Codesys* V2.3, tal como o *SoftPLC SP PLCWinNT* V2.4. A Tabela 3.2.3 apresenta as características funcionais do modelo usado:

Tabela 3.2.3: Características funcionais do EC4-222-MRDA1 (27)

Tensão de alimentação	24 V DC
Entradas digitais	12, a 12 V DC
Das quais podem ser utilizadas como analógicas	4, de 0 a 10 V DC
Saídas digitais	6, do tipo relé
Saídas analógicas	1, de 0 a 10 V DC
Portas de comunicação:	
RJ45	Porta 1, para programação, comunicação série, ou comunicação por Ethernet
	Porta 2, para comunicação por CAN
Expansibilidade	Módulos de I/O e módulos de rede

3.2.4.2. Capacidades de Comunicação

Com a instalação de módulos de rede, estes controladores podem ser ligados às redes AS-i, *Profibus* DP, CAN ou *DeviceNet* com os módulos de rede adequados instalados. Nestas redes, o controlador apenas pode funcionar como escravo e não como mestre.

Um controlador EC4-200 pode ser utilizado como um controlador individual ou ligado a dispositivos de I/O através de uma rede *CANopen*. Esta ligação também permite comunicar com outros dispositivos que utilizem *CANopen*.

Suporta também variáveis de rede sobre CAN. Pode comunicar por *Modbus* TCP sobre *Ethernet* ou por *Modbus* RTU utilizando a porta série, como mestre ou como escravo.

3.2.4.3. Potencialidades no Contexto da Dissertação

Este controlador possui uma interface *Ethernet*, mas só a pode utilizar para efeitos de programação ou para aceder a um servidor OPC. Pode ser ligado a redes CAN, em particular às redes *CANopen* e *DeviceNet*, ou a uma rede *Profibus-DP*, necessitando de interfaces adicionais para estes casos. Sobre as redes CAN, suporta ainda o uso de variáveis de rede.

Pode comunicar por *Modbus*, variantes RTU ou TCP, enquanto mestre ou enquanto escravo. Esta é a opção mais viável para comunicação com os outros controladores.

3.2.5. *Omron* SYSMAC CP1L

A série de controladores SYSMAC CP1L foi produzida em 2007 pela *Omron*. Trata-se de um melhoramento sobre a série SYSMAC CPM1A, cuja produção se iniciou em 1997.

Para a execução deste trabalho, esteve disponível um SYSMAC CP1L-M30DR-D (Ilustração 3.2.8), ao qual estiveram ligados os dispositivos:

- *Option Boards*:
  - *Ethernet/IP* CP1W-EIP61;
  - *Modbus* TCP/IP CP1W-MODTCP61;
- Placa de aquisição *Advantech* USB-4750.

Ilustração 3.2.8: *Omron* SYSMAC CP1L-M30DR-D usado, com duas *option boards Ethernet* e a placa *Advantech*

3.2.5.1. Características funcionais

Este controlador é programado através do *software CX-Programmer* (28). Permite programar em linguagens LD, SFC (*Sequential Function Chart*) e ST, não compatíveis com IEC 61131. As funções de organização do programa também não são tão versáteis como nos *softwares* de programação anteriormente mencionados.

A Tabela 3.2.4 apresenta algumas características relevantes do modelo utilizado.

Tabela 3.2.4: Características funcionais dos controladores *Omron* estudados (28)

Controlador:	CP1L-M30DR-D
Tensão de alimentação:	24 V DC
Inputs discretas:	18, a 24 V DC
Outputs discretas:	12, do tipo relé
Expansibilidade:	
I/O discreto:	Até 3 unidades
I/O analógico:	
Option Boards:	Até duas
Portas de comunicação:	Portas série ou *Ethernet* adicionáveis através das *Option Boards*

A esta série de controladores podem ser ligadas até três unidades de expansão (Ilustração 3.2.9) para aumentar o número de pontos de I/O, quer discretos, quer analógicos (28).

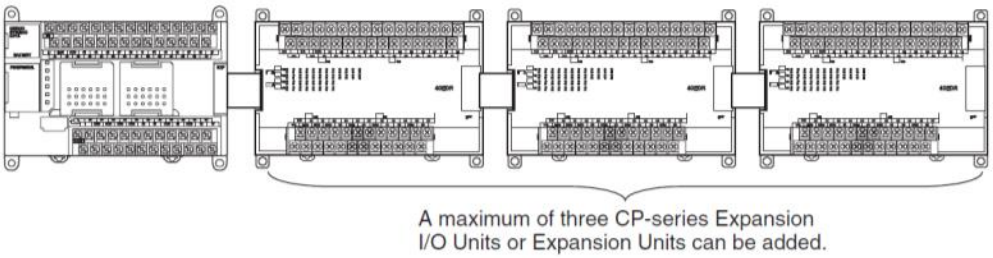

Ilustração 3.2.9: Ligação de um *Omron* CP1L a três unidades de I/O (28)

3.2.5.2. Capacidades de Comunicação

Existem, exclusivamente para os CP1L, módulos de dimensão reduzida, ditas *option boards* (Ilustração 3.2.10), que permitem adicionar portas RS-232C, RS-422A/485 e Ethernet, entre outras funcionalidades. Entre as *option boards*, existem algumas conferem ao dispositivo funcionalidades para comunicar através de determinados protocolos, tais como *Modbus* TCP e *Ethernet/IP*. Os modelos CP1L-M suportam até duas *option boards*.

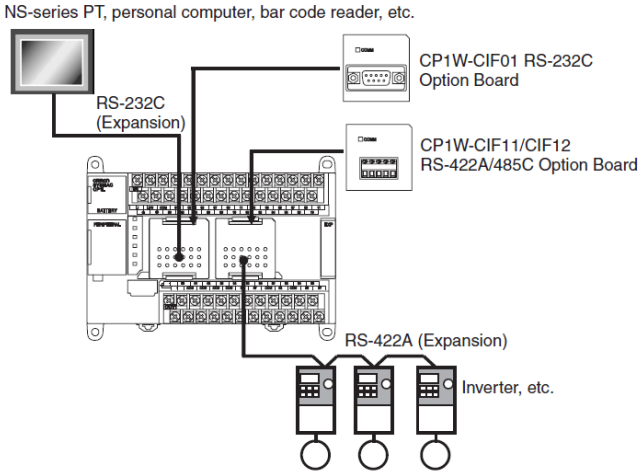

Ilustração 3.2.10: Exemplo de aumento das potencialidades de comunicação com recurso a *option boards* (28)

Protocolo *Ethernet/IP*

Esta tecnologia foi introduzida em 2001 e é atualmente gerida pela ODVA.

É compatível com protocolos de *Internet* típicos, tais como HTTP e FTP, e protocolos industriais comuns como OPC. Dado que é usada sobre *Ethernet*, não existe um limite para o número de dispositivos que podem ser interligados. Os sistemas *Ethernet/IP* podem ser configurados para operar numa relação mestre/escravo ou numa hierarquia de controlo distribuído com comunicações par-a-par. (29)

Um PLC CP1L pode ser configurado como escravo *Ethernet/IP*, recorrendo à instalação de um módulo adequado.

A série CP1L é capaz de realizar comunicações por *Modbus*. Com uma porta série, pode realizar trocas de dados por *Modbus* RTU, mas apenas como mestre (28). Contudo, com o protocolo *Modbus* TCP, é possível utilizar realizar trocas de dados como cliente ou como servidor ou de ambos os modos em simultâneo, usando duas *option boards Modbus* TCP. Por *Ethernet/IP*, apenas é possível a este controlador comunicar como escravo.

Ligar uma unidade *DeviceNet* CPM1A-DRT21 a funcionar como escravo permite a ambos os modelos funcionarem como escravos *DeviceNet*. Podem ser ligadas até 3 unidades ao PLC, estabelecendo assim 192 pontos de I/O entre o respetivo controlador e o mestre *DeviceNet* (28). Através de *DeviceNet*, é possível construir redes que suportem dispositivos de diversos fabricantes.

Por último, menciona-se que ambos os controladores podem participar numa rede de sensores e atuadores *CompoBus/S*, que não foi utilizada neste trabalho (28).

3.2.5.3. Potencialidades no Contexto da Dissertação

Os CPM1A apresentam-se bastante limitados em termos de comunicação com outros controladores, sendo compatíveis apenas com as redes *DeviceNet* e *CompoBus/S*. Com o protocolo série *Host Link*, da *Omron*, podem realizar comunicações com PCs ou HMIs compatíveis.

Os CP1L dispõem de *Option Boards* que lhes permitem comunicar sobre redes série ou *Ethernet*, utilizando os protocolos *Modbus*, variantes RTU e TCP, ou *Ethernet/IP*. Contudo, para *Modbus* RTU, apenas pode participar enquanto mestre da rede. No caso de *Ethernet/IP*, estes controladores apenas desempenham o papel de escravos. O *SoftPLC Codesys Control Win* V3 pode participar enquanto mestre desta ligação.

3.2.6. *Modicon* TSX Micro

Esta série de controladores foi fabricada pela *Modicon Telemecanique*, que foi comprada em 1989 pela AEG e que mais tarde passou a fazer parte da *Schneider Electric* (30) em 1994. Estes controladores foram introduzidos no final da década de 1980 e foi alvo de evolução até ao princípio deste século. Neste trabalho, utilizou-se um *Modicon* TSX-3722001 (Ilustração 3.2.11), ao qual estiveram ligados os dispositivos:

- Módulo de I/O TSX DMZ 28 DT;
- Carta PCMCIA RS485 TSX SCP 114;
- Módulo TSX ETZ 510;
- Placa de aquisição de dados *Advantech* USB-4750;
- Conversor RS-485/RS-232C *Westermo* MD-45 (31).

3.2.6.1. Características Funcionais

Os PLCs TSX *Micro* são programados através do *software* PL7 *Micro*. Esta aplicação utiliza as suas próprias versões das linguagens LD, ST e IL e inclui, mais notoriamente, a possibilidade de programar na linguagem gráfica definida pelo GRAFCET. Algumas das características funcionais do modelo TSX Micro usado estão expostas na Tabela 3.2.5.

Ilustração 3.2.11: *Modicon* TSX Micro com os módulo ETZ 510 e DMZ 28 DT, a placa de aquisição *Advantech* a carta PCMCIA e o conversor *Westermo*

Aos TSX Micro podem ser adicionados módulos de I/O e módulos de comunicação. Os módulos de comunicação permitem a estes controladores comunicar sobre diversas redes, incluindo as TCP/IP, redes de sensores e atuadores e redes CAN. Também podem adicionar interfaces série ao controlador, como é o caso das cartas PCMCIA (32).

Tabela 3.2.5: Características funcionais do TSX-3722001 e do módulo DMZ 28 DT (32)

Tensão de alimentação	100 a 240 V AC
Portas de comunicação:	
Integradas	2 portas de comunicação série
Adicionáveis por módulos	Diversas: série, *Ethernet*, CAN, entre outras
Módulo TSX DMZ 28 DT:	
Inputs discretas	16, a 24 VDC
Outputs discretas	12 saídas por relé

3.2.6.2. Capacidades de Comunicação:

Os PLCs TSX Micro possuem uma porta integrada RS485 para comunicar com outros controladores TSX 37. Estas ligações utilizam o protocolo mestre-escravo *Uni-Telway*, do qual a *Schneider Electric* é proprietária, ou *Modbus* RTU, que por sua vez é um protocolo aberto.

Os modelos TSX 37 21/22 podem utilizar uma carta de comunicação PCMCIA, que permite comunicar em *full-duplex* assíncrono através de redes: *CANopen*, *Fipio*, *Uni-Telway*, *Modbus*, *Modbus Plus* ou *Fipway* (32).

Para além disso, os TSX Micro podem ligar a uma rede *Ethernet* TCP/IP por um módulo externo e autónomo TSX ETZ 410 ou TSX ETZ 510 (Ilustração 3.2.12). Estes módulos são também usados para ligar os PLCs a um *modem* externo.

O módulo TSX ETZ 410 inclui um perfil de comunicação *Modbus* TCP/IP sobre *Ethernet* e também a opção de um servidor *Web* integrado. O módulo TSX ETZ 510 apresenta todas as funções incluídas no modelo 410 e adiciona mais funcionalidades ao servidor Web (32). O controlador comunica com o módulo através do protocolo série Uni-Telway, compatível com Modbus. Este protocolo pode convertido em *Modbus* TCP – e vice-versa – pelo módulo ETZ quando se pretende comunicar sobre ele.

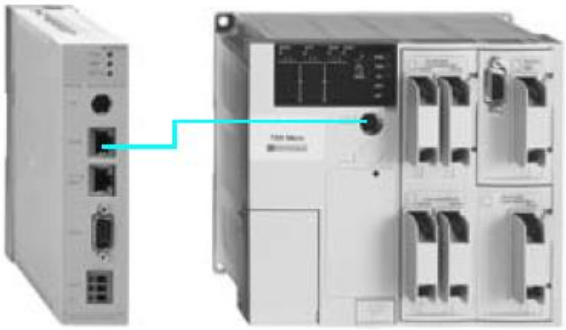

Ilustração 3.2.12: Controlador TSX *Micro* (à direita) ligado a um TSX ETZ (à esquerda) pelas interfaces série (32)

3.2.6.3. Potencialidades no Contexto da Dissertação

Sem recorrer à adição de módulos, o TSX *Micro* pode comunicar utilizando a sua porta série. Para este modo estão disponíveis os protocolos *Modbus* RTU, para dispositivos generalizados, ou PPP, para dispositivos da *Schneider Electric*.

O uso de módulos adequados permite ligar esta série de controladores a uma grande variedade de redes, desde as redes série às redes CAN e TCP/IP. Com a adição do módulo TSX ETZ 510, usado nesta dissertação, também estão disponíveis os protocolos *Modbus* TCP e *Uni-Telway*. Através da carta PCMCIA, o controlador pode realizar comunicação por *Modbus* RTU, por RS-485. O conversor *Westermo* permite utilizar o mesmo protocolo sobre RS-232.

A ligação do TSX *Micro* ao *Factory I/O* é possibilitada através de *Modbus* TCP ou por intermédio da placa *Advantech*.

3.2.7. HMI: *Omron* NB-Series

Neste trabalho foi possível utilizar um terminal programável, também conhecido como HMI, da *Omron*, modelo NB5Q-TW01B (Ilustração 3.2.13) (33).

Ilustração 3.2.13: Consola NB5Q-TW01B utilizada

Estas HMIs podem interligar-se com múltiplos PLCs de fabricantes diversos. Para isso, o modelo NB5Q-TW01B possui duas portas série, uma porta *Ethernet* e uma porta USB.

Podem comunicar com PLCs *Omron* através das portas série ou *Ethernet*. Podem também trocar dados com controladores dos fabricantes *Siemens*, *Mitsubishi*, *Schneider*, entre outros, recorrendo aos protocolos proprietários de cada dispositivo.

Por último, enquanto a consola HMI da Siemens apresentada anteriormente está limitada a comunicar com controladores Siemens por protocolos proprietários, as *Omron* NB-Series conseguem concretizar trocas de dados através do protocolo *Modbus*, variantes RTU, ASCII ou TCP, tomando o papel de mestre ou de escravo conforme o utilizador achar adequado.

3.3. Métodos de Comunicação entre Controladores e Cenários Virtuais

A Tabela 3.3.1 resume as opções de comunicação entre os controladores, a HMI *Omron*, e os ambientes virtuais *Factory I/O*. Destaca-se uma multitude de possibilidades para a realização das trocas de dados, observando-se uma predominância das variantes série e TCP do protocolo *Modbus*. As redes *Profibus*, *Profinet* e CAN, apesar de não virem a ser implementadas nesta dissertação, surgem algo frequentemente enquanto opções realizáveis.

Tabela 3.3.1: Comunicações realizáveis entre cenários e dispositivos

	Codesys Control Win V3	*Codesys SP PLCWinNT* V2.4	*Siemens* S7-1200	*Siemens* S7-200	*Moeller* EC4-222	*Omron Sysmac* CP1L	*Modicon* TSX *Micro*
Codesys Control Win V3		*Modbus* (RTU, TCP), Variáveis de rede (UDP/IP)	*Profinet, Profibus, Modbus* (RTU, TCP)	*Profinet, Profibus, Modbus* (RTU, TCP)	*Profibus*, CAN, Variáveis de rede (CAN), *Modbus* (RTU, TCP)	*DeviceNet, Modbus* (RTU, TCP), *Ethernet/IP*	CAN, *Modbus* (RTU, TCP)
Codesys SP PLCWinNT V2.4			*Modbus* (RTU, TCP)	*Modbus* (RTU, TCP)	*Modbus* (RTU, TCP)	*Modbus* (RTU, TCP)	*Modbus* (RTU, TCP)
Siemens S7-1200				*Profinet, Profibus, Modbus* (RTU, TCP), PtP	*Profibus, Modbus* (RTU, TCP)	*Modbus* (RTU, TCP)	*Modbus* (RTU, TCP)
Siemens S7-200					*Profibus, Modbus* (RTU, TCP)	*Modbus* (RTU, TCP)	*Modbus* (RTU, TCP)
Moeller EC4-222						*Modbus* (RTU, TCP)	CAN, *Modbus* (RTU, TCP)
Omron Sysmac CP1L							*DeviceNet, Modbus* (RTU, TCP)
Omron NB5Q	*Modbus* (RTU, TCP)	*Modbus* (RTU, TCP)	*Modbus* (RTU, TCP), Protocolo proprietário sobre TCP da *Siemens*	*Modbus* (RTU, TCP), Protocolo proprietário da *Siemens* (série ou TCP)	*Modbus* (RTU, TCP)	*Modbus* (RTU, TCP), Protocolo proprietário *Host Link* (série ou TCP)	*Modbus* (RTU, TCP)
Factory I/O	*Modbus* TCP, Servidor OPC	*Modbus* TCP, Servidor OPC	*Modbus* TCP, *driver* S7-1200	*Modbus* TCP, *driver* S7-200	*Modbus* TCP , placa de I/O *Advantech* 4750	*Modbus* TCP , placa de I/O *Advantech* 4750	*Modbus* TCP, placa de I/O *Advantech* 4750

3.4.Conclusões do Capítulo

O capítulo apresentou as várias tecnologias a utilizar no desenvolvimento da dissertação.

O *Factory I/O* permite inúmeras possibilidades para a elaboração de ambientes fabris virtuais. Dentre os atuadores utilizáveis, optou-se pela utilização de equipamentos de transporte de paletes e objetos, como forma de construir sistemas de eventos discretos automatizáveis. Os dispositivos de operação e sinalização serão utilizados como uma interface com o utilizador.

O leque de PLCs disponíveis foi apresentado, concluindo-se que existe uma diversidade de possibilidades de comunicação, recorrendo a redes e protocolos com diferentes suportes físicos de diversas eras tecnológicas. Verifica-se que o protocolo *Modbus* é a opção de comunicação mais comum entre os controladores lógicos, a maioria dos quais consegue utilizar as variantes RTU e TCP, tanto como mestre, tanto como escravo.

Os *SoftPLCs Codesys* constituem uma solução de controlo interessante. A realização de trocas de dados entre os dois *SoftPLCs* por variáveis de rede sobre UDP afigura-se um exercício pertinente. Também constitui uma prática interessante a efetuação de comunicação série sem protocolo entre os dois controladores Siemens S7, através dos modos PtP e *Freeport.*

Relativamente ao CP1L, utilizando as *option boards* adequadas, é possível testar várias soluções de comunicação entre ele e os restantes controladores. Estas soluções incluem o protocolo *Modbus* e, mais aliciante, comunicação *Ethernet/IP* com o *Codesys Control Win* V3.Relativamente ao *Modicon* TSX *Micro*, a solução de comunicação mais interessante é *Modbus* TCP, visto que para isso é necessária a configuração de um dispositivo adicional, o TSX ETZ 510, que pratica a conversão entre os protocolos *Uni-Telway* e *Modbus* TCP.

A oferta tecnológica é complementada pelas HMIs, que permitirão a eventual inserção de controlo de supervisão, recorrendo a comunicação vertical. A HMI *Omron* NB5Q é bastante flexível, podendo comunicar com todos os PLCs escolhidos sobre diversos meios.

Os próximos capítulos relatam os desenvolvimentos que foram explorados na realização desta dissertação. Estes desenvolvimentos consistem na combinação destas tecnologias de forma a criar sistemas automatizados constituídos por controladores e cenários controláveis.

Capítulo 4 – Conceção de Modelos Fundamentais de Produção Flexível

Este capítulo começa com uma breve introdução aos sistemas de eventos discretos e sistemas de fabricação responsivos, classes em que se inserem os cenários virtuais de interesse para esta dissertação. A essa, segue-se a apresentação das ferramentas de modelação a utilizar na especificação e conceção dos casos de estudo e na implementação dos controladores.

Os modelos de implementação dos controladores lógicos assentam essencialmente em diagramas definidos pelas normas UML e *SysML*. Esta opção é justificada pela excelente versatilidade, cobertura e atualidades destas ferramentas na modelação de padrões e técnicas de programação modernas utilizadas em sistemas sequenciais distribuídos, caraterística intrínseca aos cenários tratados nesta dissertação.

O resto deste capítulo iniciará uma abordagem prévia da implementação do controlo dos cenários virtuais. Para isso, definir-se-ão arquiteturas de controlo lógico a aplicar aos casos de estudo. Esta exposição, auxiliada pelas normas de modelação apresentadas, divide-se em dois tipos.

O primeiro consiste na elaboração de arquiteturas de controlo direto dos componentes do *Factory I/O*, organizados em subconjuntos de atuadores e sensores *Factory I/O* que desempenharão funções predefinidas, maioritariamente focadas na movimentação de itens. É esclarecido qual o comportamento dinâmico pretendido e a lógica do controlo direto aplicado a cada um deles.

O segundo tipo trata-se de um dos desafios centrais à implementação do controlo distribuído, ou cooperativo (34), caracterizado pela utilização de múltiplos controladores em cada cenário, cada um responsável pelo controlo de um ou mais subsistemas. Aqui serão especificadas as soluções controlo global do ambiente virtual e dos controladores, sendo explicado o sentido das interações hierárquicas entre o cenário, os controladores e as tarefas programadas.

Com esta apresentação ficarão estabelecidas as bases que permitirão compreender as linhas dos desenvolvimentos relatados no Capítulo 5.

4.1. Sistemas de Automação Discreta e Responsiva

Ao enfrentar problemas de automação, existe uma necessidade de categorizar os sistemas de fabricação automatizados de acordo com certos critérios, como, por exemplo, se se tratam de sistemas discretos ou de sistemas contínuos, ou se se classificam como casos de automação rígida ou de automação flexível. Na conceção de um sistema automatizado, estes critérios devem ser definidos de acordo com vários parâmetros, como o objetivo do sistema, os seus requisitos funcionais, tempo de vida, etc.

Relevantes a esta dissertação, definem-se aqui os conceitos de sistemas de eventos discretos e de sistemas de fabricação responsivos, classes em que se incluem os cenários virtuais a desenvolver.

Relativamente aos sistemas de eventos discretos (*discrete event systems* ou DES), estes definem-se como sendo um sistema cujo comportamento pode ser descrito por um conjunto discreto de estados, estados estes cujas transições são observadas em instantes discretos no tempo. Essas transições são associadas a eventos. (35)

Um evento é uma ocorrência que se observa de forma discreta e espontânea e que é observada num curto intervalo de tempo. Aquilo que se identifica como sendo um evento pode variar desde ocorrências simples e genéricas como o premir de um botão num painel de operador ou algo mais específico como o fim do transporte de um objeto ou a deteção de uma avaria.

O estudo de sistemas de eventos discretos invoca uma multitude de questões pertinentes tais como:

- Controlo e requisitos de tempo real;
- Comportamentos baseados em eventos;
- Desempenhos síncronos e assíncronos;
- Sequenciação, concorrência ou conflito de operações e de tarefas;
- Coordenação e comunicação entre controladores;
- Indeterminismo;
- Ocorrência de impasses (*deadlock*) (36) (37).

Estes tópicos, em particular a questão dos requisitos de tempo real, conduzem à discussão de sistemas de fabricação responsivos, sob a qual também se inserirão os casos de estudo desta dissertação.

As empresas e organizações fabris devem ser capazes de reagir e responder rapidamente a repetidas alterações de mercado de modo a prever e continuamente melhorar a produtividade dos seus sistemas de fabricação (38).

Tendo isto em conta, os sistemas de fabricação responsivos, também referidos como sistemas de fabricação ágeis, são definidos pela sua capacidade de modificar o seu funcionamento de acordo com as repetidas alterações de mercado numa maneira pouco custosa e, simultaneamente, prosperar perante estas incertezas (38) (39).

Por outras palavras, um sistema de fabricação deve ser reconfigurável, de modo a permitir a uma organização ser responsiva perante mercados incertos e inconstantes e ser capaz de operar eficiente e produtivamente. Por sua vez, isto requer metodologias fiáveis para modelar, analisar e conceber estes sistemas de fabricação responsivos. Nomeadamente, é importante estabelecer as relações de causa-efeito entre as variáveis de fabrico, tais como a cadência de produção, tempo de produção, prazos de entrega, etc. (38).

Utilizando os cenários virtuais, é possível simular soluções de conceção de sistemas automatizados capazes de satisfazer os requisitos inerentes na fabricação ágil ou responsiva. É, portanto, interessante para este trabalho adotar modelos de controlo que se adaptam relativamente bem a alterações dos requisitos ou do equipamento.

4.2. Normas e Princípios de Modelação UML e *SysML*

Um modelo é uma representação de algum sistema ou objeto dentro de um determinado meio. Um modelo capta os aspetos importantes daquilo que modela sob um certo ponto de vista, simplificando ou omitindo o acessório (40). Em engenharia, assim como noutras áreas, utilizam-se modelos para determinados propósitos, nomeadamente:

- Apresentar e listar com detalhe os requisitos para que todas as partes interessadas possam compreendê-los;
- Auxiliar o projeto e conceção de um sistema;
- Apurar decisões de *design*;
- Organizar, encontrar, filtrar, recolher, examinar, e editar a informação sobre sistemas grandes e complexos (40).

No decorrer deste trabalho, recorreu-se a várias técnicas e linguagens de modelação para ilustrar os casos de estudo elaborados, tratando-se estes de cenários virtuais manipulados por controladores lógicos. Deste modo, pretende-se facilitar a apresentação e análise destes sistemas sob vários níveis, mais especificamente aos níveis da organização física, da programação, da lógica comportamental e das comunicações.

4.2.1. UML - *Unified Modeling Language*

UML (41) é uma linguagem genérica de modelação visual usada para especificar, visualizar, construir e documentar os componentes de um sistema. Apresenta aspetos decisivos desses sistemas e é usada para compreender, projetar, configurar, monitorizar e controlar informação sobre eles (40).

Foi desenvolvida na década de 1990 numa tentativa de simplificação e consolidação do grande número de métodos de desenvolvimento de sistemas que foram surgindo na segunda metade do século XX (40). É atualmente gerida pelo OMG (*Object Management Group*).

UML compreende uma vasta diversidade de tipos de diagramas para representar os inúmeros aspetos de um sistema, tais como a sua estrutura organizacional, as interações entre os seus componentes ou o seu comportamento dinâmico. É utilizada nas áreas de informação, finanças, telecomunicações, transporte, ciência e investigação, entre outras (42). É vastamente utilizada na modelação de sistemas de *software* podendo, no entanto, ser usada noutro tipo de sistemas.

4.2.2. *SysML - Systems Modeling Language*

A também definida e publicada pelo OMG, *SysML* (43) é uma linguagem baseada em UML, reutilizando parte das suas especificações e adicionando outras funcionalidades (44).

Enquanto UML apresenta-se como uma linguagem mais generalizada, *SysML* foi concebida para oferecer estruturas simples mas eficazes para modelar uma variedade de problemas de engenharia. É particularmente eficaz na especificação de requisitos, estrutura, comportamentos, restrições e propriedades de sistemas (44).

4.2.3. Classes de Modelos Adotadas

Dentre estas duas normas, procedeu-se a uma seleção de classes de diagramas cujas propriedades tornam-nas adequadas à representação dos casos de estudo a construir. Esta seleção teve em conta a natureza desses cenários enquanto sistemas de eventos discretos, ou sistemas responsivos, automatizados.

Portanto, como já foi dito, é pretendido modelar os casos de estudo a nível da estrutura física do cenário, da lógica comportamental especificada, da programação que materializa esse comportamento e da implementação das comunicações entre dispositivos.

Entre os tipos de diagramas definidos pelas especificações UML ou *SysML*, recorreu-se aos diagramas de atividade, de estados, de sequência e de componentes para a realização deste trabalho. Entre os diagramas referidos, os de componentes são definidos apenas por UML e não por *SysML*, sendo os restantes especificados por ambas as normas.

4.2.3.1. Diagramas de Atividade e Diagramas de Estados

Estes dois tipos de diagramas, em conjunto com os diagramas de sequência, servem para exprimir o comportamento dinâmico de um sistema.

Um diagrama de atividade (*activity diagram*) (41) (43) representa as várias ações de um sistema, permitindo assim exprimir comportamentos complexos. Podem também demonstrar o fluxo de objetos (matéria, energia ou dados) através de uma atividade, assim como a lógica por detrás dela.

São constituídos por atividades, objetos, que são tratados ou processados pelo sistema, e transições, que representam os eventos que provocam a mudança de atividade. Através da partição de atividades, é possível representar vários sistemas num só diagrama, demonstrando assim a dinâmica de um sistema com uma estrutura de alto nível.

Os diagramas de atividade destacam-se por serem bons a exprimir o fluxo de objetos ao longo da evolução do sistema, assim como a lógica por detrás do seu desempenho. A Ilustração 4.2.1 mostra um simples diagrama da transferência de uma palete entre dois tapetes de rolos numa linha de produção.

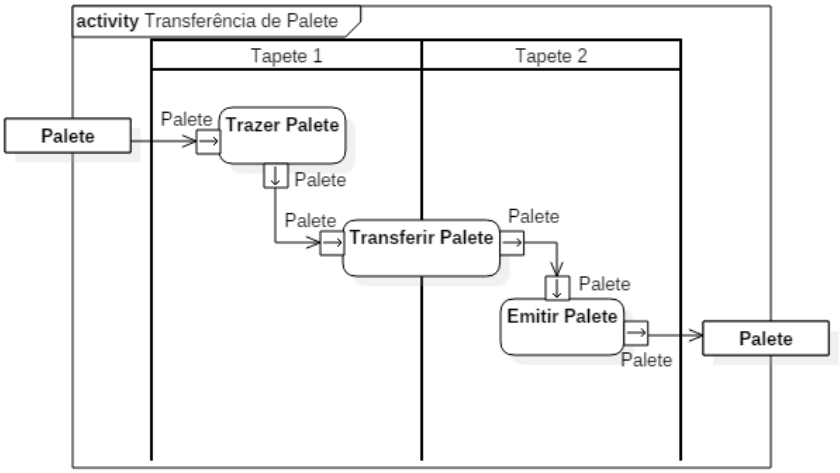

Ilustração 4.2.1: Diagrama de atividade da transferência duma palete entre dois tapetes transportadores em série

Estes diagramas possuem, como desvantagem, uma certa ambiguidade, na medida em que nem sempre exprimem, de modo adequado, quais as estruturas que executam cada ação. Por este motivo, os diagramas de atividade não constituem um bom utensílio para criar uma representação detalhada de um sistema complexo (44).

Por outro lado, os diagramas de estados (*state machine diagrams* ou *statechart diagrams*) focam-se na forma de como o estado dum sistema ou de parte dele se altera em resposta a eventos ao longo do tempo.

O funcionamento de um sistema é representado por um conjunto de estados. Um estado pode ser simples ou ser composto por subestados. As transições representam os eventos e condições que provocam a mudança de estado. Dentro de um estado podem estar representadas ações, sendo especificadas as condições que as provocam, assim como a sequência em que elas ocorrem (44).

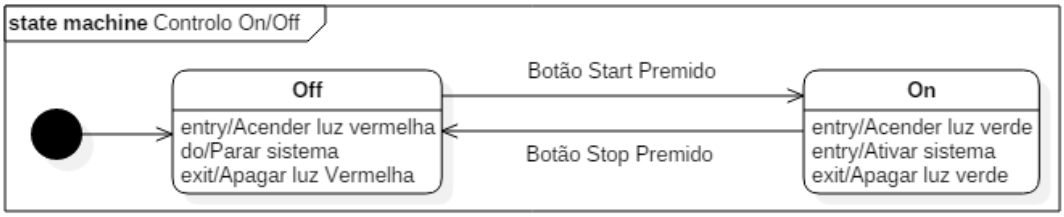

Ilustração 4.2.2: Diagrama de estados do controlo On/Off simples de um sistema

Enquanto os diagramas de atividade focam-se na ordem segundo a qual um sistema de desenvolve, assim como no fluxo de objetos, os diagramas de estados atribuem maior destaque às ações decorrentes e especificam melhor quais as estruturas envolvidas nessas ações. O diagrama de estados da Ilustração 4.2.2 apresenta o caso de um sistema que pode ser ligado ou desligado através da comutação de dois botões, *Start* e *Stop*. A cada instante, o estado atual desse sistema é indicado por uma de duas luzes, verde e vermelha.

Utilizou-se os diagramas de atividade e de estados para modelar aspetos comportamentais e de implementação de baixo nível. Estes aspetos são, para os diagramas de atividade, o desempenho de alguns componentes ou detalhes de determinadas atividades e, para os diagramas de estados, as ações de controlo lógico implementadas.

4.2.3.2. Diagramas de Sequência

Os diagramas de sequência (*sequence diagrams*) (41) (43) constituem outra classe de diagramas que exprimem informação acerca do comportamento dinâmico de um sistema. No diagrama são representados os participantes de um sistema que interagem entre si através de mensagens (44). As ações que ocorrem internamente a um participante são representadas por auto-mensagens.

Este tipo de modelo especifica a ordem segundo a qual as ações e interações decorrem. Também apresenta quais os componentes ou estruturas que evocam cada ação (44). Isto é exemplificado no diagrama da Ilustração 4.2.3, onde um controlador lógico ordena a ativação de um motor de um sistema controlado, ação essa que é executado pelo último dos dois participantes.

Neste trabalho, foram utilizados diagramas de sequência para representar as trocas de informação entre controladores e a ordem segundo a qual se realizam as mensagens, que varia de acordo com o tipo de comunicações implementadas. Também foram usados para a representação de determinadas interações específicas aos casos de estudo.

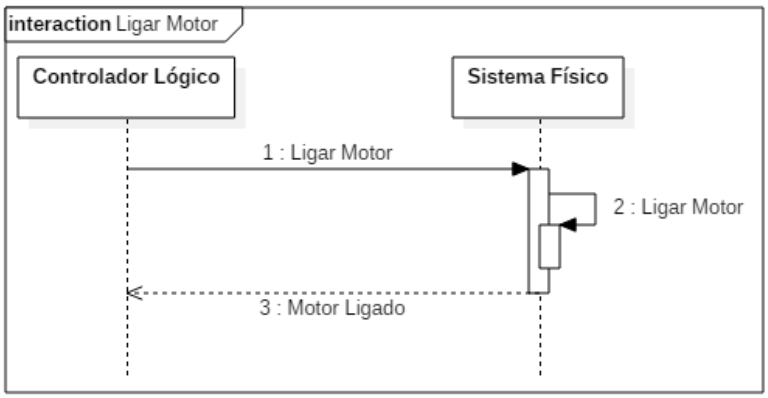

Ilustração 4.2.3: Diagrama de sequência da troca de mensagens entre um controlador lógico e um sistema físico

4.2.3.3. Diagramas de Componentes

Os diagramas de componentes (*component diagrams*) (41) mostram a definição, estrutura interna e dependências entre os componentes de um sistema (40). Entres os diagramas até aqui introduzidos, estes tratam-se dos únicos que são definidos apenas pela norma UML, e não por *SysML*.

Define-se componente como sendo uma parte modular de um sistema cuja implementação requer determinadas interfaces externas. Essas interfaces podem ser implementadas pelos próprios componentes ou externamente fornecidas.

Os componentes podem também possuir artefactos. Estes constituem elementos de *software* que fazem parte da implementação como, por exemplo, código, *scripts*, etc. Os diagramas mostram as dependências entre componentes e artefactos (40).

Neste trabalho, adotou-se os diagramas de componentes para ilustrar as ligações e interfaces realizadas entre os vários controladores usados e os cenários virtuais *Factory I/O*. O diagrama da Ilustração 4.2.4 mostra um exemplo duma ligação entre dois controladores realizada através do protocolo *Modbus* TCP.

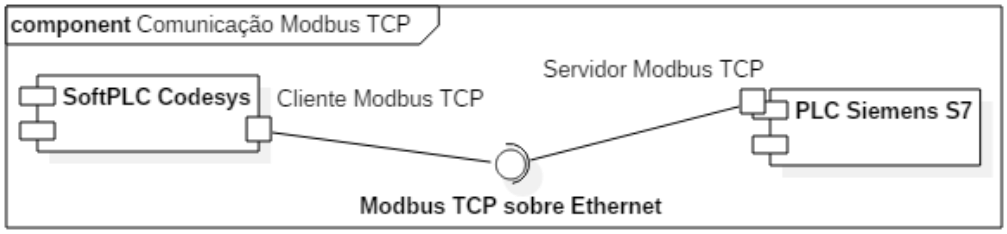

Ilustração 4.2.4: Diagrama de componentes da realização de uma interface por *Modbus* TCP entre dois controladores

As normas UML e *SysML* são amplas e flexíveis, tendo sido apenas apresentados os aspetos essenciais ao trabalho a desenvolver. Com esta exposição, ficam apresentadas as ferramentas de modelação a utilizar na especificação dos casos de estudo no resto desta dissertação.

4.3. Modelação e Controlo Direto de Atividades Fundamentais

Será aplicada uma arquitetura de controlo distribuído aos ambientes virtuais, sendo feita uma divisão dos mesmos em subsistemas, alvos do controlo direto dum controlador a eles atribuídos. Estes subsistemas serão compostos por elementos físicos do cenário, tais como sensores, atuadores, ou dispositivos de interface humana, que constituirão o periférico do respetivo controlador.

Previamente à apresentação dos cenários e do seu controlo, serão expostos alguns destes subsistemas, que consistem em conjuntos de sensores e atuadores posicionados de modo a, juntamente com o controlo implementado, exercerem um dado aspeto da atividade do cenário. Na construção dos cenários, estes conjuntos serão unidos como blocos físicos funcionais, permitindo um desenvolvimento sistemático e coerente dos ambientes virtuais.

O diagrama de contexto (Ilustração 4.3.1) ilustra de forma simples a interação entre os controladores e dos sistemas de controlo direto a incluir nos cenários virtuais.

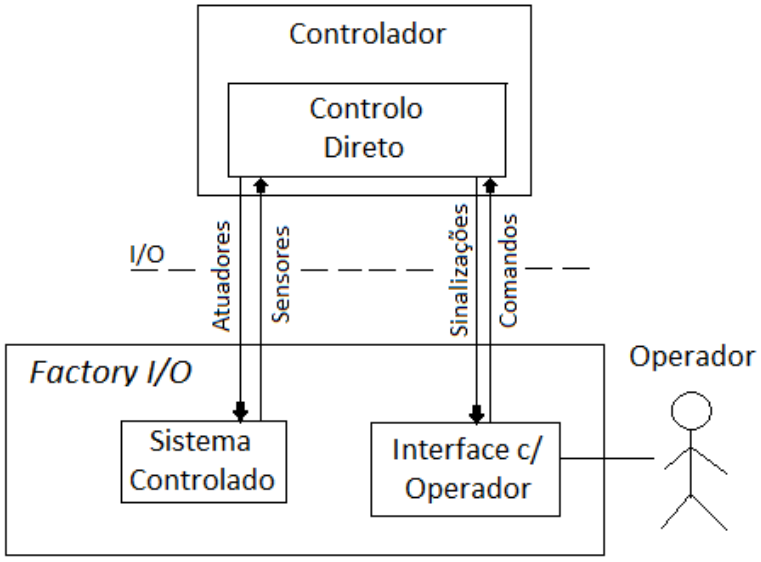

Ilustração 4.3.1: Controlo direto dos dispositivos *Factory I/O*

4.3.1. Tapetes Transportadores Unidirecionais

Os tapetes transportadores serão frequentemente utilizados nos cenários virtuais. Por este motivo, surge a necessidade de definir modelos de controlo para utilizá-los de forma recorrente, eficaz e consistente, enquanto sistemas controlados. Como tal, apresentam-se três configurações físicas para os tapetes transportadores a utilizar. O comportamento esperado, assim como a solução de controlo implementada são especificados pelo modelo apresentado para cada caso.

4.3.1.1. Tapete alimentador

Sobre este tapete (Ilustração 4.3.2) encontra-se, num dos seus extremos, um emissor e, no outro extremo, um sensor de posição. Este conjunto tem como função receber itens transportáveis a partir do emissor e descolá-los para algum outro componente ou atuador que se encontre a jusante.

Inicialmente, o tapete encontra-se inativo, apenas ativando quando recebe um sinal *OrdemStart*. O conjunto volta ao estado inativo se receber um outro sinal *OrdemStop*. No estado ativo, o conjunto recebe itens transportáveis e desloca-os até ao sensor, situação na qual encontra-se parado. Nesse ponto, é enviado ao controlador um sinal *SensorOn* que indica a presença de um objeto preparado para ser deslocado para os atuadores a jusante.

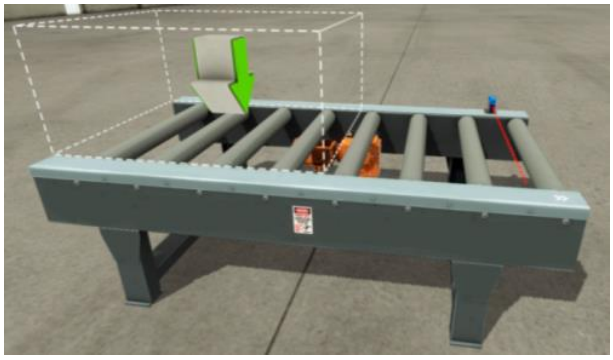

Ilustração 4.3.2: Tapete alimentador com emissor e sensor de posição

Só após receber um sinal *OrdemMover* da parte do controlador, o tapete inicia a transferência do item para o ponto seguinte do cenário. Finda esta fase, o conjunto repete o procedimento, sendo o tapete atuado, à espera de outro item. Estes comportamentos são representados pelo diagrama de estados da Ilustração 4.3.3.

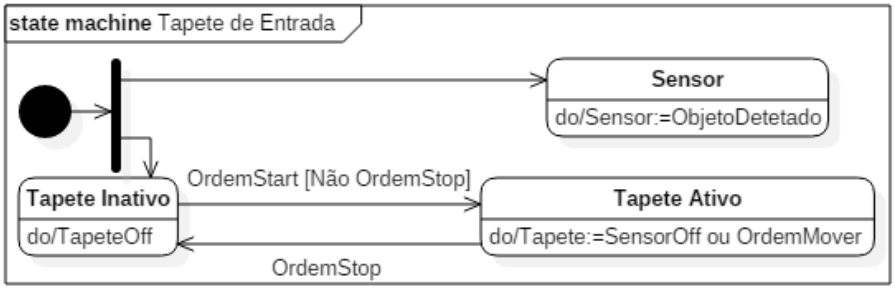

Ilustração 4.3.3: Diagrama de estados de um tapete de entrada

A estes tapetes podem ser adicionados outros sensores com o propósito de fazer a distinção (altura de uma caixa ou cor de uma peça) dos itens a transportar.

4.3.1.2. Tapete de saída

Em contraste com os tapetes alimentadores, os tapetes de saída (Ilustração 4.3.4) constituem um ponto terminal do ambiente virtual. Possuem um sensor à entrada e um eliminador no outro extremo.

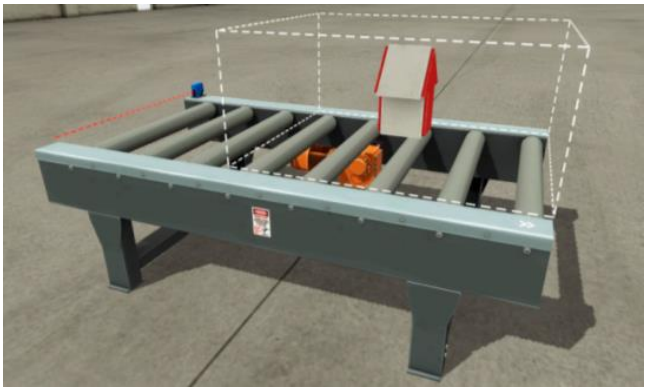

Ilustração 4.3.4: Tapete de saída com sensor de posição e eliminador

Foi adotada uma solução simples para o controlo (Ilustração 4.3.5) de um destes tapetes: ao receber um objeto, o que é assinalado pela atuação do sensor, inicia-se a sua movimentação. A atuação do tapete termina ao fim de algum tempo após o objeto deixar de ser detetado pelo sensor. Caso seja ordenada a paragem súbita do ambiente, o temporizador é também desativado e apenas reativado quando a operação é retomada.

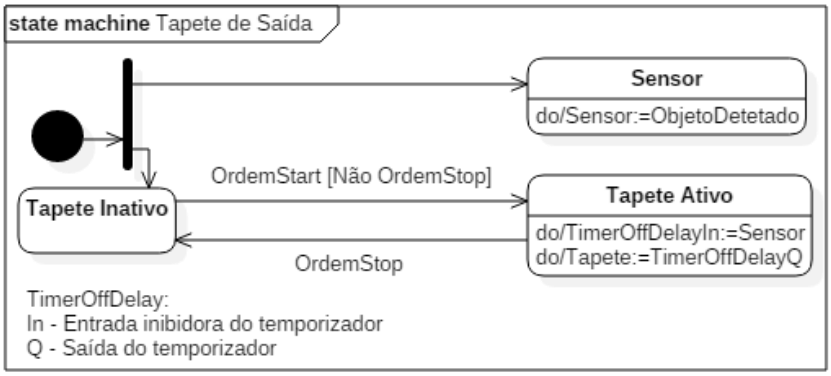

Ilustração 4.3.5: Diagrama de estados de um tapete de saída

4.3.1.3. Tapete intermédio

Os tapetes intermédios (Ilustração 4.3.6) possuem um sensor em cada extremo. O seu propósito é receber itens originados de outros pontos do cenário e reencaminhá-los para outros locais dentro do mesmo ambiente virtual.

É importante impedir a sobreocupação destes tapetes por parte dos itens, ou alguma disposição de objetos que possa provocar alguma colisão ou obstrução na instalação. Por estes motivos, estes elementos devem ser limitados, por parte do programa, a apenas transportarem um objeto de cada vez. Gerir o estado de ocupação – ou de não ocupação – dos tapetes é função dos controladores.

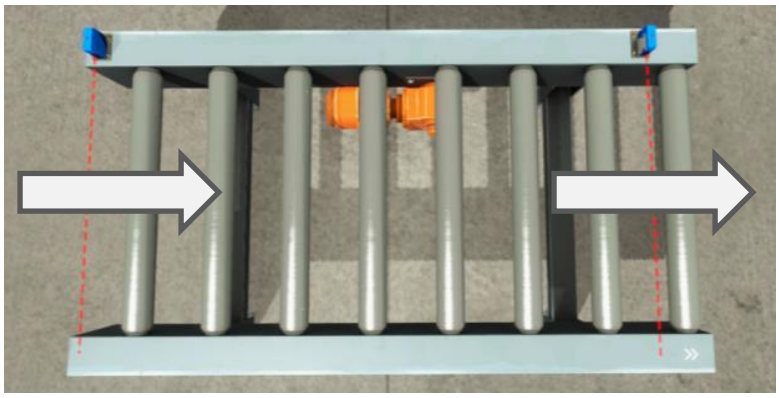

Ilustração 4.3.6: Tapete intermédio com dois sensores

O diagrama de estados da Ilustração 4.3.7 apresenta o comportamento de um tapete intermédio.

Para além dos modos ativo e inativo, definidos para os conjuntos anteriores, estes apresentam os estados vazio e ocupado. O conjunto encontra-se inicialmente no estado vazio, passando a estar ocupado quando é detetada uma transição ascendente do sensor de entrada. Regressa ao estado vazio quando é detetada uma transição descendente no sensor de saída.

No estado ocupado, o tapete transporta o item até ao sensor de saída, esperando pelo sinal *OrdemMover* para emiti-lo ao atuador ou conjunto a jusante. A atuação do tapete apenas decorre se o tapete se encontrar no modo ativo.

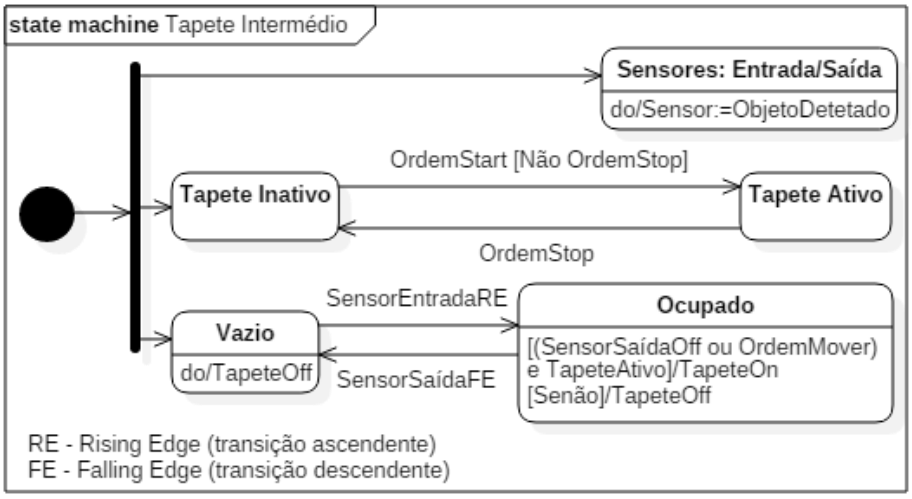

Ilustração 4.3.7: Diagrama de estados de um tapete intermédio

4.3.2. Mesa Rotativa

Como foi mencionado no capítulo anterior, as mesas rotativas (Ilustração 4.3.8) incluem um tapete transportador bidirecional de rolos, cujo comportamento é definido pelo diagrama da Ilustração 4.3.9. Estes tapetes podem deslocar peças num de dois sentidos, indicado pelo valor de duas variáveis binárias inibidoras. A cada variável corresponde um sentido; se ambas estiverem ativas, o tapete fica parado.

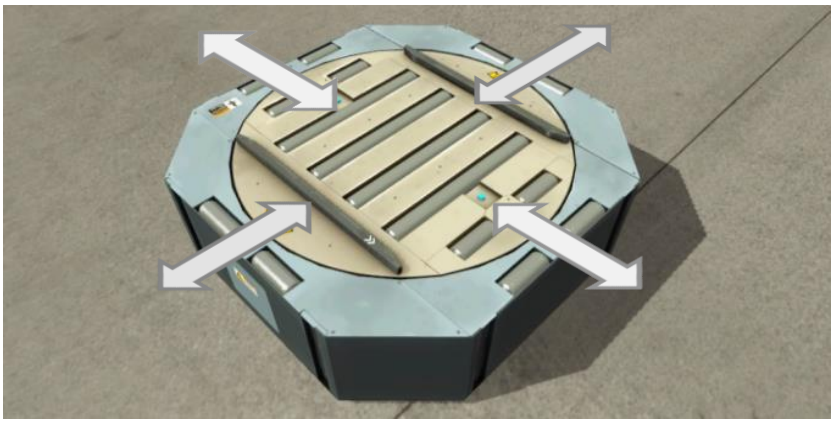

Ilustração 4.3.8: Mesa rotativa

Esta definição representada pelo diagrama aplica-se a outros atuadores digitais capazes de exercer uma ação em dois sentidos opostos, tais as mesas de transferência.

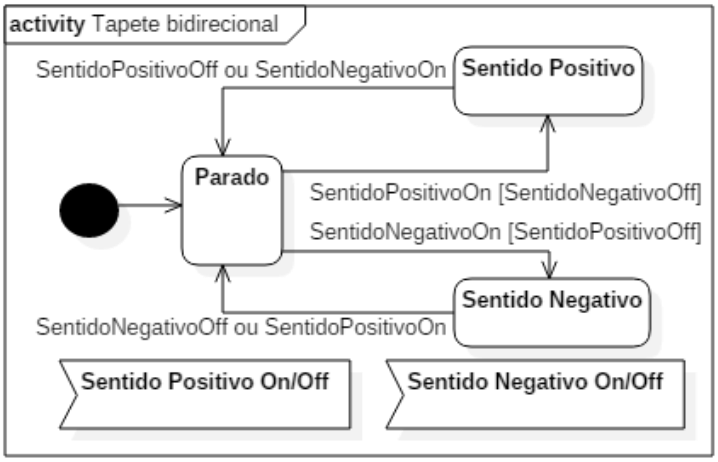

Ilustração 4.3.9: Diagrama de atividade de um tapete transportador bidirecional

O diagrama de atividade da Ilustração 4.3.10 ilustra a dinâmica da orientação das mesas rotativas. Quando um dos sinais de rotação, em qualquer sentido, é ativado, a mesa roda até atingir a posição limite. Similarmente ao respetivo tapete de rolos, se a mesa receber ordem para rodar em ambos os sentidos em simultâneo, a mesa para na sua posição atual.

Encontrando-se numa posição limite, a mesa emite um sinal digital a indicar em qual das posições se encontra. Quando se encontra numa posição intermédia, não emite sinal. Os sensores de posição funcionam identicamente aos sensores anteriormente definidos.

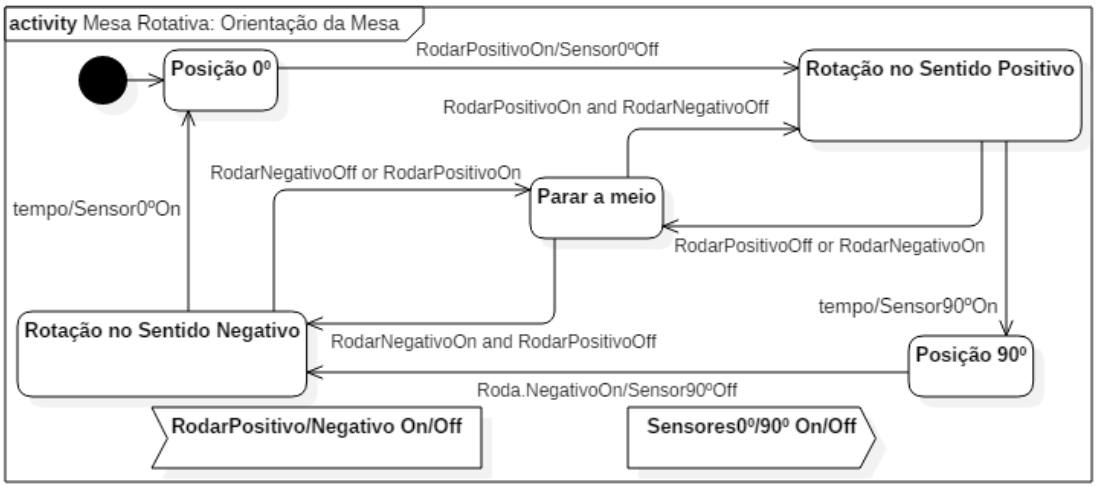

Ilustração 4.3.10: Diagrama de estados duma mesa rotativa

4.3.3. Interseção com Elementos Transportadores

Este conjunto (Ilustração 4.3.11) é constituído por atuadores transportadores, tipicamente tapetes, anexados em torno de um elemento central, responsável por receber e reencaminhar itens manipuláveis. A inclusão destes subconjuntos permitirá a prática do controlo e gestão de recursos comuns a tarefas distintas duma mesma atividade.

Os atuadores anexos podem ser tapetes transportadores com as configurações definidas anteriormente ou podem eles próprios ser elementos centrais duma outra intersecção. O elemento central deve ser capaz de mover objetos em duas ou mais direções, pelo que esta função recaiu nas mesas rotativas, de transferência ou nos separadores a rodízios. Nos extremos dos atuadores deverão existir sensores devidamente colocados, de modo que se monitorize a presença dos itens transportáveis, assim como a sua transferência entre transportadores.

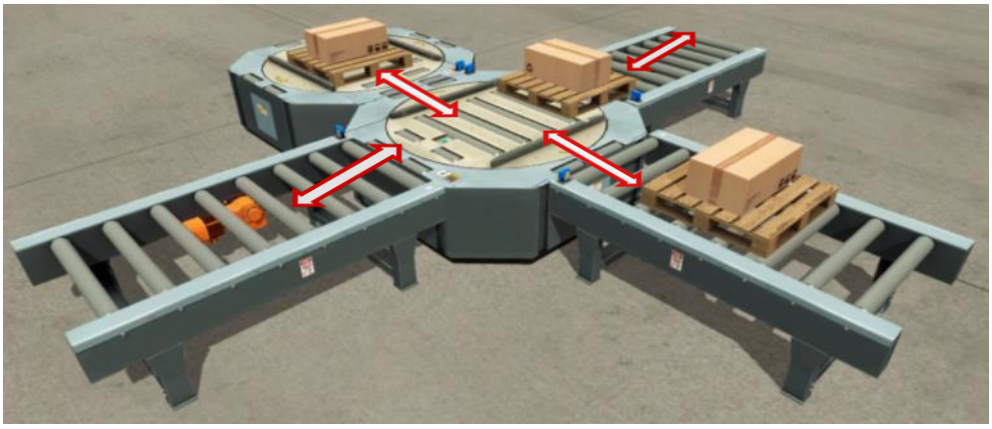

Ilustração 4.3.11: Cruzamento de uma mesa rotativa central, com três tapetes e outra mesa rotativa

Esta configuração permite que os elementos anexados funcionem como entradas, donde originam os objetos a mover, saídas, para estes são reencaminhados ou ambas, de acordo com as especificações do cenário.

O diagrama de estados da Ilustração 4.3.12 apresenta a lógica funcional dum destes cruzamentos. Inicialmente, encontra-se desocupado e espera pela presença de um objeto num dos elementos transportadores à sua entrada. Confirmada tal situação, é necessário decidir se o elemento central aceita o item ou não e, na situação de várias das suas entradas estarem ocupadas, qual delas terá prioridade sobre as restantes. Estas questões são decididas pelo controlador e determinadas pela lógica da aplicação.

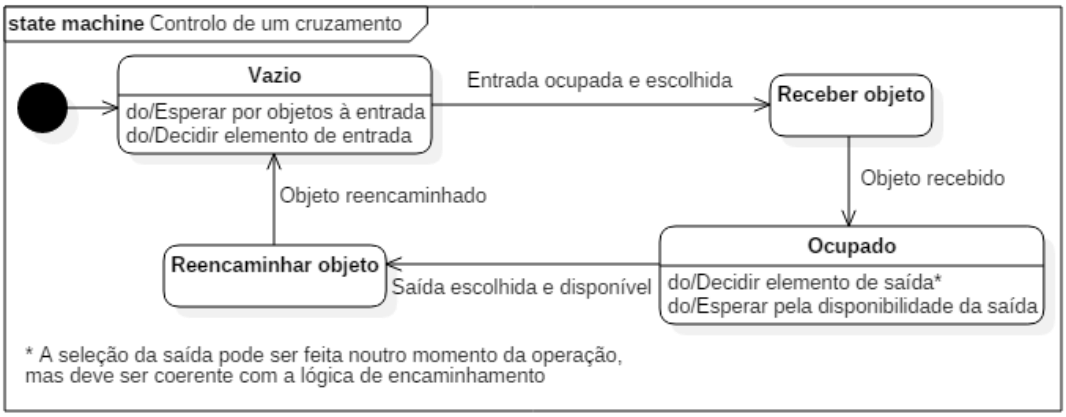

Ilustração 4.3.12: Diagrama de estados do controlo de um cruzamento

Iniciada e eventualmente concluída a receção do objeto, surgem questões idênticas relativamente ao destino do item a transportar. Ainda mais, é necessária a disponibilidade do atuador de destino para que este possa receber o objeto. À falta disso, pode dar-se uma obstrução da instalação na qual o elemento central não é capaz de transferir o item que ele contém. Normalmente, pelo bem da eficiência e da evasão de situações obstrutivas, estas questões são resolvidas previamente à receção do objeto. No entanto, tratando-se isto duma tarefa do controlador e não dos componentes físicos, o diagrama apresenta esta função na fase intermédia entre a receção e o reencaminhamento do objeto.

Finda a transferência do item, o elemento central volta a estar disponível, iniciando-se um novo ciclo.

4.3.4. Manipulador *Pick and Place* de 2 Eixos

O diagrama de atividade da Ilustração 4.3.13 apresenta os vários componentes de um manipulador *Pick and Place* de dois eixos. Os dois eixos são controladores independentemente um do outro. O controlo de movimentação é feito por um controlador interno ao manipulador e dá-se em resposta ao valor de referência transmitido pelo controlador lógico. O valor da posição atual é transmitido dentro da mesma gama que a referência.

A ventosa pneumática e o sensor de posição tratam-se de um atuador e de um sensor digitais, respetivamente.

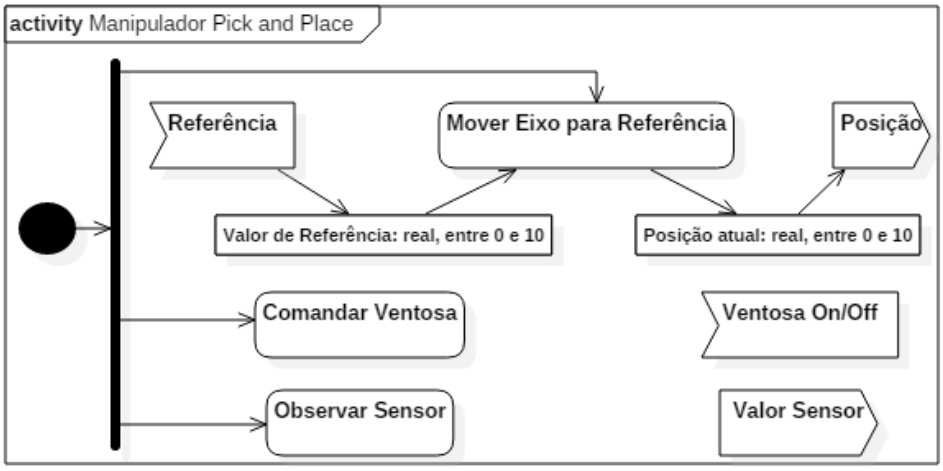

Ilustração 4.3.13: Diagrama de atividade de um manipulador *Pick and Place* de dois eixos

4.3.5. Transferência por Manipulador *Pick and Place* de 2 Eixos

Este conjunto de dispositivos (Ilustração 4.3.14) é constituído por um manipulador Pick and Place que exerce a transferência duma peça entre dois pontos, normalmente tapetes. Em ambos os pontos estão posicionados sensores para detetar as peças. Entre os dois tapetes existe uma barreira que impede o movimento horizontal do manipulador sem que o eixo vertical deste esteja completamente retraído.

O diagrama de estados da figura representa um atuador de dois eixos que permite o transporte de peças entre dois - ou mais - pontos. Na situação de repouso, este atuador encontra-se parado e retraído segundo o eixo vertical, ou seja, na sua posição mais alta. Nesta situação, espera pela ordem de deslocar uma peça a partir de um ponto inicial A para um destino B. Quando tal ordem é recebida, o manipulador começa por mover-se na horizontal e só desce quando a coordenada horizontal do ponto A é atingida.

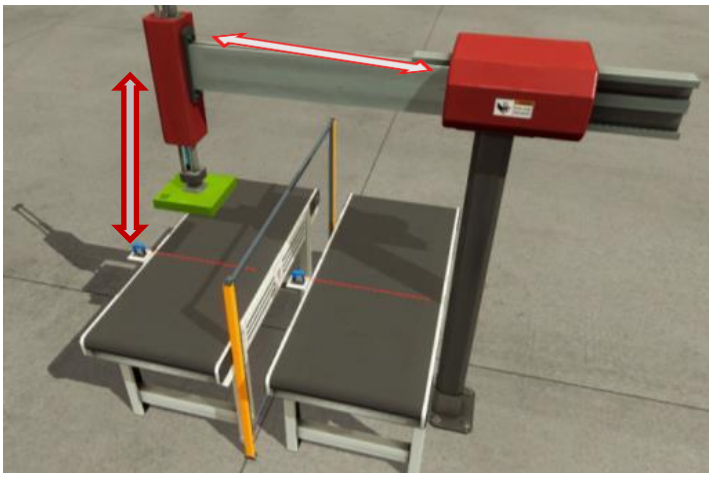

Ilustração 4.3.14: Transferência duma peça por um manipulador *Pick and Place* de dois eixos

Quando o sensor do manipulador deteta a peça, esta é recolhida pela ventosa pneumática, evento após o qual o manipulador volta a subir. A operação de colocar a peça no seu destino é feita segundo a mesma sequência, sendo a ventosa desativa apenas quando o ponto B é atingido. Por fim, o robô volta a subir e entra em repouso até receber nova ordem de funcionamento.

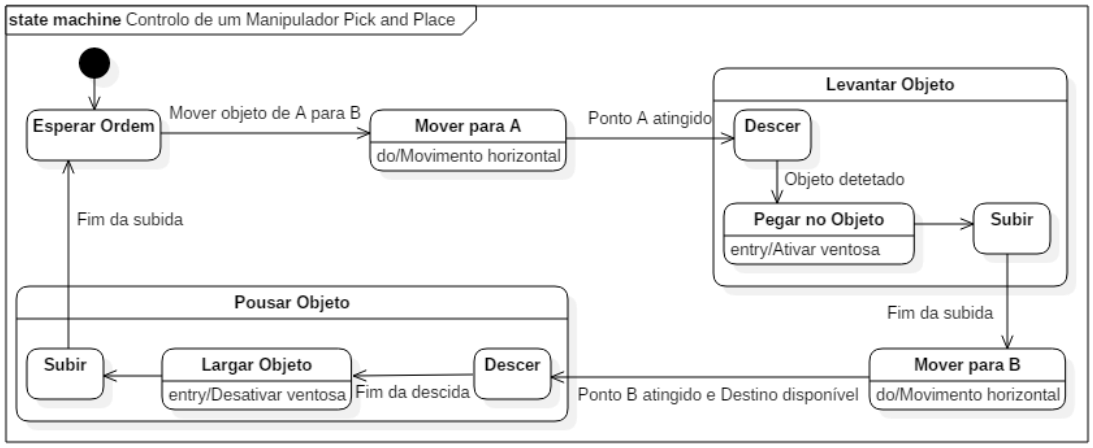

Ilustração 4.3.15: Diagrama de estados da transferência de peças por um manipulador Pick and Place

Com esta apresentação, ficam estabelecidos vários conjuntos de equipamentos que serão usados para desenvolver os cenários virtuais, assim como a sua lógica funcional e de controlo. Estas definições preliminares permitirão uma apresentação mais compacta e uma compreensão mais direta dos casos de estudo do Capítulo 5.

4.4. Coordenação do Controlo dos Cenários Virtuais

Para que se observe um comportamento fluido, coerente e isento de erros em cada ambiente virtual, é necessário programar cada controlador lógico de modo que estes façam uma correta gestão do programa e das suas variáveis. Isto implica certas especificações de comportamento lógico por parte dos controladores:

- Cada controlador só inicia e prossegue a monitorização e controlo dos componentes – sensores e atuadores – do cenário virtual quando este se encontra ativo e não no modo de pausa;
- A desativação ou reiniciação do cenário conduzirá também à reiniciação do estado dos controladores e das suas variáveis. Nesta situação, os controladores apenas retomarão o funcionamento da instalação quando se verificar que todos os controladores foram devidamente reiniciados;
- Através de um painel de operador ou HMI, um utilizador pode ordenar o início ou paragem do funcionamento de um ambiente virtual.

As soluções adotadas para concretizar estes requisitos são apresentadas nesta secção. Estas soluções baseiam-se na utilização dum controlador coordenador, responsável pela gestão global do cenário virtual e dos restantes controladores, ditos seguidores. Por este motivo, entende-se que se tratam de soluções de controlo cooperativo, ou coordenado (34). Esta classe de controlo difere do dito controlo de supervisão, ou hierárquico (34), na medida em que todos os controladores se encontram ao mesmo nível hierárquico, ao passo que na supervisão o controlador coordenador apresentaria precedência hierárquica sobre os demais.

O diagrama da Ilustração 4.4.1 exprime a interação entre dois controladores responsáveis pelo controlo distribuído e coordenado dum mesmo cenário virtual.

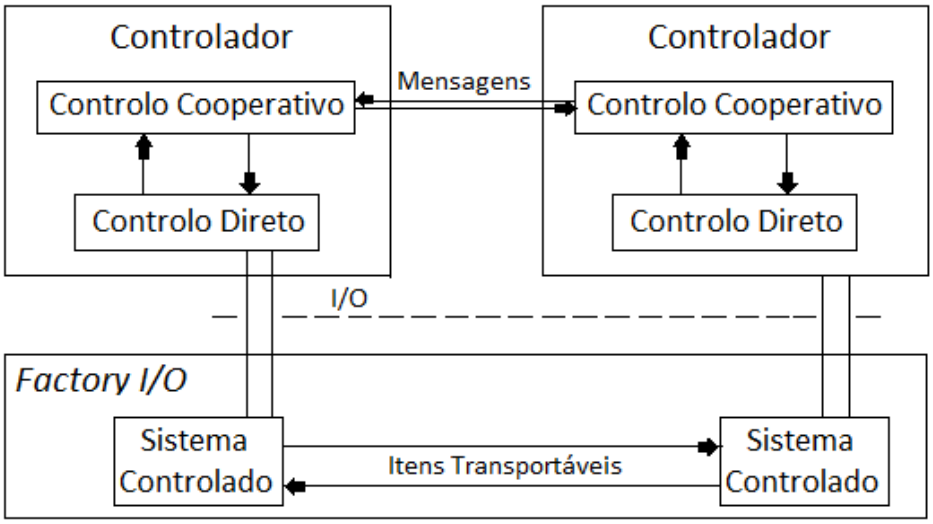

Ilustração 4.4.1: Controlo distribuído e coordenado dum cenário virtual

4.4.1. Variáveis de Controlo Operacional

Começa-se por definir as variáveis de interação entre os cenários e os controladores, relevantes para o controlo geral do ambiente virtual. Os algoritmos de controlo serão definidos em torno destas variáveis.

Os controladores alterarão o seu modo operacional e em resposta aos valores de dois conjuntos de variáveis binárias (Tabela 4.4.1). O primeiro diz respeito ao estado de operação do cenário, consistindo nas variáveis *FactoryRun*, *FactoryPause* e *FactoryReset*, que respetivamente representam se o cenário está ativo ou inativo, pausado ou não, ou a ser reiniciado.

Tabela 4.4.1: Variáveis de estado do cenário e de comando da operação

Variável	Significado
Estado do Cenário:	
FactoryRun	Cenário ativo (modo de corrida) ou inativo (modo de edição)
FactoryPause	Cenário pausado ou não pausado
FactoryReset	Cenário está ou não a ser reiniciado
Botões:	
BotãoStart	Início de operação
BotãoReset	Reinício de operação e saída do estado do sistema
BotãoStop	Paragem do sistema/Entrada no modo de limpeza
BotãoEmergência	Paragem de Emergência/Entrada no modo de segurança

O outro conjunto trata-se dos sinais afetos aos botões presentes no painel do operador ou, no caso de alguns cenários, na HMI. Estes botões constituem uma interface entre um operador e a instalação, permitindo o comando desta. O propósito de cada botão é detalhado mais adiante, nesta secção.

4.4.2. Controlo *Start*/*Stop* ou *On*/*Off*

A primeira solução de controlo, representada no diagrama da Ilustração 4.4.2, possui apenas dois estados, *On* e *Off*. Os controladores iniciam sempre no estado *Off*, no qual esperarão por uma transição ascendente da variável do botão *Start*, mas apenas quando o cenário estiver ativo, não reiniciado e o botão *Stop* não atuado.

Encontrando-se no estado *On*, dá-se o início operacional do cenário: os atuadores entram em funcionamento e os itens são movidos conforme pretendido. O controlador regressa ao estado *Off* quando o *botão* for premido.

No estado, *Off*, os atuadores são todos imediatamente parados, independentemente do seu estado de operação. Se se der a desativação ou reiniciação do cenário, o controlador procederá à reiniciação de todas as variáveis de controlo. O controlador apenas regressa ao seu estado inicial quando o cenário estiver novamente ativo e as variáveis terem sido reiniciadas. Esta última condição é indicada pelo valor positivo da variável *SinalResetConfirm*. O mecanismo por detrás da reiniciação das variáveis é detalhado mais adiante, na secção 4.4.4. A função de cada modo de controlo é resumido na Tabela 4.4.2.

Em cada momento, os botões cuja atuação pode acionar uma mudança de estado terão a respetiva luz acesa.

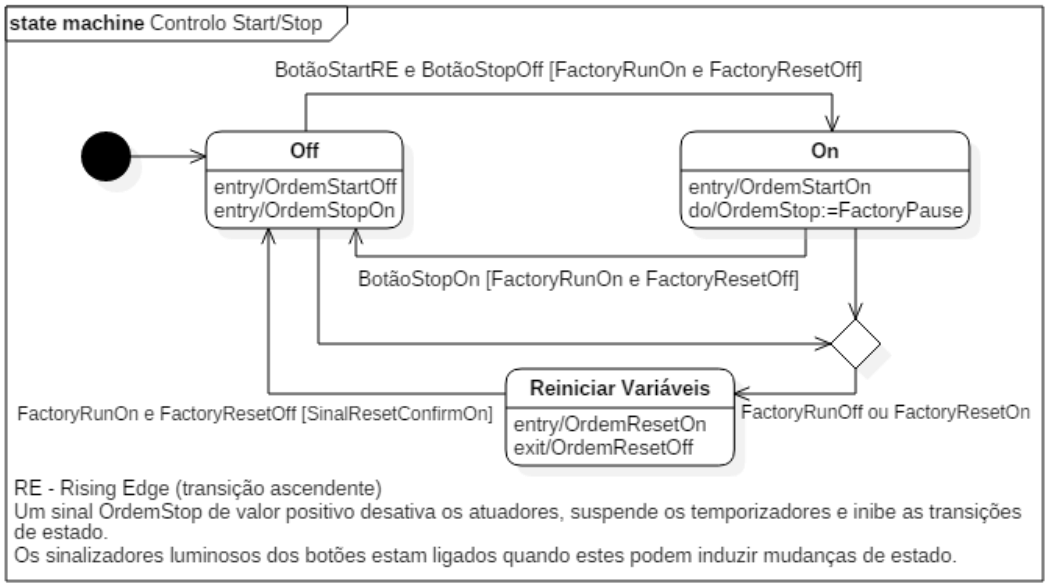

Ilustração 4.4.2: Diagrama de estados do controlo *On*/*Off*

Tabela 4.4.2: Modos de Controlo *On*/*Off*

Modo do Controlador:	Funções:
Off	Desliga os atuadores e Inibe as transições de estado
On	Permite o comando dos atuadores e as transições de estado; Inibe as transições de estado e suspende os temporizadores em caso do cenário entrar em modo de pausa
Reiniciar variáveis	Reinicia as variáveis das tarefas e dos controladores; Regressa ao modo Off quando esta tarefa for concluída e o *Factory I/O* regressar ao modo de corrida

4.4.3. Controlo *Start*/Limpeza/*Stop*/Segurança

Os modos de controlo definidos anteriormente foram mais tarde desenvolvidos de modo a permitir um controlo mais sofisticado do ambiente virtual. Para isso, introduziram-se duas novas funções: o modo de segurança e o modo de limpeza. A inserção destes dois estados dentro do controlo é visível no diagrama de estados da Ilustração 4.4.3.

O modo de limpeza é ativado quando, no modo de funcionamento normal, o botão *Stop* estiver atuado. Neste modo, os emissores são desativados o cenário funciona normalmente até todos os itens terem sido encaminhadas para os eliminadores. É ainda utilizado um temporizador que é apenas ativado quando não existem itens a serem detetados pelos sensores dos tapetes de entrada e as restantes regiões do cenário estiverem livres. Quando este temporizador atingir o tempo definido, o cenário regressará ao modo de paragem.

O modo de segurança é ativado quando o botão de emergência for premido, em qualquer situação em que o cenário se encontre. Aí, todos os atuadores a param e o controlador espera pela libertação do botão de emergência, seguida pela atuação do botão de *Reset* para regressar ao funcionamento normal. Dando-se a reiniciação do cenário, o controlador sai também do modo de segurança.

No âmbito da vida real, a atuação dum botão de emergência provoca um corte da alimentação dos atuadores por via de cablagem física. Tal não se verifica no Factory I/O, dado que estes botões são interpretados como um outro botão qualquer. A forma mais realista de impor este comportamento nos ambientes virtuais seria através da utilização dum controlador de segurança encarregado de monitorizar os botões de emergência.

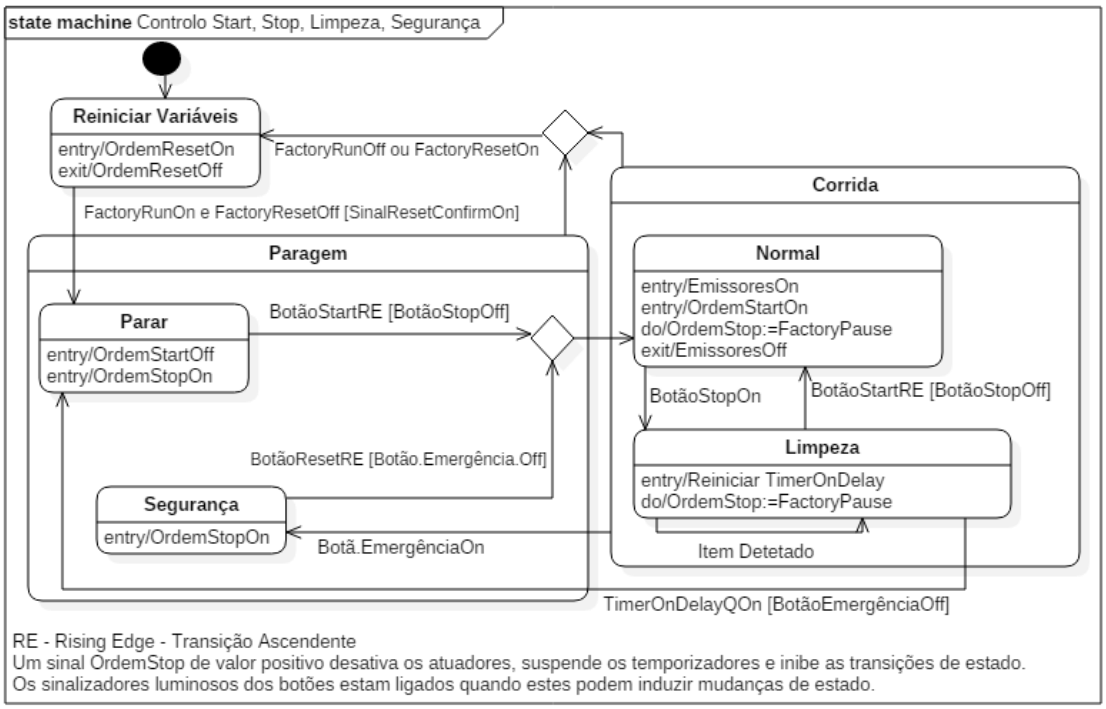

Ilustração 4.4.3: Diagrama de estados do Controlo *Start*/Limpeza/*Stop*/Segurança

Tabela 4.4.3: Modos de controlo *Start*/Limpeza/*Stop*/Segurança

Modo do Controlador:	Funções:
Paragem: Parar	Desliga os atuadores e Inibe as transições de estado
Paragem: Segurança	Desliga os atuadores e Inibe as transições de estado; Apenas regressa ao modo de corrida depois da atuação do botão *Reset*
Corrida: Normal	Permite o comando dos atuadores e as transições de estado; Inibe as transições de estado e suspende os temporizadores em caso do cenário entrar em modo de pausa
Corrida: Limpeza	Desliga os emissores; Regressa ao modo de paragem quando deixarem de ser detetados itens no ambiente
Reiniciar variáveis	Reinicia as variáveis das tarefas e dos controladores; Regressa ao modo de paragem quando esta tarefa for concluída e o *Factory I/O* regressar ao modo de corrida

4.4.4. Sincronização Funcional entre Controladores

Até este ponto, foram especificadas as soluções de coordenação tal como são implementadas no controlador coordenador. As mesmas transições de estado necessitarão de ser aplicadas aos restantes controladores do cenário. Como tal, o controlador coordenador terá como como função adicional o envio de sinais a cada controlador seguidor de modo a provocar as mudanças de estado de operação.

Para além das variáveis específicas à operação de cada cenário, os controladores ou tarefas trocarão os sinais *Start*, *Stop*, *Reset* e *ResetConfirm*. Os três primeiros provocam,

respetivamente, o início, paragem, ou reiniciação dos controladores seguidores de forma similar àquela que é ilustrada. Quando é ordenado a reiniciação das suas variáveis, cada controlador devolve um sinal *ResetConfirm*, que informa o coordenador que as variáveis foram devidamente reiniciadas. Esta lógica encontra-se representada no diagrama da Ilustração 4.4.4. Estas variáveis encontram-se listadas na Tabela 4.4.4.

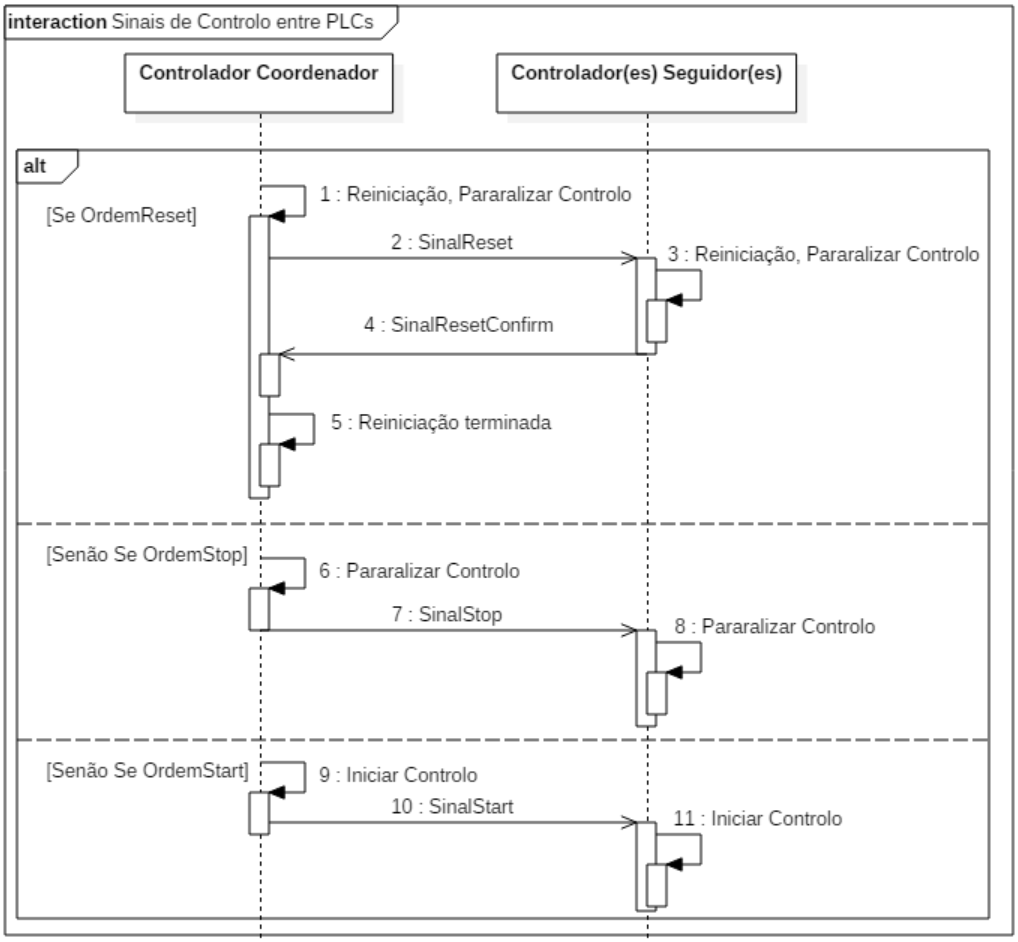

Ilustração 4.4.4: Diagrama de sequência das ordens *Start*, *Stop* e *Reset*

Para efeitos de coerência, estas variáveis designam-se por 'ordens' quando são internas ao respetivo controlador e por 'sinais' quando são transmitidas entre PLCs.

Tabela 4.4.4: Ordens e sinais trocados entre tarefas e controladores

Ordem/Sinal:	Significado:	Enviado pelo controlador/tarefa:
Start	Início do funcionamento	Controlador coordenador/tarefa de coordenação
Stop	Paragem ou interrupção do funcionamento	
Reset	Paragem do funcionamento e reiniciação das variáveis	
ResetConfirm	Confirmação da reiniciação	Controlador seguidor/restantes tarefas

Com isto, fica resumida a implementação do controlo global dos cenários e dos PLCs. Em conjunto com os elementos de controlo direto apresentados anteriormente, fica estabelecida a maior parte dos aspetos de controlo lógico relativos aos casos de estudo do Capítulo 5.

4.5. Conclusões do Capítulo

Ao abordar as questões levantadas pelo estudo dos sistemas de eventos discretos e pelos sistemas de fabricação responsivos, destaca-se uma necessidade de adotar ferramentas e metodologias para a análise, modelação e conceção dos cenários a desenvolver e, em particular, dos controladores lógicos responsáveis pela sua operação.

Dada a natureza dos ambientes virtuais, estes constituem uma forma estudar conceitos relativos aos sistemas de eventos discretos e ao seu controlo. É também possível praticar a conceção de modelos de controlo adequados à automação ágil e responsiva, e que permitem a adaptação rápida a modificações dos requisitos funcionais ou no equipamento utilizado.

Através do estudo de técnicas de modelação, foi possível adotar um conjunto de normas de modelação de sistemas. Como tal, os casos de estudo concebidos poderão ser representados por uma variedade de diagramas que concedem uma visualização em vários níveis. Esses níveis são, nomeadamente, o nível da implementação física, ou de *hardware*, da comunicação e do comportamento dinâmico.

Para representar a lógica por detrás do controlo dos ambientes virtuais, assim como alguns subconjuntos de componentes ou operações, decidiu-se usar os diagramas de estados e diagramas de atividade. Isto deve-se ao facto destes modelos serem mais detalhados na apresentação das ações e transições de estado dum sistema.

No entanto, para demonstrar o comportamento lógico e a dinâmica funcional dos cenários a construir, de um ponto de vista mais amplo, optou-se por uma especificação mais simples e compacta, as Redes de Petri, que serão apresentadas no início do Capítulo 5.

Escolheram-se os diagramas de sequência para apresentar a comunicação entre controladores, mais especificamente as mensagens entre eles trocadas, a natureza das mesmas e a sequência em que ocorrem. Para a representação da implementação das comunicações, optou-se pelos diagramas de componentes que constituem uma vista estática do sistema, ilustrando as interfaces entre os seus elementos. Estas duas espécies de diagramas serão utilizadas mais proeminentemente do Capítulo 5, onde os controladores lógicos serão devidamente postos em funcionamento.

Na parte final deste capítulo, foi feita uma definição e modelação preliminar de vários subconjuntos de componentes *Factory I/O* e de determinadas soluções para a execução do controlo distribuído dos cenários. A apresentação prévia destes blocos permitirá uma documentação mais compacta e compreensível dos casos de estudo a tratar no próximo capítulo.

Capítulo 5 – Modelação e Programação do Controlo Distribuído

Este capítulo é maioritariamente dedicado ao controlo cooperativo de sistemas de automação. Os sistemas alvo são cenários virtuais construídos por integração de recursos elementares apresentados no anterior capítulo - cada um deles controlado diretamente por um PLC particular - acrescentando-se agora a camada de comunicações e controlo cooperativo necessários a um efetivo controlo distribuído.

O capítulo inicia-se pela apresentação de uma técnica de modelação adicional, as Redes de Petri, cuja utilização foi especificamente reservada para a conceção e definição dos casos de estudo a tratar neste capítulo. Esta tarefa assenta essencialmente nas redes de Petri, não só por lidarem bastante bem com a complexidade dos sistemas de eventos discretos aqui abordados, mas sobretudo porque, sendo executáveis, permitem analisar e validar de antemão o modelo de controlo a desenvolver.

O resto deste capítulo encontra-se dividido em vários subcapítulos, cada um correspondendo a um caso de estudo mais ou menos elaborado, constituído por um cenário virtual, pelos controladores lógicos selecionados e pela implementação do controlo e das comunicações. Os casos de estudo são expostos de forma consistente, começando-se pela definição do cenário virtual em causa, sua estrutura e comportamento pretendido, descrevendo-se depois a distribuição e implementação do controlo e terminando-se com a descrição da realização das comunicações.

Esta documentação é suportada, por um lado, pelos modelos de controlo direto e controlo cooperativo apresentados no capítulo anterior - e que são comuns a vários dos casos de estudo - e, por outro pela inclusão de modelos UML adicionais e de redes Petri, que ilustram vários aspetos da execução do ambiente virtual.

Após apresentar cada caso de estudo, são feitas observações relativas às facilidades e também aos desafios por detrás da sua execução, e são dados exemplos de possíveis desenvolvimentos aplicáveis a eles. No final deste capítulo são feitas conclusões gerais relativas ao que foi exposto ao longo dele.

5.1. Redes de Petri

Em 1962, Carl A. Petri criou uma ferramenta matemática para o estudo de comunicações com autómatos: as "Redes de Petri" (45).

As redes de Petri, enquanto ferramenta gráfica e matemática, são usadas em inúmeros contextos, incluindo a modelação, análise formal e conceção de sistemas de eventos discretos; em particular, no âmbito dos sistemas de produção ágeis e flexíveis (45).

As Redes de Petri podem representar certas propriedades e comportamentos tais como a sincronização de processos, impactos possíveis de eventos assíncronos, operações sequenciais, operações concorrentes e conflitos de partilha de recursos (35).

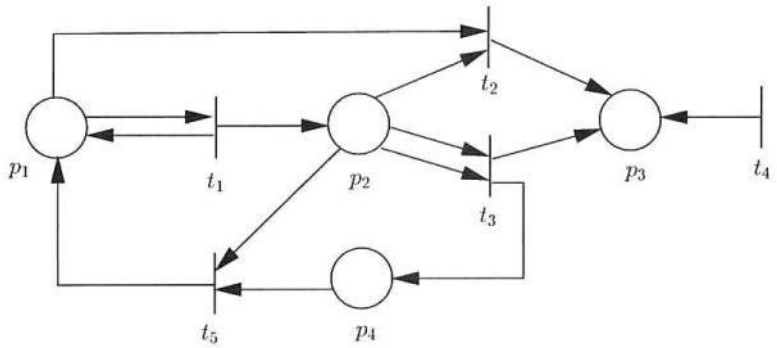

Ilustração 5.1.1: Exemplo de uma rede de Petri (35)

Uma rede de Petri (Ilustração 5.1.1), na sua representação mais básica, é constituída por três classes de elementos:

- Lugares, que representam os possíveis estados em que um dado sistema se pode encontrar;
- Transições, que consistem em eventos ou ações que provocam mudanças de estado;
- Arcos, que fazem a ligação entre lugares e transições (e vice-versa), revelando o sentido no qual o sistema evolui.

Os lugares que estão ocupados num dado instante dum determinado sistema são assinalados por marcas. Um lugar pode conter mais do que uma marca, o que é útil para quantificar a disponibilidade de um determinado recurso, como a ocupação de um *buffer* ou máquina, entre outros casos. A uma rede de Petri, atribui-se marcas aos seus lugares de modo a indicar o estado inicial do sistema. A isso denomina-se a marcação inicial da rede.

Uma transição diz-se habilitada quando os lugares de um sistema cujos arcos apontam para essa transição estão ocupados por marcas. Apenas estando cumprida essa condição, juntamente com quaisquer outras condições lógicas que dizem respeito ao sistema, pode uma transição ser disparada. Após o disparo de uma transição, as marcas existentes nos lugares de entrada são eliminadas e são adicionadas novas marcas aos lugares à saída da transição.

Cada arco pode apenas ligar um lugar a uma transição, e vice-versa, pois as mudanças de estado (ou de marcação da rede) só ocorrem com o disparo de uma transição. A cada arco pode ser atribuído um peso, que corresponderá à quantidade de marcas a serem eliminadas ou adicionadas aos lugares de saída ou entrada, respetivamente, por cada disparo.

A (Ilustração 5.1.2) mostra um exemplo bastante trivial duma caixa a ser transferida entre dois tapetes de movimentação. Inicialmente, a caixa está pousada no tapete da esquerda, situação indicada pela marcação inicial da rede, com uma marca em P1 e os restantes lugares vazios. Depois, inicia-se a transferência da caixa entre os dois tapetes, passando a marca de P1 para P2. Terminada a deslocação, a caixa passa a pertencer apenas ao tapete da direita, e a marca passa a pertencer ao lugar P3. O disparo das transições T1 e T2 representam o início e o fim, respetivamente, da operação de transporte da caixa.

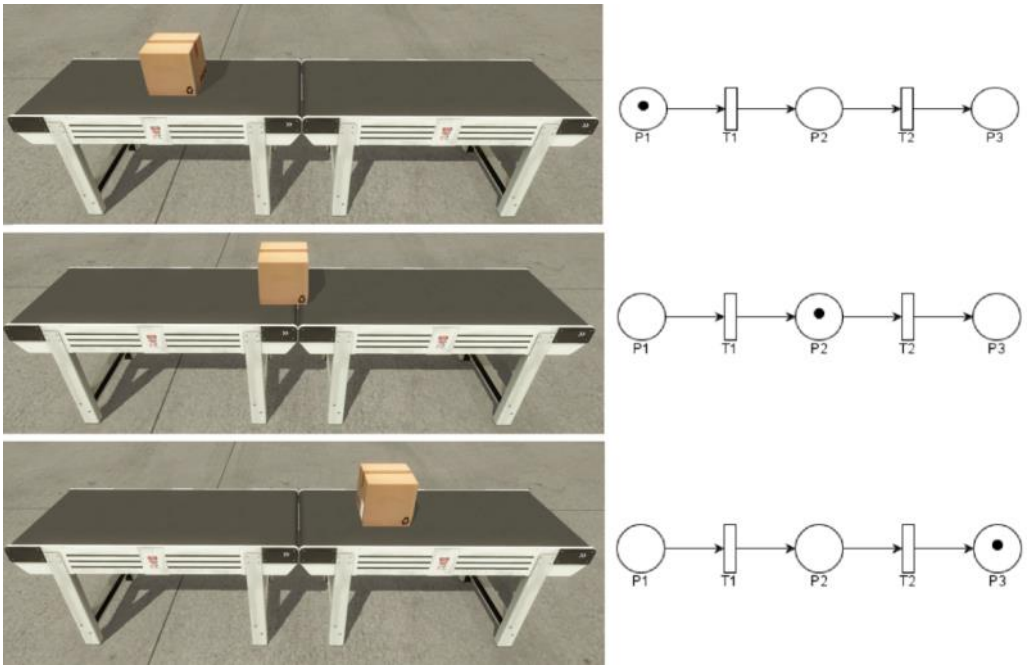

Ilustração 5.1.2: Representação em rede de Petri da transferência duma caixa entre dois tapetes transportadores

Existem vários formalismos, adotados por diversos autores (35) (46), para definir sistemas com maior grau de complexidade, para hierarquizar um sistema em subsistemas, ou para transmitir uma maior quantidade de informação através da rede. Exemplos incluem redes de Petri temporizadas (35) e redes de Petri coloridas (46).

Neste trabalho, dado que os sistemas concebidos apresentam um baixo grau de complexidade, a maior parte destes foi modelada com o formalismo básico, ou seja, com lugares, transições, arcos e marcações iniciais.

Nalguns casos mais elaborados, utilizou-se também arcos inibidores. Estes arcos, representados por uma linha terminada num círculo, indicam que a transição a jusante desse arco só estará habilitada se o número de marcas presente no lugar a montante for inferior ao peso do arco (Ilustração 5.1.3).

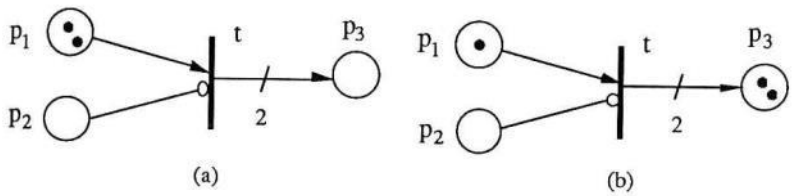

Ilustração 5.1.3: Disparo de uma transição com um arco inibidor (45)

5.1.1. Análise de Redes de Petri

A análise de uma rede de Petri pode constituir um ponto de partida para aferir o funcionamento do sistema modelado. As redes podem ser avaliadas sob diversos parâmetros, dos quais se destacam:

- Cobertura (*reachability* ou *coverability*): A possibilidade de um determinado estado ser alcançável a partir de outro, em particular, da marcação inicial, através de uma sequência de transições (35) (45);
- Limitação (*boundness*): O número máximo de marcas que pode conter cada lugar, para qualquer estado alcançável a partir do inicial (35) (45);
- Segurança (*safety*): Relacionada com a limitação de uma rede. Uma rede limitada a uma marca em todos os estados costuma dizer-se "segura" (35) (45);
- Vivacidade (*liveliness*): A capacidade de disparar qualquer transição a partir do estado inicial, através de uma sequência de transições. Uma transição que nunca possa ser disparada diz-se "morta" (35) (45);
- Conservação (*conservativeness*): Quando um número de marcas numa rede é constante para todos os estados alcançáveis a partir do inicial, essa rede diz-se estritamente conservativa. Esta propriedade pode ser relevante em sistemas nos quais os recursos não devem ser criados nem destruídos (35) (45);
- Reversibilidade (*reversibility*): A capacidade de um sistema de regressar ao seu estado inicial a partir de qualquer outro estado para o qual tenha progredido. É um fator importante para garantir a recuperação de um sistema de controlo de processo a partir de uma situação de erro (45);
- Ocorrência de situações de impasse (*deadlock*): O sistema enfrenta um impasse quando se encontra num estado no qual nenhuma transição esteja habilitada, logo o sistema não pode alterar o estado no qual se encontra. Uma rede dita "viva" pode garantir uma operação livre de impasses (*deadlock-free*).

5.1.1.1. Árvore de Cobertura

A árvore de cobertura (*reachability tree* ou *coverability tree*) é uma técnica de análise gráfica que consiste na construção duma árvore cujos nodos representam os estados possíveis duma dada rede de Petri e cujos arcos representam transições.

Inicia-se pela marcação do estado inicial da rede, a partir do qual se desenham os arcos correspondentes a cada transição habilitada nesse estado, marcando-se de seguida o estado alcançado pelo sistema após essa transição. Ao repetir o procedimento para cada novo estado, é - teoricamente - possível determinar todos os estados possíveis do sistema representado.

Este método potencia a avaliação duma rede de Petri de acordo com todos os parâmetros acima mencionados, em particular segundo a sua acessibilidade, limitação e reversibilidade (35).

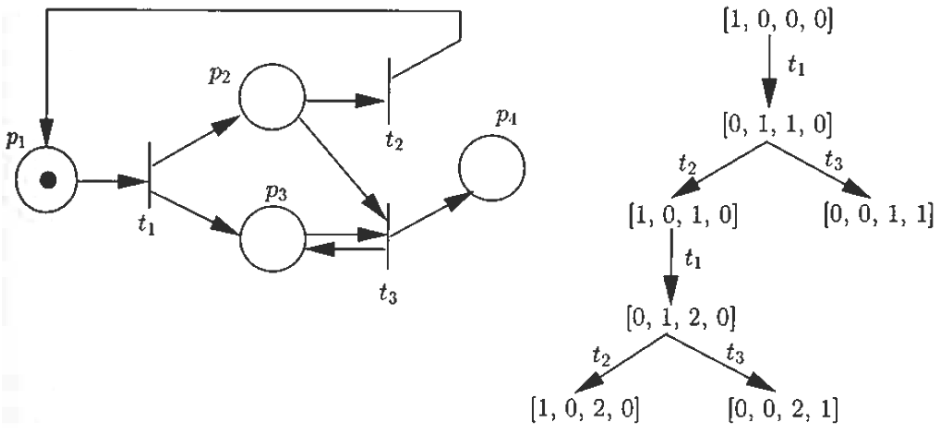

Ilustração 5.1.4: Rede de Petri com respetiva árvore de cobertura (35)

5.1.1.2. Evasão de Situações de Impasse por Redes de Petri

Em (47) é proposto um método para conceber estratégias de controlo anti-impasse em sistemas de automação flexível, usando como base as redes de Petri. Pretende-se que este método seja minimamente restritivo, permitindo maximizar o uso dos recursos do sistema (47).

Este método consiste na determinação de situações de *deadlock* possíveis, através da análise duma rede da Petri representativa do sistema em estudo. Para cada dada situação de impasse, são determinadas as sequências de transições que podem conduzir a ela. À rede de Petri usada, são adicionados novos lugares que impõem restrições às transições, impedindo que se deem as sequências que provocariam o *deadlock* (47). Este método será aplicado nos casos de estudo expostos nos subcapítulos 5.4, 5.5 e 5.7.

Graças à sua flexibilidade, simplicidade e capacidades enquanto método de análise de sistemas, as redes de Petri apresentam-se como sendo uma ferramenta para representação e especificação dos casos de estudo a desenvolver.

5.2. Encaminhador por mesa rotativa, com duas entradas e duas saídas, Caso 1

Este primeiro cenário (Ilustração 5.2.1) consiste na interseção entre dois tapetes alimentadores (tapetes 1 e 2) e dois tapetes de saída (tapetes 3 e 4) com uma mesa rotativa. Sobre estes atuadores serão transportadas e encaminhadas caixas sobre paletes.

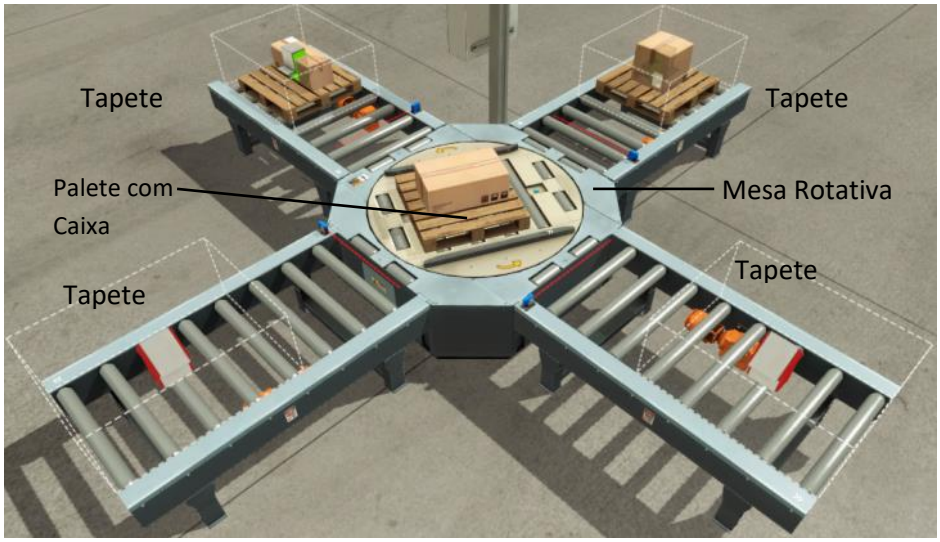

Ilustração 5.2.1: Encaminhador por mesa rotativa, com duas entradas e duas saídas

O cenário dispõe também de um painel de operador (Ilustração 5.2.2) com botões e mostradores numéricos. Estes últimos servem como contadores que indicam quantas caixas foram recebidas de cada tapete de entrada e quantas foram encaminhadas para os tapetes de saída. Os botões de *Start* e de *Stop* permitem a um operador dar à instalação a ordem de início de operação ou de paragem súbita.

Ilustração 5.2.2: Painel de operador

5.2.1. Comportamento

Neste cenário, pretendeu-se que as caixas - transportadas sobre paletes - emitidas a partir dos tapetes 1 e 2 fossem reencaminhadas para os tapetes 3 e 4 sobre uma determinada lógica de envio.

Definiu-se que a lógica de encaminhamento seria a seguinte: as caixas vindas do tapete 1 têm prioridade sobre as que vêm do tapete 2 e são sempre encaminhadas para o tapete 3. As caixas do tapete 2 são enviadas alternamente para os tapetes 3 e 4, sendo a primeira enviada para o tapete 4. Os quatro mostradores numéricos mostram o número de paletes emitidas pelos tapetes alimentadores e o número de paletes recebido pelos tapetes de saída.

5.2.2. Modelação por Rede de Petri

Na Ilustração 5.2.3 está apresentada uma rede de Petri ilustrativa deste cenário. A atribuição dos lugares e transições aos componentes físicos e eventos do cenário está apresentada na Tabela 5.2.1.

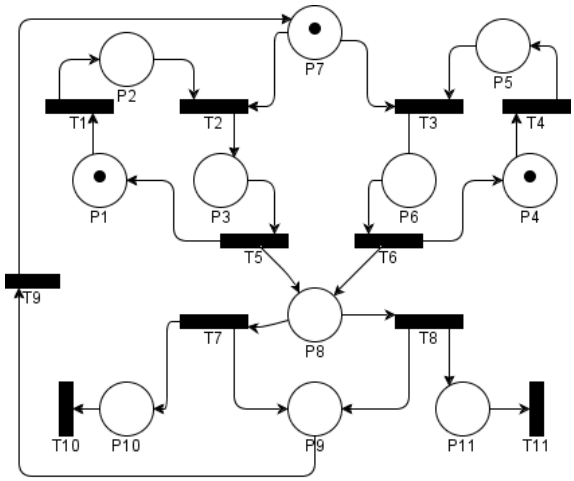

Ilustração 5.2.3: Rede de Petri do cenário virtual

Tabela 5.2.1: Legenda da rede de Petri

Lugar	Descrição	Transição	Descrição
P1	Tapete 1 inativo (espera receber palete)	T1	O sensor 1 deteta uma palete; envia pedido para transferi-la para a mesa
P2	Tapete 1 contém uma palete	T2	Mesa rotativa aceita pedido para receber palete do tapete 1
P3	Transferência de palete do tapete 1 para a mesa rotativa	T3	Mesa rotativa aceita pedido para receber palete do tapete 2
P4	Tapete 2 inativo (espera receber palete)	T4	O sensor 2 deteta uma palete; envia pedido para transferi-la para a mesa
P5	Tapete 2 contém uma palete	T5	Palete sai do tapete 1
P6	Transferência de palete do tapete 2 para a mesa rotativa	T6	Palete sai do tapete 2
P7	Mesa rotativa à espera de uma palete	T7	Início do envio da palete para o tapete 3
P8	Mesa rotativa detém uma palete e reorienta-se para o tapete de saída.	T8	Início do envio da palete para o tapete 4
P9	Mesa rotativa move a palete para o tapete de saída	T9	Sinal descendente do sensor respetivo do tapete de saída que recebe a palete.
P10	Tapete 3 recebe uma palete e leva-a ao eliminador	T10	A palete abandona o tapete 3
P11	Tapete 4 recebe uma palete e leva-a ao eliminador	T11	A palete abandona o tapete 4

Quanto aos lugares P10 e P11, a rede concebida não impõe um limite físico para o número de paletes que os tapetes 3 e 4 possam deter, o que sugere um sistema inseguro. Contudo, no cenário as paletes que entrem nestes tapetes são rapidamente encaminhadas para os respetivos eliminadores, pelo que não constituem um problema. Numa situação real, seria fundamental garantir que a instalação parasse - ou alterasse conformemente - o seu funcionamento perante um risco de exceder o limite de ocupação dos tapetes.

Através da análise da rede, é possível observar que a rede é reversível. Após a saída de uma palete da mesa rotativa, o sistema pode regressar ao estado inicial {P1, P4, P7}. Relativamente aos lugares P1 a P9, a rede é confinada a uma marca, pois esses lugares nunca podem exceder o valor unitário.

5.2.3. Distribuição do Controlo Lógico

O controlo do cenário foi distribuído entre um *SoftPLC Codesys Control Win* V3 e o PLC *Siemens* S7-1200. O *softPLC Codesys* V3.5 foi programado para ler os sensores difusos na periferia da mesa rotativa e comandar os tapetes de rolos. O controlo da mesa rotativa e o processamento da lógica de encaminhamento das caixas foi efetuado pelo PLC *Siemens* S7-1200. O S7-1200 foi também responsável por ler os sinais dos botões, e de dar ao *softPLC Codesys* ordem de atuação ou de paragem dos tapetes, assim como sondá-lo para saber se existe ou não uma palete a ser detetada por cada um dos sensores de posição.

Na Tabela 5.2.2 são listadas as variáveis trocadas entre o ambiente virtual e os PLCs, sendo indicado qual o controlador a que cada variável foi atribuída, e se esta constitui uma entrada ou uma saída do PLC.

Tabela 5.2.2: Variáveis trocadas entre o cenário e os controladores

Variáveis de entrada:	Tipo de variável:	Controlador:
Factory.Run	Binária	*Siemens* S7-1200
Factory.Pause		
Factory.Reset		
Botão *Start*		
Botão *Stop*		
Sensor de trás da mesa rotativa		
Sensor frontal da mesa rotativa		
Sensor de posição de 0° da mesa rotativa		
Sensor de posição de 90° da mesa rotativa		
Sensores difusivos dos tapetes 1 a 4	4 variáveis binárias	*Codesys Control Win* V3
Variáveis de saída:	Tipo de variável:	Controlador:
Luz do botão *Start*	Binária	*Siemens* S7-1200
Luz do botão *Stop*		
Mostradores 1 a 4	4 variáveis inteiras	
Tapete da mesa – sentido positivo	Binária	
Tapete da mesa – sentido negativo		
Rotação da mesa – sentido positivo		
Rotação da mesa – sentido negativo		
Tapetes de rolos 1 a 4	4 variáveis binárias	*Codesys Control Win* V3

5.2.4. Implementação das Comunicações

A comunicação entre controladores foi feita por *Modbus* TCP, sobre duas maneiras: na primeira, o controlador *Siemens* tomou o papel de cliente e o *SoftPLC Codesys* foi usado como servidor; na segunda, o *Siemens* agiu como servidor e o *Codesys* como cliente. Estes dois casos são representados pelos diagramas de sequência da Ilustração 5.2.4 e da Ilustração 5.2.5, respetivamente.

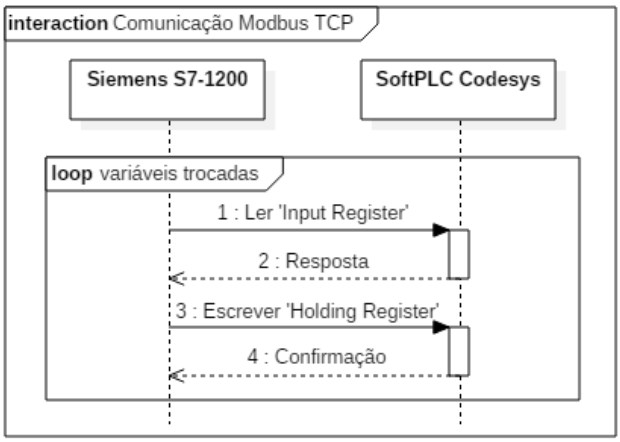

Ilustração 5.2.4: Diagrama de sequência da comunicação por *Modbus* TCP

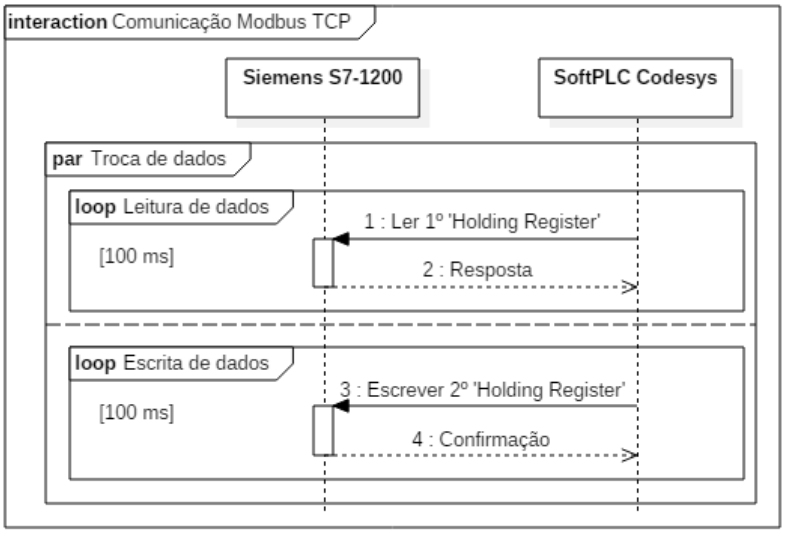

Ilustração 5.2.5: Diagrama de sequência da comunicação por *Modbus* TCP

Em ambos os casos, as variáveis a comunicar são escritas em dois registos de 16 *bit* situados na memória do servidor *Modbus*. Um desses registos é lido pelo cliente e escrito pelo servidor, o outro é lido pelo servidor e escrito pelo cliente.

A realização das comunicações entre os controladores e o *Factory I/O* está representada no diagrama de componentes da Ilustração 5.2.6. É utilizado o *Connect I/O* para ligar o cenário virtual aos controladores. A troca de dados com o *SoftPLC Codesys* é feita através de um servidor *Codesys* OPC DA, enquanto a ligação com o *Siemens* S7-1200 é efetuada através de um driver do fabricante. O *SoftPLC* pode ser corrido tanto no mesmo PC que o *Factory I/O*, tanto como num PC diferente.

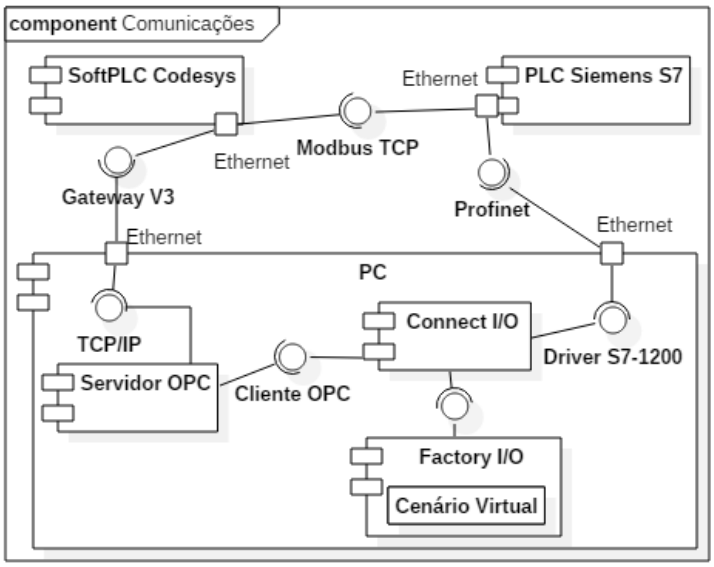

Ilustração 5.2.6: Diagrama de componentes da comunicação entre o cenário e os controladores

5.2.5. Coordenação do Controlo

Relativamente aos algoritmos de controlo implementados, adotou-se o controlo *Start*/*Stop* descrito no Capítulo 4.4. Para o controlo dos atuadores utilizou-se os modelos definidos para os tapetes de entrada, de saída e do cruzamento com elementos transportadores, apresentados no mesmo capítulo. Estes algoritmos foram devidamente convertidos em código para os programas dos controladores. Dada a simplicidade inerente ao controlo dos tapetes, não foi aplicada a função de reiniciação às variáveis do controlador *Codesys*.

A Tabela 5.2.3 apresenta os sinais que são trocados entre os controladores. O controlador *Siemens* envia as ordens *Start* e *Stop*, assim como ordens de movimentação de transferência de paletes dos tapetes de entrada para as mesas. Do *Codesys*, são lidos os valores atuais de cada sensor, cuja leitura permite monitorizar a posição atual de cada palete.

Tabela 5.2.3: Variáveis trocadas entre controladores

Variável:	Tipo de variável:	PLC que a determina:	Significado:
SinalStart	Binária	*Siemens* S7-1200	Início de funcionamento
SinalStop			Paragem de funcionamento
SinalMoverT1			Transferência de palete do tapete 1 para a mesa rotativa
SinalMoverT2			Transferência de palete do tapete 2 para a mesa rotativa
Sensores 1 a 4		*Codesys Control Win* V3	Valores dos sensores difusivos dos tapetes 1 a 4

Para melhor demonstrar o significado destes sinais, elaborou-se dois diagramas de sequência ilustrativos das atividades de transferência duma palete dum tapete alimentador para a mesa (Ilustração 5.2.7), ou desta para um tapete de saída (Ilustração 5.2.8). Nestas atividades, os atores são subsistemas controlados do cenário que trocam mensagens entre si e com as tarefas de controlo.

Quando uma palete é detetada pelo sensor do tapete alimentador, este para a movimentação, esperando pelo sinal *MoverT#* por parte da mesa rotativa para começar a transferência. A mesa rotativa, por sua vez, espera pela ordem, dada pelo controlador, para receber a palete desse tapete. Nesse momento, a mesa reorienta-se e depois ativa o tapete, enviando o sinal *MoverT#*, e dando início à transferência. Esta operação termina com a deteção de sinais descendentes (FE) dos respetivos sensores de posição de cada atuador.

A atividade de transferência da mesa para o tapete de saída é parecida, mas a monitorização por parte dos controladores dá maior enfase aos sinais dos sensores.

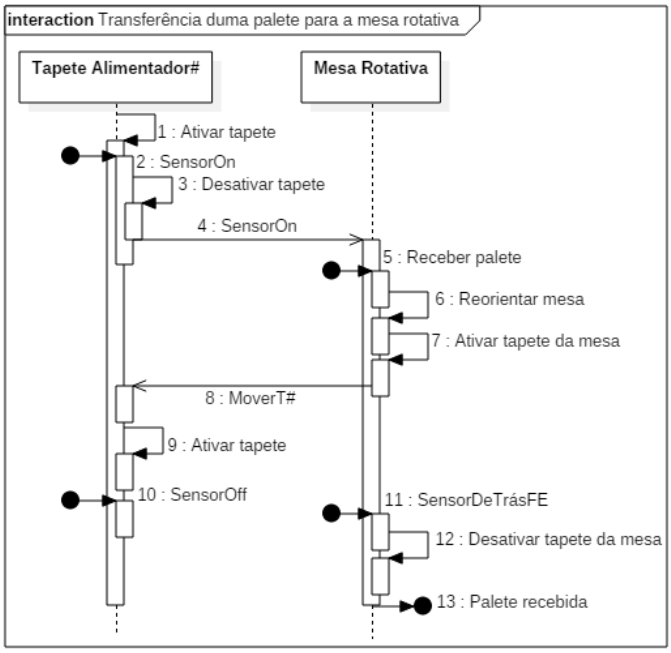

Ilustração 5.2.7: Diagrama de sequência da transferência duma palete para a mesa rotativa

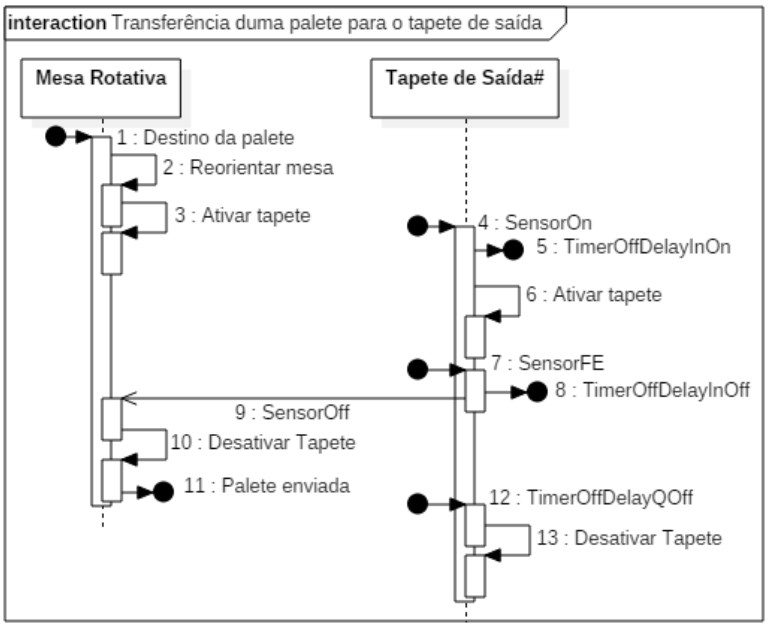

Ilustração 5.2.8: Diagrama de sequência da transferência duma palete para o tapete de saída

5.2.6. Resultados e Observações

Posto o cenário em funcionamento, observou-se um desenrolar fluido da atividade pretendida. Foi implementado um total de três métodos de comunicação, *Modbus* TCP, servidor OPC e comunicação por protocolo proprietário da *Siemens*. A sua implementação foi facilitada pela existência de funções adequadas nos *software* utilizados, consequência da atualidade destes controladores.

Este simples caso de estudo pode ser expandido em vários aspetos, tais como implementando uma seleção da lógica de encaminhamento, através de botões adicionais no painel de operador, ou a implementação duma função de identificação de caixas, com a utilização de um número maior ou de tipos diferentes de sensores.

Estas modificações serão implementadas nalguns dos casos de estudo posteriores.

5.3. Encaminhador por mesa rotativa, com duas entradas e duas saídas, Caso 2

Para este caso de estudo (Ilustração 5.3.1), reutilizou-se a configuração física do cenário anterior para exercer uma implementação do controlo de forma alternativa ao cenário anterior. Foram utilizados controladores diferentes, com um modelo de controlo mais complexo. Os sensores difusivos foram substituídos por sensores retrorrefletivos, pelo que o seu sinal de saída é invertido.

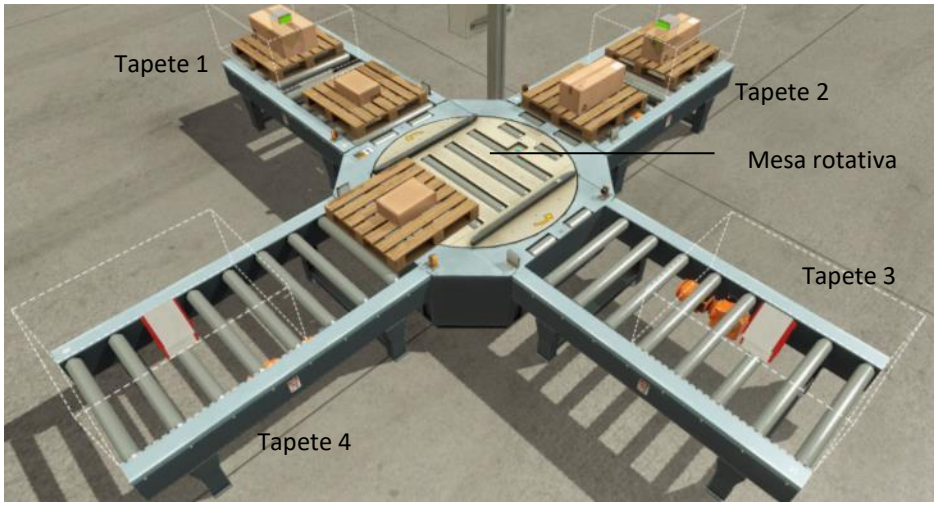

Ilustração 5.3.1: Encaminhador por mesa rotativa, com duas entradas e duas saídas

Este cenário incluiu também um painel de operador (Ilustração 5.3.2) idêntico ao do caso anterior, sendo desta vez atribuídas funções aos botões *Reset* e de emergência.

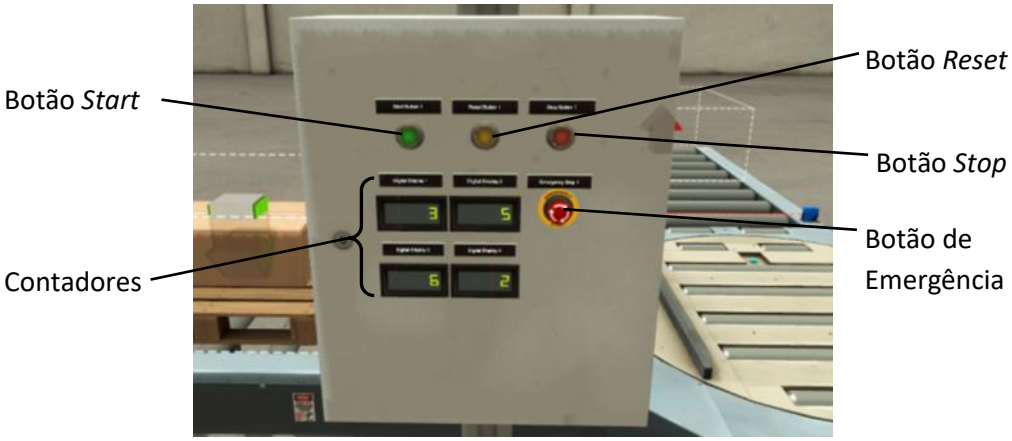

Ilustração 5.3.2: Painel de operador

5.3.1. Comportamento

A lógica de encaminhamento foi também modificada em relação ao do caso anterior. As paletes do tapete 1 continuam a deter prioridade, mas após cada terceira palete emitida por este tapete, a próxima palete do tapete 2 terá prioridade. O destino de cada palete é decidido alternadamente entre os tapetes 3 e 4, e é independente da origem da palete. Os contadores presentes no painel de operador também registarão as paletes que são recebidas de cada tapete de entrada ou emitidas para cada tapete de saída.

5.3.2. Modelação por Rede de Petri

Visto que este caso de estudo apresenta a mesma configuração física que o anterior (secção 5.2), a rede de Petri ilustrativa desse caso (Ilustração 5.3.3) é também aplicável a este, possuindo a mesma legenda (Tabela 5.3.1).

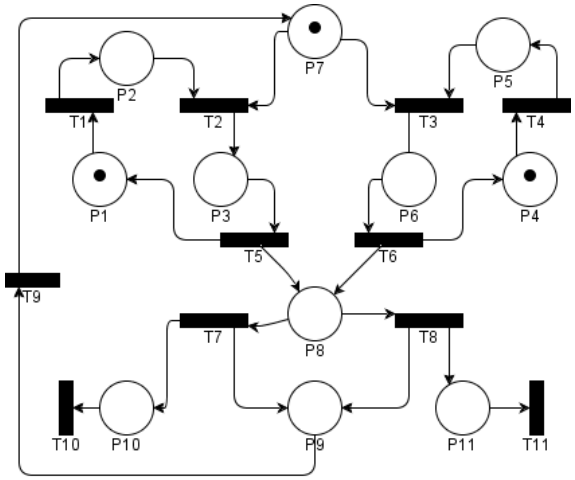

Ilustração 5.3.3: Rede de Petri do cenário virtual

Tabela 5.3.1: Legenda da rede de Petri

Lugar	Descrição	Transição	Descrição
P1	Tapete 1 inativo (espera receber palete)	T1	O sensor 1 deteta uma palete; envia pedido para transferi-la para a mesa
P2	Tapete 1 contém uma palete	T2	Mesa rotativa aceita pedido para receber palete do tapete 1
P3	Transferência de palete do tapete 1 para a mesa rotativa	T3	Mesa rotativa aceita pedido para receber palete do tapete 2
P4	Tapete 2 inativo (espera receber palete)	T4	O sensor 2 deteta uma palete; envia pedido para transferi-la para a mesa
P5	Tapete 2 contém uma palete	T5	Palete sai do tapete 1
P6	Transferência de palete do tapete 2 para a mesa rotativa	T6	Palete sai do tapete 2
P7	Mesa rotativa à espera de uma palete	T7	Início do envio da palete para o tapete 3
P8	Mesa rotativa detém uma palete e reorienta-se para o tapete de saída.	T8	Início do envio da palete para o tapete 4
P9	Mesa rotativa move a palete para o tapete de saída	T9	Sinal descendente do sensor respetivo do tapete de saída que recebe a palete.
P10	Tapete 3 recebe uma palete e leva-a ao eliminador	T10	A palete abandona o tapete 3
P11	Tapete 4 recebe uma palete e leva-a ao eliminador	T11	A palete abandona o tapete 4

5.3.3. Distribuição do Controlo Lógico

O controlo deste ambiente virtual foi repartido entre os PLCs *Siemens* S7-1200 e *Modicon* TSX *Micro*.

O PLC TSX *Micro* ficou encarregue do controlo de todos os sensores e atuadores da instalação, à parte dos dispositivos no painel de operador. É, portanto, o principal responsável pelo desenrolar da atividade. O controlador *Siemens* S7-1200 observa o estado do cenário, interage através do painel de operador e supervisiona o estado do TSX *Micro*, conferindo-lhe autorização para efetuar transições de estado. Este controlador também determina a prioridade e o destino das paletes a encaminhar.

Na Tabela 5.3.2 são listadas as variáveis trocadas entre o ambiente virtual e os controladores, indica-se a que controlador é atribuída cada variável.

Tabela 5.3.2: Variáveis trocadas entre o cenário e os controladores

Variáveis de entrada:	Tipo de variável:	Controlador:
Factory.Run	Binária	*Siemens* S7-1200
Factory.Pause		
Factory.Reset		
Sensores retrorrefletivos dos tapetes 1 a 4		*Modicon* TSX *Micro*
Sensor de trás da mesa rotativa		
Sensor frontal da mesa rotativa		
Posição de 0° da mesa rotativa		
Posição de 90° da mesa rotativa		
Botão *Start*		*Siemens* S7-1200
Botão *Reset*		
Botão *Stop*		
Botão de Emergência		
Variáveis de saída:	**Tipo de variável:**	**Controlador:**
Tapetes de rolos 1 a 4	Binária	*Modicon* TSX *Micro*
Tapete da mesa – sentido positivo		
Tapete da mesa – sentido negativo		
Rotação da mesa – sentido positivo		
Rotação da mesa – sentido negativo		
Luz do botão *Start*		*Siemens* S7-1200
Luz do botão *Reset*		
Luz do botão *Stop*		
Mostradores 1 a 4	Inteira	

Para este cenário, adotou-se o controlo com as funções de *Reset* e de Segurança, definidas na secção 4.4.3. O controlador S7-1200 agiu como controlador líder, comandando a coordenação com o TSX *Micro*. O controlo dos conjuntos dos atuadores foi feito de forma similar ao do cenário anterior.

5.3.4. Implementação das Comunicações

Foi praticada comunicação entre os dois controladores de diferentes maneiras, ou por *Modbus* RTU ou por *Modbus* TCP/IP. Também foi possível executar a ligação, quer usando o S7-1200 como mestre e o TSX Micro como escravo quer ao contrário. Em suma, realizaram-se comunicações *Modbus* de 4 maneiras diferentes (Tabela 5.3.3).

Tabela 5.3.3: Modos de comunicação implementados

Método	Protocolo	*Siemens* S7-1200	*Modicon* TSX *Micro*
1	*Modbus* RTU	Mestre	Escravo
2		Escravo	Mestre
3	*Modbus* TCP	Cliente	Servidor
4		Servidor	Cliente

Em *Modbus* RTU, a ligação física foi feita entre as portas série por RS-232 e por RS-485, utilizando um conversor *Westermo* MD-45. Em *Modbus* TCP, as mensagens *Modbus* são convertidas em protocolo Uni-telway - e vice-versa - por parte do módulo ETZ 510, sendo a ligação entre o módulo e o TSX Micro feita com esse protocolo.

Em todos os casos, os dados trocados entre controladores foram compilados em variáveis do tipo *word* e tratados como registos Modbus. Estas interações são demonstradas no diagrama de sequência da Ilustração 5.3.4. Neste diagrama não é feita distinção entre as variantes série e TCP de Modbus, mas apenas entre cada controlador ser o elemento ativo ou passivo da comunicação.

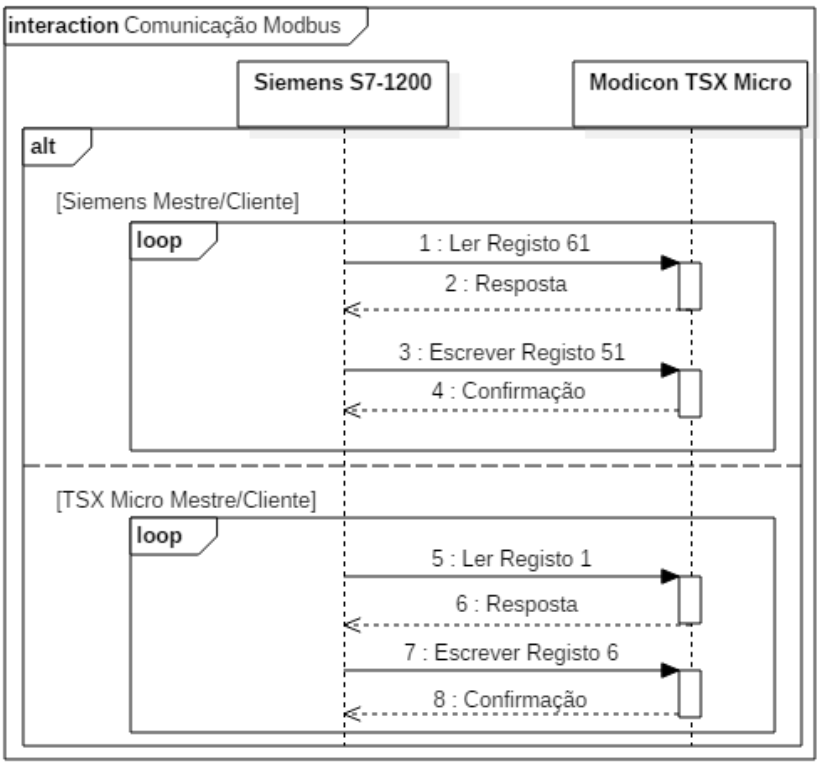

Ilustração 5.3.4: Diagrama de sequência das comunicações por *Modbus*

O diagrama de componentes da Ilustração 5.3.5 apresenta a realização das comunicações. Todos os dispositivos envolvidos, incluindo os módulos ligados ao TSX *Micro*, estão representados, assim como os protocolos de comunicação e os *drivers* do *Connect I/O*. Ambas as variantes das comunicações *Modbus* estão representadas no diagrama, pois utilizam recursos físicos diferentes.

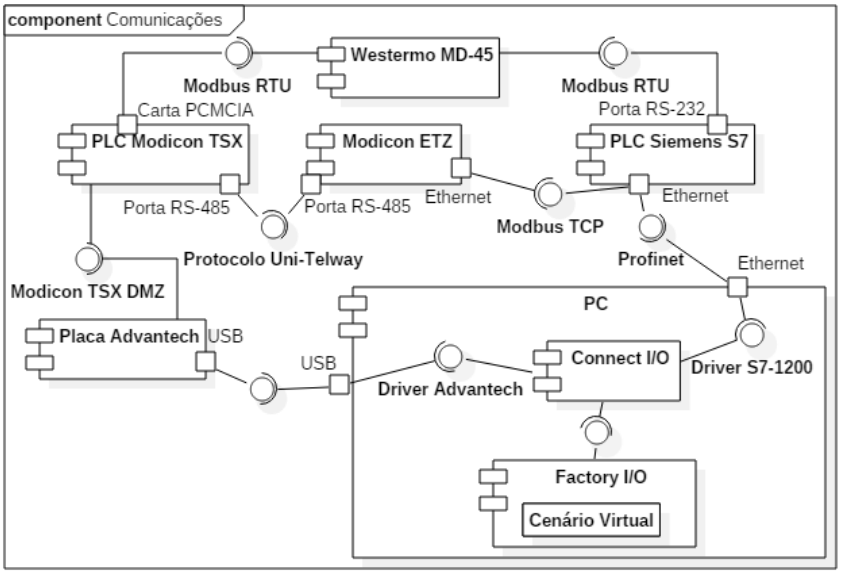

Ilustração 5.3.5: Diagrama de componentes da comunicação entre o cenário e os controladores

5.3.5. Coordenação do Controlo

A quantidade de informação a ser trocada aumentou notavelmente em relação ao cenário anterior. Isto deve-se ao aumento da complexidade do controlo e ao facto do controlador *Siemens* observar as etapas da atividade e os valores dos sensores e da atuação dos tapetes de rolos. Tal como apresentado na Tabela 5.3.4, a etapas 0 e 1 do TSX Micro correspondem aos estados de funcionamento e de não funcionamento. As etapas numeradas a partir de 50, descritas na Tabela 5.3.5, dizem respeito a diferentes estados do encaminhamento das paletes por via da mesa, atividade principal do cenário.

Tabela 5.3.4: Variáveis trocadas entre controladores

Variável	PLC que a determina	Significado
SinalStart	Siemens S7-1200	Início de funcionamento
SinalStop		Paragem de funcionamento
SinalReset		Reiniciação de variáveis
ReceberPalete		Autorização para a mesa receber uma palete
EmitirPalete		Autorização para a mesa emitir uma palete
Prioridade		Tapete alimentador (1 ou 2) que detém a prioridade atual
Destino		Tapete de saída (3 ou 4) da palete a mover de momento
SinalResetConfirm		
Etapa 0	*Modicon* TSX *Micro*	Etapa de não funcionamento
Etapa 1		Etapa de funcionamento
Etapas 50, 51, 52, 55, 60, 65		Etapas da atividade
Sensores 1 a 4		Valores dos sensores retrorrefletivos dos tapetes 1 a 4
Tapetes 1 a 4		Valores da atuação dos tapetes de rolos

As variáveis *ReceberPalete* e *EmitirPalete* correspondem à autorização por parte do S7-1200 para deixar a mesa rotativa receber uma palete ou transferi-la para outro tapete. Isto tem como objetivo impedir que este controlador dê continuação ao encaminhamento, caso se perca a ligação entre ele e o S7-1200. As variáveis *Prioridade* e *Destino* indicam, respetivamente, qual o tapete que detém prioridade e qual o tapete de destino da próxima palete a receber. Todas estas variáveis são do tipo binário.

Tabela 5.3.5: Inserção das etapas na rede de Petri

Etapa:	Significado:	Lugares correspondentes na rede de Petri:
50	Etapa inicial	P7
51	Esperar pela próxima palete a receber	P7
52	A mesa recebe uma palete do tapete 1	P3
55	A mesa recebe uma palete do tapete 2	P6
60	A mesa envia uma palete para o tapete 3	P8; ou P9 e P10
65	A mesa envia uma palete para o tapete 4	P8; ou P9 e P11

As ações de transferência de paletes entre elementos transportadores foram feitas de forma similar ao do caso de estudo anterior, tendo sindo incluídos os sinais atrás mencionados. Essas interações apresentam-se nos diagramas de sequência da Ilustração 5.3.6 e da Ilustração 5.3.7.

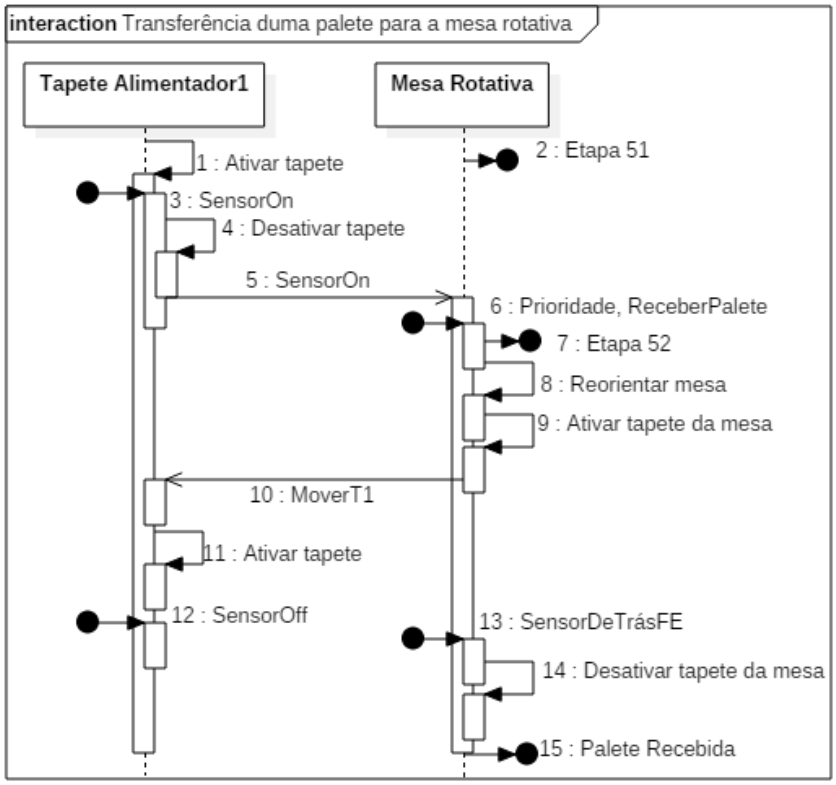

Ilustração 5.3.6: Transferência duma palete do tapete 1 para a mesa rotativa

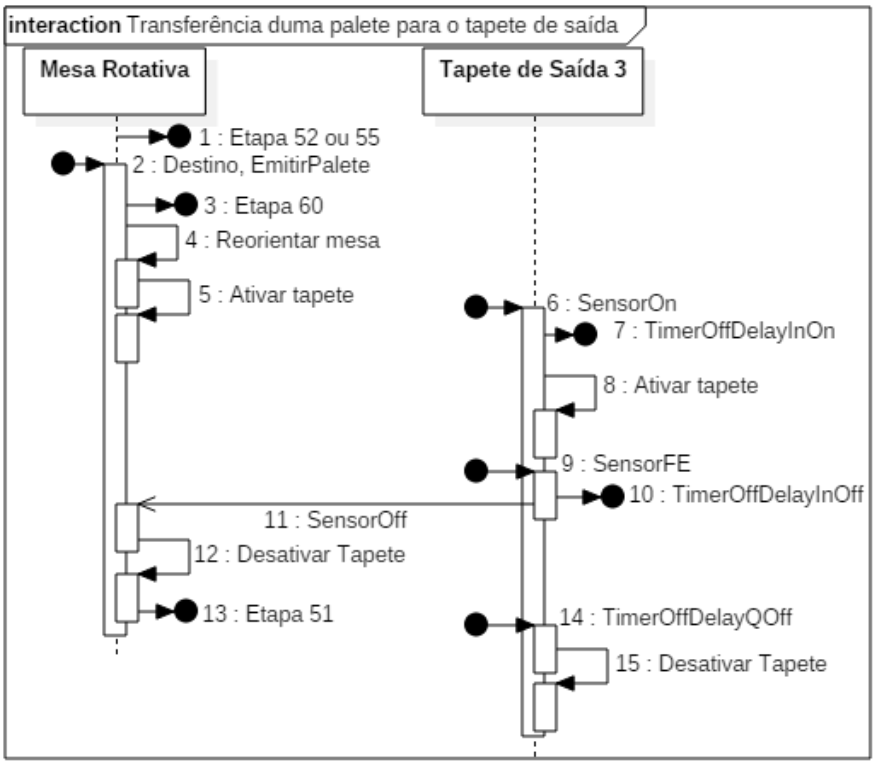

Ilustração 5.3.7: Transferência duma palete da mesa rotativa para o tapete 3

5.3.6. Resultados e Observações

Tal como no caso de estudo anterior, o funcionamento pretendido para o cenário foi executado com sucesso. Desta vez, foi necessário recorrer a um conjunto de *hardware* adicional, constituído pelos módulos, a placa de aquisição e o conversor série, para realizar as comunicações necessárias.

Para utilizar *Modbus* TCP, foi necessário configurar o módulo *Ethernet* TSX ETZ como um dispositivo individual, através dum *browser* dum PC. A falta de atualidade deste dispositivo fez-se notar, na medida em que foi necessário instalar uma versão mais antiga do *software* Java (versão 7.45) para aceder à página da configuração do TSX ETZ.

As duas variantes do protocolo *Modbus* utilizadas, RTU e TCP, não são exclusivas entre si, dado que utilizam diferentes recursos físicos dos dois controladores. Como tal, é possível utilizar ambos os protocolos em simultâneo, obtendo trocas de dados que seriam em princípio mais rápidas. No entanto, este método é menos seguro, visto que a comunicação entre os controladores dependeria de dois meios físicos em vez de apenas um.

Os dois controladores partilharam uma maior quantidade de dados da operação. Apesar de nem toda essa informação ser diretamente necessária para o funcionamento do cenário, a sua partilha facilita a implementação de um possível sistema de supervisão.

5.4. Encaminhador por duas mesas de transferência, com três entradas e três saídas

Neste cenário (Ilustração 5.4.1) encontram-se 8 tapetes de rolos unidos a duas mesas de transferência. Os tapetes alimentadores (Tapetes 1, 2 e 6) emitem paletes sobre as quais estão pousadas peças coloridas. Os dois tapetes intermédios (3 e 5) formam a ligação entre as duas mesas. Os tapetes de saída (4, 7 e 8) constituem os pontos terminais do conjunto.

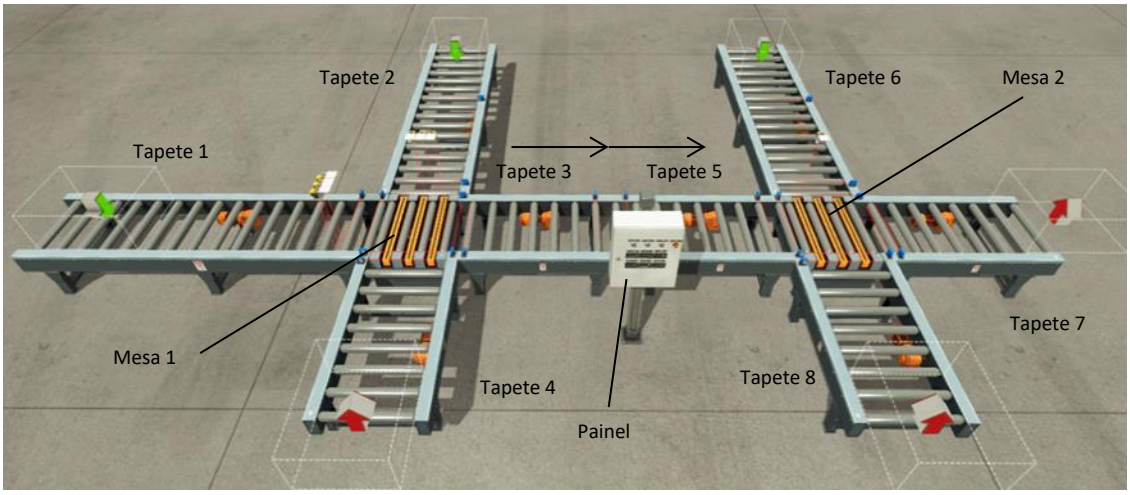

Ilustração 5.4.1: Encaminhador por duas mesas de transferência, com três entradas e três saídas

Cada tapete alimentador foi munido de sensores de visão para permitir a distinção das peças a deslocar. Essa distinção constituirá a base da lógica de encaminhamento a implementar. Cada mesa de transferência foi também equipada com 4 sensores difusivos – um para cada entrada ou saída (Ilustração 5.4.2).

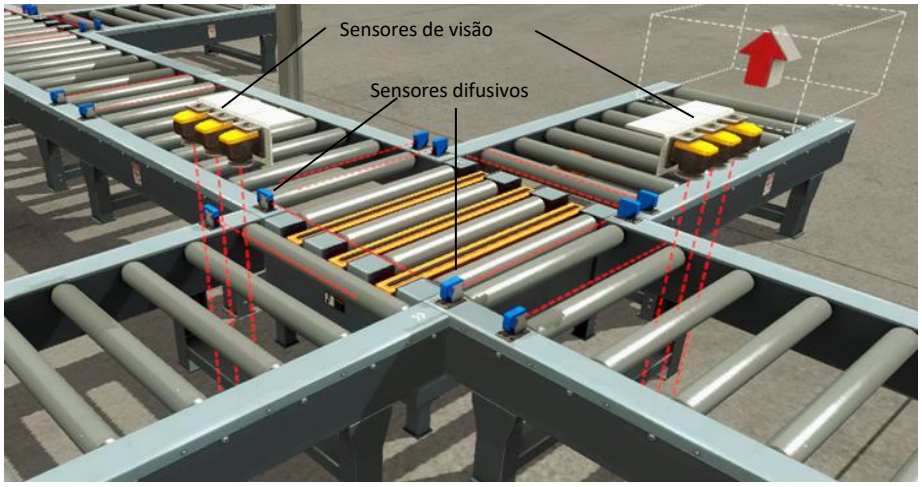

Ilustração 5.4.2: Intersecção de tapetes com mesa de transferência

5.4.1. Comportamento

A Tabela 5.4.1 lista as peças a serem emitidas pelos tapetes alimentadores, assim como o destino pretendido para cada uma delas.

Tabela 5.4.1: Itens transportados pelo cenário

Tipo	Cor	Tapete alimentador	Tapete de saída
Matéria-prima	Azul	Tapete 2	Tapete 4
	Verde	Tapete 1	
Tampa	Azul	Tapetes 2 e 5	Tapete 7
	Verde	Tapetes 1 e 5	
Base	Azul	Tapetes 2 e 5	Tapete 8
	Verde	Tapetes 1 e 5	

A mesa de transferência 1 desloca as peças com a seguinte lógica de encaminhamento: as tampas detêm prioridade sobre as bases e estas detêm-na sobre as matérias-primas. As peças emitidas pelo tapete 2 detêm prioridade sobre o tapete 1. As tampas e as bases terão de passar pelos tapetes 3 e 5 e pela segunda mesa de transferência para atingirem o destino correspondente. Se existirem duas peças a serem transportadas pelos tapetes intermédios 3 e 5, o que constitui o limite de ocupação dos mesmos, a mesa 1 apenas encaminhará matérias-primas para o tapete 4. A mesa 2 oferece sempre prioridade às peças encaminhadas pelos tapetes intermédios, de modo a minimizar os períodos de ocupação dos mesmos.

5.4.2. Modelação por Rede de Petri

Tal como nos cenários anteriores, desenhou-se uma rede de Petri (Ilustração 5.4.3 a)), cuja legenda está apresentada na Tabela 5.4.2. A rede é parcialmente simétrica, podendo ser dividida em dois subconjuntos, um para cada intersecção.

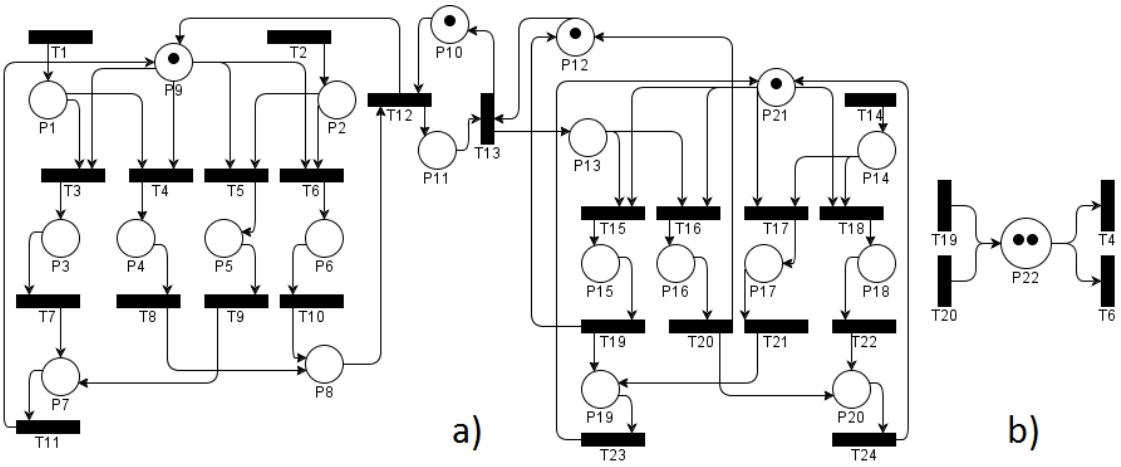

Ilustração 5.4.3: Rede de Petri do cenário virtual

Relativamente aos casos anteriores, tomou-se uma abordagem diferente para representar os diversos atuadores. Os tapetes de entrada 1, 2 e 6, representados pelos lugares P1, P2 e P14, respetivamente, foram definidos sem um limite de capacidade. Por outro lado, os tapetes de saída 4, 7 e 8 não foram ilustrados, presumindo-se que as paletes são garantidamente removidas ao entrarem neles. Por último, os tapetes intermédios 3 e 5, assim como as mesas de transferência foram representadas com um limite de ocupação de uma peça por elemento.

Este limite é imposto através dos lugares P10 e P12, para os tapetes 3 e 5, e P9 e P12, para as mesas 1 e 2, respetivamente: cada atuador pode apenas receber uma palete enquanto estiver desocupado (uma única marca na respetiva posição). A ausência da marca indica a ocupação do respetivo elemento

Tabela 5.4.2: Legenda da rede de Petri

P1	Peças no tapete 1	T1	Entrada duma peça para o tapete 1
P2	Peças no tapete 2	T2	Entrada duma peça para o tapete 2
P3	Peça a ser movida do tapete 1 para a mesa 1 com destino ao tapete 4	T3	Início da passagem de peça do tapete 1 para a mesa 1, com destino ao tapete 4
P4	Peça a ser movida do tapete 1 para a mesa 1 com destino ao tapete 3	T4	Início da passagem de peça do tapete 1 para a mesa 1, com destino ao tapete 3
P5	Peça a ser movida do tapete 2 para a mesa 1 com destino ao tapete 4	T5	Início da passagem de peça do tapete 2 para a mesa 1, com destino ao tapete 4
P6	Peça a ser movida do tapete 2 para a mesa 1 com destino ao tapete 3	T6	Início da passagem de peça do tapete 2 para a mesa 1, com destino ao tapete 3
P7	Peça a ser movida da mesa 1 para o tapete 4	T7	Peça sai do tapete 1 e começa a ser movida para o tapete 4
P8	Peça a ser movida da mesa 1 para o tapete 3	T8	Peça sai do tapete 1 e começa a ser movida para o tapete 3
P9	Mesa de transferência 1 desocupada	T9	Peça sai do tapete 2 e começa a ser movida para o tapete 4
P10	Tapete 3 desocupado	T10	Peça sai do tapete 2 e começa a ser movida para o tapete 3
P11	Tapete 3 ocupado por uma peça	T11	Peça sai da mesa 1 para o tapete 4
P12	Tapete 5 desocupado	T12	Peça sai da mesa 1 para o tapete 3
P13	Tapete 5 ocupado por uma peça	T13	Peça sai do tapete 3 para o tapete 5
P14	Peças no tapete 6	T14	Entrada duma peça para o tapete 6
P15	Peça a ser movida do tapete 5 para a mesa 2 com destino ao tapete 7	T15	Início da passagem de peça do tapete 5 para a mesa 2, com destino ao tapete 7
P16	Peça a ser movida do tapete 5 para a mesa 2 com destino ao tapete 8	T16	Início da passagem de peça do tapete 5 para a mesa 2, com destino ao tapete 8
P17	Peça a ser movida do tapete 6 para a mesa 2 com destino ao tapete 7	T17	Início da passagem de peça do tapete 6 para a mesa 2, com destino ao tapete 7
P18	Peça a ser movida do tapete 6 para a mesa 2 com destino ao tapete 8	T18	Início da passagem de peça do tapete 6 para a mesa 2, com destino ao tapete 8
P19	Peça a ser movida da mesa 2 para o tapete 7	T19	Peça sai do tapete 5 e começa a ser movida para o tapete 7
P20	Peça a ser movida da mesa 2 para o tapete 8	T20	Peça sai do tapete 5 e começa a ser movida para o tapete 8
P21	Mesa de transferência 2 disponível	T21	Peça sai do tapete 6 e começa a ser movida para o tapete 7
P22	Disponibilidade dos tapetes 3 e 5	T22	Peça sai do tapete 6 e começa a ser movida para o tapete 8
		T23	Peça sai da mesa 2 para o tapete 7
		T24	Peça sai da mesa 3 para o tapete 8

Tomando como exemplo o tapete 3, este só pode receber uma peça estando desocupado (P10). Ao receber uma peça (T12), passa a estar ocupado por ela (P11). Só voltará a ter a disponibilidade de receber a peça seguinte quando a peça atual ter sido movida para o tapete 5 (T13).

Os lugares associados à deslocação de uma palete desde um tapete para uma das mesas foram distinguidos pelos tapetes de origem e de destino, obtendo-se 4 lugares por mesa. Isto é devido à imposição do requisito da mesa 1 não poder tentar mover uma peça direcionada à mesa 2 estando os tapetes 3 e 5 ocupados.

Para representar esta restrição na rede, desenhou-se uma rede de Petri adicional (Ilustração 5.4.3 b)), que a impõe. Esta rede, composta pelo lugar P22 foi obtida recorrendo ao método proposto por (47). De cada vez que as transições T4 ou T6 forem disparadas, que dizem respeito a peças que passarão pelos tapetes 3 e 5, o lugar P22 perde uma das suas duas marcas iniciais. P22 apenas recupera essas marcas após o disparo das transições T19 ou T20, que assinalam a libertação do tapete 5.

O conjunto das duas redes de Petri é reversível e todas as transições são vivas, confirmando a capacidade do cenário correr de forma contínua e sem situações de *deadlock*.

5.4.3. Distribuição do Controlo Lógico

Os controladores utilizados foram um *SoftPLC Codesys Control Win* V3 e dois PLCs *Siemens*, o S7-1200 e o S7-200.

Tabela 5.4.3: Variáveis trocadas entre o cenário e os controladores

Variáveis de entrada:	Tipo de variável:	Controlador:
Factory.Run	Binária	*Siemens* S7-1200
Factory.Pause		
Factory.Reset		
Sensores difusivos das mesas de transferência	8 variáveis binárias (4 por mesa)	
Sensores difusivos dos tapetes 1, 2, 3 e 4	4 variáveis binárias	*Siemens* S7-200
Sensores de visão dos tapetes 1 e 2	6 variáveis binárias	
Sensores difusivos dos tapetes 5, 6, 7 e 8	4 variáveis binárias	*Codesys Control Win* V3
Sensor de visão do tapete 5	Inteira	
Botão *Start*	Binária	*Siemens* S7-1200
Botão *Reset*		
Botão *Stop*		
Botão de Emergência		
Variáveis de saída:	**Tipo de variável:**	**Controlador:**
Tapetes de rolos 1 a 4	4 variáveis binárias	*Siemens* S7-200
Tapetes de rolos 5 a 8	4 variáveis binárias	*Codesys Control Win* V3
Mesa de transferência 1 e 2 – rolos e correias – sentidos positivo e negativo	8 variáveis binárias (4 por mesa)	*Siemens* S7-1200
Luz do botão *Start*	Binária	
Luz do botão *Reset*		
Luz do botão *Stop*		
Mostradores 1 a 6	6 variáveis inteiras	

O controlo foi distribuído do seguinte modo: os tapetes 1 a 4 foram controlados pelo *Siemens* S7-200, os restantes tapetes pelo *SoftPLC Codesys* e as mesas de transferência pelo S7-1200. Este último também foi responsável por gerir a lógica do encaminhamento das peças. Os sensores associados a cada atuador foram lidos pelo respetivo controlador.

Na Tabela 5.4.3 são listadas as variáveis trocadas entre o ambiente virtual e os PLCs.

5.4.4. Implementação da Comunicação

A comunicação entre o *SoftPLC Codesys* e o *Siemens* S7-1200 foi feita por *Modbus* TCP, sendo o S7-1200 o cliente da ligação. Os dois PLCs da *Siemens*, o S7-200 e o S7-1200, trocaram dados por comunicação série sem protocolo, sobre uma ligação RS-485.

Dado ter-se optado este último modo de comunicação, o tratamento dos dados envolvidos teve que ser implementado com mais cuidado dentro dos programas de cada controlador, enquanto nos cenários anteriores bastou mapear as variáveis aos respetivos endereços *Modbus*. A comunicação série entre os controladores S7-200 e S7-1200 é realizada em modo half-duplex. Ao fim de uma transmissão, o equipamento transmissor prepara-se para receber uma mensagem e o recetor inicia a transmissão seguinte.

Dado tratar-se de envio de bits sem protocolo, o emissor não recebe confirmação de que a mensagem foi recebida. O envio de uma mensagem por um aparelho iniciado quando uma mensagem é recebida com sucesso. Para evitar a perda de comunicação caso uma mensagem seja mal recebida, S7-1200 foi programado para começar uma nova transmissão passado algum tempo sem receber resposta do S7-200.

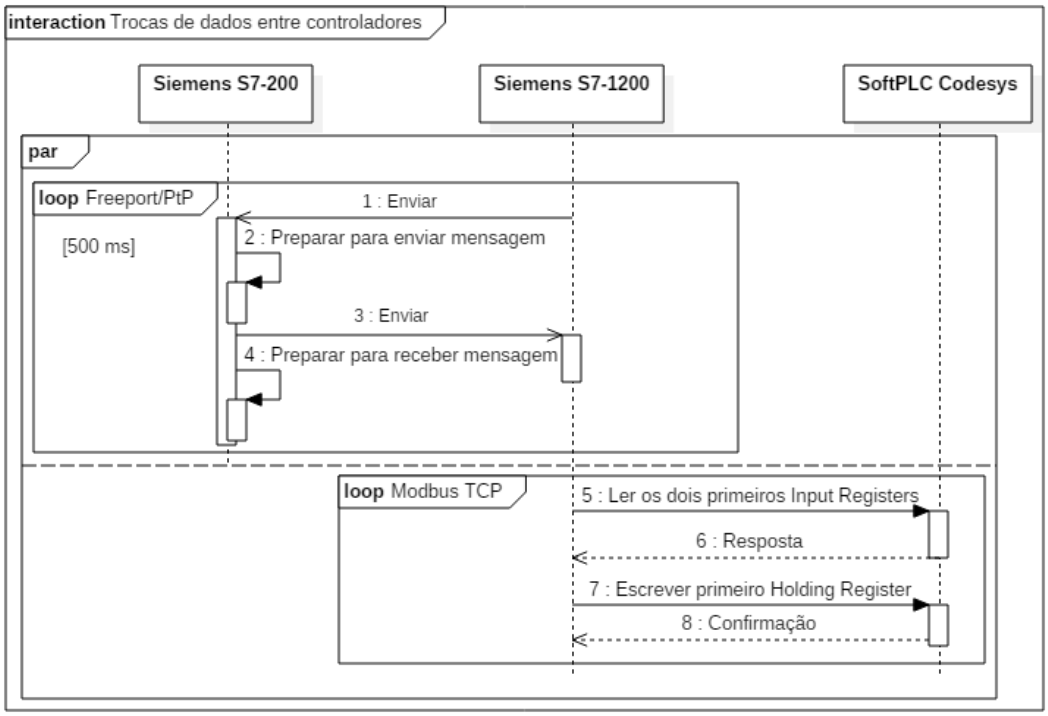

Ilustração 5.4.4: Diagrama de sequência da comunicação entre controladores

O diagrama de sequência da Ilustração 5.4.4 representa as trocas de dados entre os diversos controladores. Os dois tipos de comunicação foram feitos de modo concorrente e independente. O diagrama demonstra também a forma como cada mensagem trocada pelo S7-200 desencadeia a alternância entre os papéis de transmissor e recetor deste PLC.

As interfaces no diagrama de componentes da Ilustração 5.4.5 apresentam-se similares às que foram executadas para o cenário da secção 5.2, com a adição do PLC *Siemens* S7-200, cujas ligações são feitas por RS-485, com o S7-1200, e por intermédio da placa de aquisição *Advantech*, com o *Factory I/O*.

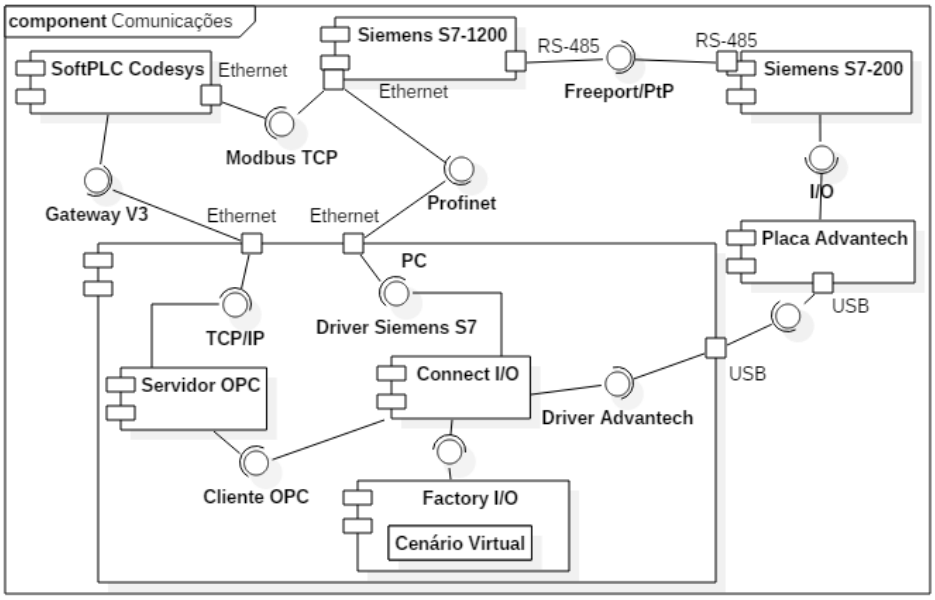

Ilustração 5.4.5: Diagrama de componentes da comunicação entre o cenário e os controladores

5.4.5. Coordenação do Controlo

De modo idêntico ao cenário apresentado em 5.3, adotou-se o controlo com as funções de *Reset* e de Segurança. O controlador S7-1200 agiu como coordenador dos restantes PLCs. Para o controlo dos conjuntos de atuadores, adotou-se os modelos apresentados dos tapetes alimentadores, de saída e intermédios e da interseção com elementos transportadores (secções 4.3.1 e 4.3.3).

As variáveis trocadas entre controladores, listadas na Tabela 5.4.4, incluem os sinais de ordem de movimentação dos tapetes alimentadores e intermédios. O *Siemens* S7-1200 também recebe os valores dos sensores de visão, de modo a executar a lógica de encaminhamento, e dos sensores de entrada dos tapetes intermédios, para poder verificar a transferência de paletes da parte destes.

As atividades de transferência de paletes entre transportadores para este cenário são simplificadas, em relação aos casos anteriores, pelo facto das mesas de transferência não necessitarem de alterar a sua orientação.

Tabela 5.4.4: Variáveis trocadas entre controladores

Variável:	Tipo:	PLC emissor:	PLC recetor:	Significado:
SinalStart	Binária	*Siemens* S7-1200	S7-200, *Codesys*	Início de funcionamento
SinalStop				Paragem de funcionamento
SinalMoverT1			*Siemens* S7-200	Transferir palete do tapete 1 para a mesa de transferência
SinalMoverT2				Transferir palete do tapete 2 para a mesa de transferência
SinalMoverT3				Transferir palete do tapete 3 para o tapete 5
SinalMoverT5			*Codesys Control Win* V3	Transferir palete do tapete 5 para a mesa de transferência 2
SinalMoverT6				Transferir palete do tapete 6 para a mesa de transferência 2
VS1…6		*Siemens* S7-200	*Siemens* S7-1200	Sensores de visão dos tapetes 1 e 2
DS 14				Sensor de entrada do tapete 3
VS7	Inteira	*Codesys Control Win* V3		Sensor de visão do tapete 6
DS 16	Binária			Sensor difusivo de entrada do tapete 5

5.4.6. Resultados e Observações

Apesar de ter sido utilizado um controlador adicional, relativamente aos casos anteriores, o cenário mostrou o desempenho pretendido. A lógica de encaminhamento foi devidamente cumprida.

Para este caso de estudo, a maior dificuldade consistiu na implementação da comunicação série entre os dois controladores *Siemens* S7, através de comunicação série sem protocolo. Para isso, foi necessário configurar a comunicação pormenorizadamente, de modo a observar-se uma perfeita consistência nas trocas de informação. Este modo mostrou potencial de desenvolvimento para possíveis aplicações de maior complexidade.

Como os controladores implementados necessitavam de comunicar continuamente entre si para prosseguir com a atividade do cenário, é perda repentina das comunicações não provocaria uma situação de obstrução dos elementos transportadores.

5.5. Tapetes Transportadores Paralelos com Dois Manipuladores *Pick and Place*

Este cenário (Ilustração 5.5.1) apresenta dois troços de tapetes de tela sobre os quais são deslocadas peças. Cada troço é constituído por um tapete alimentador (tapetes 1 e 3), que inclui um sensor de visão, e por um tapete intermédio (tapetes 2 e 4), que conduz as peças até uma rampa no fim da qual as peças são eliminadas.

Entre os tapetes intermédios, encontra-se uma balança sobre a qual as peças podem ser pousadas, funcionando como um *buffer*. Existem também dois manipuladores robóticos *Pick and Place*, de dois eixos cartesianos, um por troço de tapetes, que permite a deslocação de peças entre um tapete e a balança. Existe também um sensor difusivo debaixo de cada manipulador, sobre o tapete intermédio correspondente.

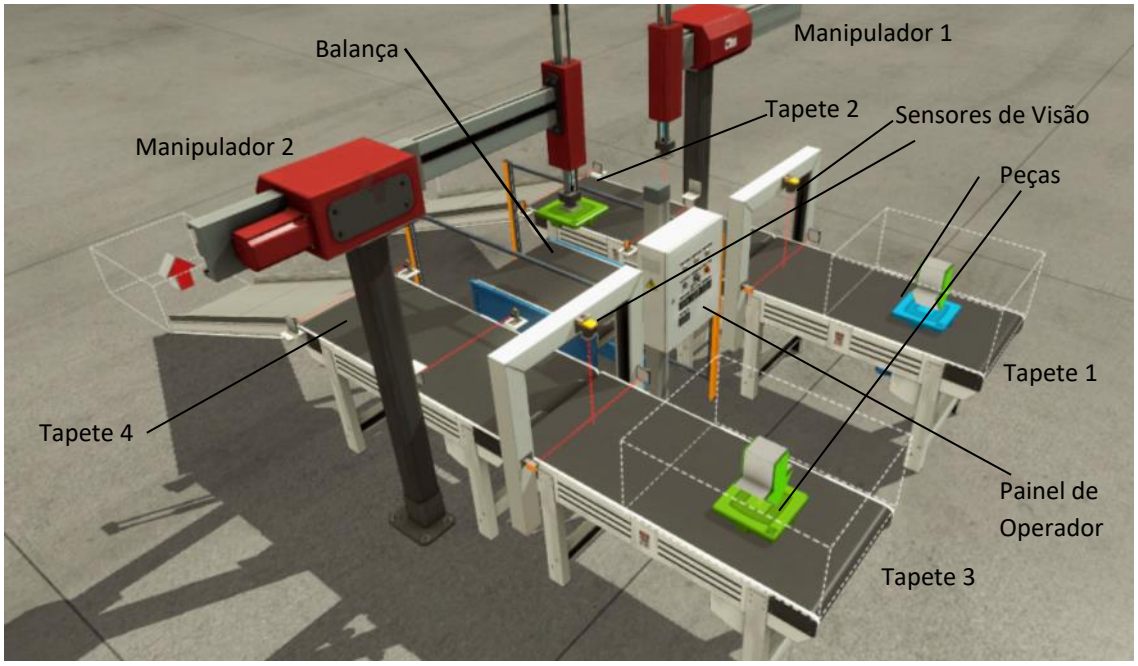

Ilustração 5.5.1: Tapetes transportadores paralelos com dois manipuladores *Pick and Place*

A configuração física deste cenário, assim como o sentido do fluxo das peças, é representada de modo simplificado na Ilustração 5.5.2. Na figura destacam-se três troços constituídos por conjuntos de atuadores: o troço 1 é constituído pelos tapetes 1 e 2 no lado esquerdo da figura, o troço 2 pelos tapetes 3 e 4, componentes análogos do lado direito, e o troço 3 compreende os tapetes 2 e 4, os manipuladores Pick and Place e a balança. Enquanto os dois primeiros troço apenas permitem a deslocação de peças num sentido, o terceiro faz a ligação entre os anteriores em ambos os sentidos.

A existência deste terceiro troço permite a transferência de peças entre os troços 1 e 2 e constitui um aspeto central do exercício praticado neste caso de estudo.

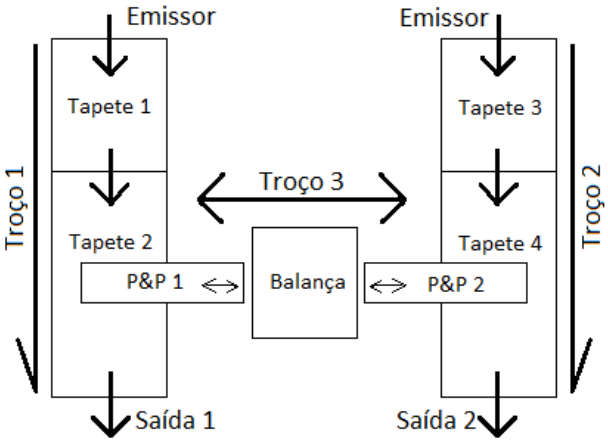

Ilustração 5.5.2: Disposição simplificada dos atuadores

O cenário inclui ainda um painel de operador (Ilustração 5.5.3). Para além dos botões e mostradores típicos inseridos nos painéis dos cenários anteriores, foram adicionados um seletor de duas posições e um sinalizador azul.

Ilustração 5.5.3: Painel de operador

5.5.1. Comportamento

Neste caso de estudo, os tapetes alimentadores irão emitir peças coloridas, que serão encaminhadas para uma das rampas 1 e 2, dependendo do tipo de peça. O sentido do encaminhamento de cada peça é representado na Tabela 5.5.1.

Tabela 5.5.1: Trajeto de cada tipo de peça

Peça:	Origem:	Destino:	Trajeto:
Base azul	Tapete 1	Saída 1	Troço 1
Tampa azul		Saída 2	Troços 1, 3, 2
Base verde	Tapete 2	Saída 1	Troço 2
Tampa verde		Saída 2	Troços 2, 3, 1

Quando uma peça começa a ser movida, ela é identificada por um sensor de visão. Dependendo do tipo de peça, esta será encaminhada diretamente para a rampa de saída no final do troço de onde originou ou transportada para a outra rampa. Neste segundo caso, a peça é movimentada através do troço 3 por via dos dois manipuladores *Pick and Place*. Através do troço 3, só pode ser movida uma peça de cada vez. Se existir uma peça em cada tapete de entrada, destinada a ser movida neste troço, uma delas será movida enquanto a outra espera que os manipuladores voltem a estar disponíveis.

Nesta situação, a peça a ser movida primeiro é decidida com base na prioridade atribuída aos tapetes. O tapete que detém prioridade é escolhido através do seletor localizado no painel. Os mostradores revelam a contagem de cada tipo de peça emitido. O sinalizador azul encontra-se aceso sempre que se dá a transferência duma peça por via do troço 3.

5.5.2. Modelação por Rede de Petri

Na Ilustração 5.5.4 a) está apresentada a rede de Petri representativa deste cenário. A legendagem dos lugares e das transições está apresentada na Tabela 5.5.2. A rede está dividida em 4 troços; cada um representa um dos trajetos possível segundo o qual são transportadas as peças.

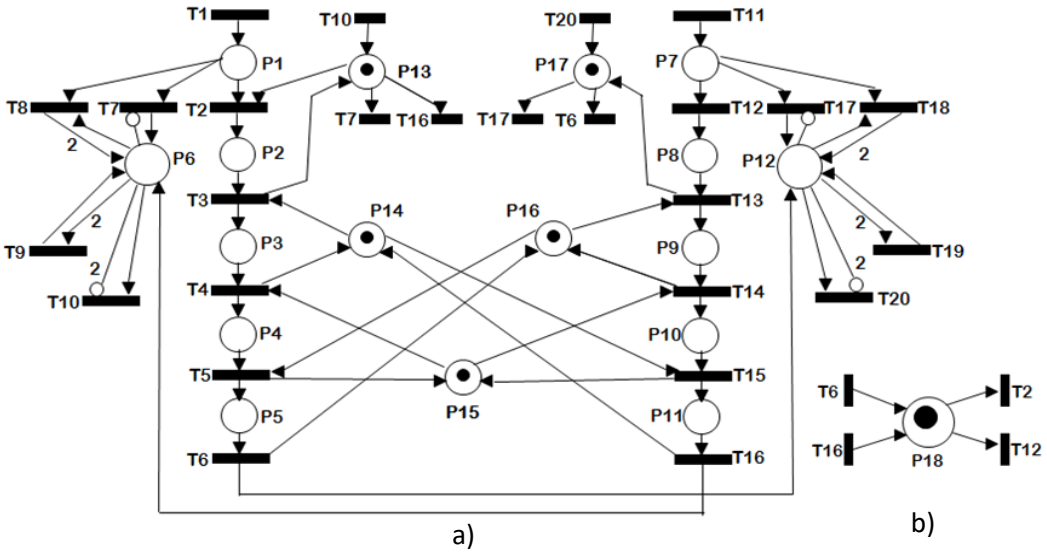

Ilustração 5.5.4: Rede de Petri do cenário virtual

Os conjuntos de lugares desde P2 a P5 e desde P8 a P11 representam peças que são movimentadas sobre o troço 3. O primeiro diz respeito à movimentação no sentido desde o tapete 2 até ao tapete 4, enquanto outro diz respeito à deslocação no sentido oposto.

Os lugares P6 e P12 representam as peças que se encontram nos tapetes 2 e 4, respetivamente, e que são diretamente dirigidas para a rampa de saída de cada tapete. Essas peças incluem aquelas que acabam de ser trazidas a um dos tapetes pelos manipuladores *Pick and Place*. A rede não impõe um limite para o número de peças que podem ocupar esses tapetes, mas no cenário executado esse valor não chegou a superar as duas peças.

Os lugares P13 a P17 representam a disponibilidade dos elementos envolvidos na deslocação de peças entre os tapetes 2 e 4, inclusive. Essas posições possuem, na marcação inicial, uma marca cada uma, marca essa que é perdida sempre o elemento está a ser utilizado e recuperada quando o elemento volta a ficar livre.

A posição P18 da Ilustração 5.5.4 b), inicialmente com e limitada a uma marca, representa se se efetua ou não a transferência duma peça sobre o troço 3. Quando essa atividade se inicia (disparo duma das transições T2 ou T12), o lugar perde a sua marca, inibindo o disparo repetido dessas mesmas transições. Dessa forma, impede-se que ocorra a movimentação simultânea de duas peças em sentidos opostos, assim como um eventual impasse do sistema.

Tabela 5.5.2: Legendagem dos lugares e transições da rede de Petri

P1	Presença duma peça no tapete 1	T1	Deteção duma peça no tapete 1
P2	Peça no tapete 2 espera ser levantada pelo manipulador 1	T2	Passagem duma peça dirigida à saída 2 do tapete 1 para o tapete 2
P3	Peça a ser levada até à balança pelo manipulador 1	T3	Manipulador 1 pega numa peça sobre o tapete 2
P4	Peça dirigida ao tapete 4 encontra-se pousada na balança	T4	Manipulador 1 pousa uma peça sobre a balança
P5	Peça dirigida ao tapete 4 a ser deslocada pelo manipulador 2	T5	Manipulador 2 pega numa peça sobre a balança
P6	Número de peças no tapete 2 dirigidas à saída 1	T6	Manipulador 2 pousa uma peça sobre o tapete 4
P7	Presença duma peça no tapete 3	T7	Passagem duma peça dirigida à saída 1 do tapete 1 para o tapete 2; tapete 2 anteriormente vazio
P8	Peça no tapete 4 espera ser levantada pelo manipulador 2	T8	Passagem duma peça dirigida à saída 1 do tapete 1 para o tapete 2; tapete 2 anteriormente ocupado
P9	Peça a ser levada até à balança pelo manipulador 2	T9	Saída duma peça do tapete 2 para a saída 1; tapete 2 permanece ocupado
P10	Peça dirigida ao tapete 2 encontra-se pousada na balança	T10	Saída duma peça do tapete 2 para a saída 1; tapete 2 fica vazio
P11	Peça dirigida ao tapete 2 a ser deslocada pelo manipulador 1	T11	Deteção duma peça no tapete 3
P12	Número de peças no tapete 4 dirigidas à saída 2	T12	Passagem duma peça dirigida à saída 4 do tapete 3 para o tapete 4
P13	Tapete 2 disponível	T13	Manipulador 2 pega numa peça sobre o tapete 4
P14	Manipulador 1 disponível	T14	Manipulador 2 pousa uma peça sobre a balança
P15	Balança disponível	T15	Manipulador 1 pega numa peça sobre a balança
P16	Manipulador 2 disponível	T16	Manipulador 1 pousa uma peça sobre o tapete 2
P17	Tapete 4 disponível	T17	Passagem duma peça dirigida à saída 2 do tapete 3 para o tapete 4; tapete 2 anteriormente vazio
P18	Troço 3 disponível	T18	Passagem duma peça dirigida à saída 2 do tapete 3 para o tapete 4; tapete 2 anteriormente ocupado
		T19	Saída duma peça do tapete 4 para a saída 2; tapete 4 permanece ocupado
		T20	Saída duma peça do tapete 4 para a saída 2; tapete 4 fica vazio

As transições T7 e T8 distinguem-se na medida em que a primeira representa uma base azul que entra no tapete 2, estando este anteriormente vazio, e a segunda diz respeito a uma peça idêntica que entre no tapete 2 quando esse já estava ocupado. As duas transições foram distinguidas uma da outra pela utilização dum arco inibidor. Um arco inibidor indica que uma dada transição só pode ser disparada quando a posição a montante tiver um número de marcas inferior ao peso do arco. As transições T9 e T10 são similares a T7 e T8, mas representam as peças que saem do tapete 2. As transições T17 a T20 são análogas às transições T7 a T10, dizendo respeito aos tapetes 3 e 4 e a tampas verdes.

A rede conserva as propriedades de reversibilidade e vivacidade que se observaram nos casos de estudo anteriores. A existência do lugar P18 torna-a livre de impasses e colisões.

5.5.3. Distribuição do Controlo Lógico

O controlo do ambiente virtual foi divido entre dois *SoftPLCs Codesys*, o *Codesys Control Win* V3 e o *Codeys SP PLCWinNT* V2.4.

Tabela 5.5.3: Variáveis trocadas entre o cenário e os controladores

Variáveis de entrada:	Tipo de variável:	Controlador:
FactoryRun	Binária	*Codesys Control Win* V3
FactoryPause		
FactoryReset		
Botão *Start*		
Botão *Reset*		
Botão *Stop*		
Botão de Emergência		
Seletor		
Sensores retrorrefletivos 1 a 3	Três variáveis binárias	
Sensor de visão 1	Inteira	
Sensor do Manipulador 1	Binária	
Estado de moção dos eixos do Manipulador 1		
Posição dos eixos do Manipulador 1	Duas variáveis reais	
Balança	Real	
Sensores retrorrefletivos 4 a 6	Três variáveis binárias	*Codeys SP PLCWinNT* V2.4
Sensor de visão 2	Inteira	
Sensor do Manipulador 2	Binária	
Movimento dos eixos do Manipulador 2		
Eixos X e Z do Manipulador 2	Duas variáveis reais	
Variáveis de saída:	**Tipo de variável:**	**Controlador:**
Luz do Botão *Start*	Binária	*Codesys Control Win* V3
Luz do Botão *Reset*		
Luz do Botão *Stop*		
Sinalizador Azul		
Mostradores 1 a 4	Quatro variáveis inteiras	
Emissores	Duas variáveis binárias	
Tapetes 1 e 2		
Ventosa do Manipulador 1	Binária	
Referência dos eixos do Manipulador 1	Duas variáveis reais	
Tapetes 3 e 4	Binária	*Codeys SP PLCWinNT* V2.4
Ventosa do Manipulador 2		
Referência dos eixos do Manipulador 2	Duas variáveis reais	

O controlo dos atuadores foi repartido entre os dois *SoftPLCs* de forma simétrica: cada controlador ficou encarregue do controlo de um dos troços de tapetes, de um dos manipuladores *Pick and Place* e os sensores associados a cada atuador. Para além disso, a balança foi monitorizada pelo *SoftPLC Codesys* V3. A interação com o painel de operador, assim como a coordenação do controlo foi atribuída ao *SoftPLC Codesys* V3. A Tabela 5.5.3 lista as variáveis comunicadas entre o cenário e os controladores.

5.5.4. Implementação da Comunicação

Os dois *SoftPLCs Codesys* realizaram a comunicação entre si através da partilha de variáveis de rede sobre UDP (Ilustração 5.5.5). As variáveis são originadas no controlador que as emite sobre uma rede Ethernet através de mensagens UDP/IP, enquanto o outro *SoftPLC* é configurado para ler essas variáveis.

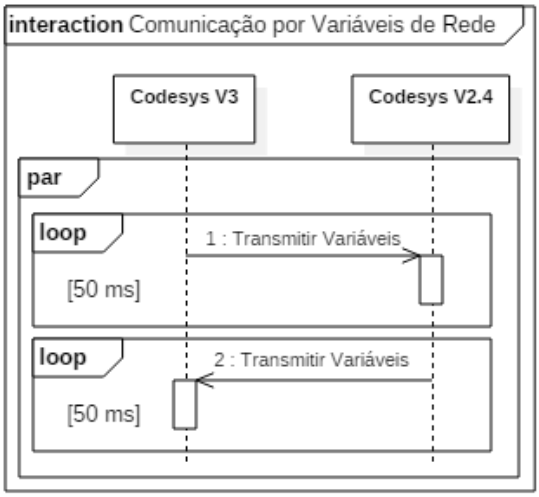

Ilustração 5.5.5: Digrama de sequência da comunicação por variáveis de rede

A implementação física das comunicações é ilustrada pelo diagrama de componentes da Ilustração 5.5.6. A comunicação por variáveis de rede é representada duas vezes, pois cada controlador define o seu próprio de variáveis a transmitir. A interação com o cenário foi feita por ligação simultânea dos dois controladores ao mesmo servidor OPC DA.

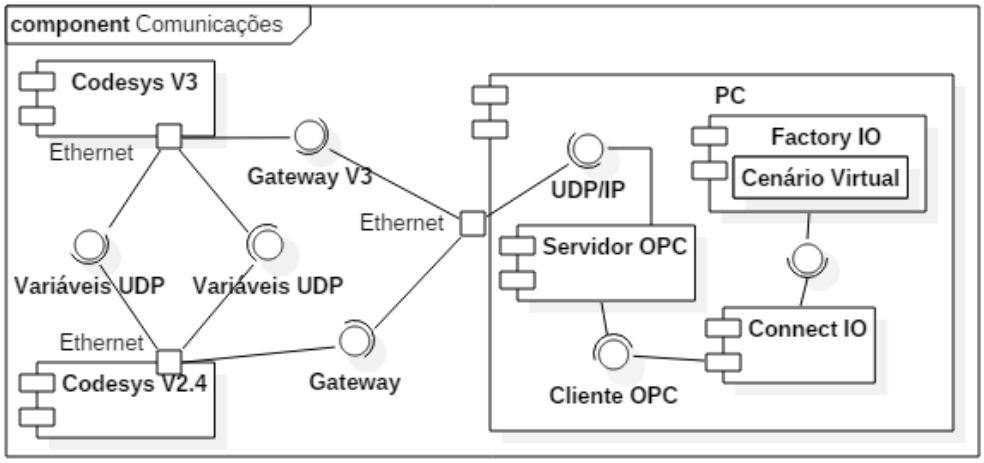

Ilustração 5.5.6: Diagrama de componentes da comunicação entre o cenário e os controladores

5.5.5. Coordenação do Controlo

Os dados trocados entre os *SoftPLCs* e o seu significado encontram-se na Tabela 5.5.4.

Para além dos sinais de controlo cooperativo definidos para os casos de estudo anteriores, os dois controladores trocam dados relativos ao transporte de peças entre os dois tapetes intermédios, 2 e 4, feito através dos dois manipuladores *Pick and Place*. Estes dados são essenciais para que o controlador *Codesys* V3 possa devidamente monitorizar o encaminhamento das peças, impedindo a colisão de equipamentos durante o transporte por via dos manipuladores *Pick and Place*.

Tabela 5.5.4: Variáveis trocadas entre controladores

Variável:	Tipo de variável:	PLC que a transmite:	Significado:
SinalStart	Binária	*Codesys Control Win* V3	Início de funcionamento
SinalStop			Paragem de funcionamento
SinalReset			Reinício das variáveis
Prioridade			Qual o troço que detém prioridade
Send2to4Req			Pedido de transporte de peça desde o tapete 2 para o tapete 4
Send2to4On			Peça encontra-se a ser transportada desde o tapete 2 para o tapete 4
Send4to2Go			Autorização para transportar peça desde o tapete 4 para o tapete 2
Send4to2End			Fim do transporte duma peça desde o tapete 4 até ao tapete 2
ScaleOn			Existência duma peça sobre a balança
PP1Extend			O manipulador 1 encontra-se a estender-se ou completamente estendido
ResetConfirm		*Codeys SP PLCWinNT* V2.4	Confirmação do reinício das variáveis
Send4to2Req			Pedido de transporte de peça desde o tapete 4 para o tapete 2
Send4to2On			Peça encontra-se a ser transportada desde o tapete 4 para o tapete 2
Send2to4End			Fim do transporte duma peça desde o tapete 2 até ao tapete 4
PP2Extend			O manipulador 2 encontra-se a estender-se ou completamente estendido
LidGreen	Inteira		Contagem das tampas verdes encaminhadas
LidBlue			Contagem das tampas azuis encaminhadas

Nos seguintes diagramas de sequência, é representada a transferência duma peça desde o tapete 1 até ao tapete 4, dividida em duas fases: a transferência feito pelo manipulador 1 (Ilustração 5.5.7) e a transferência pelo manipulador 2 (Ilustração 5.5.8), indicando-se devidamente o papel das variáveis definidas.

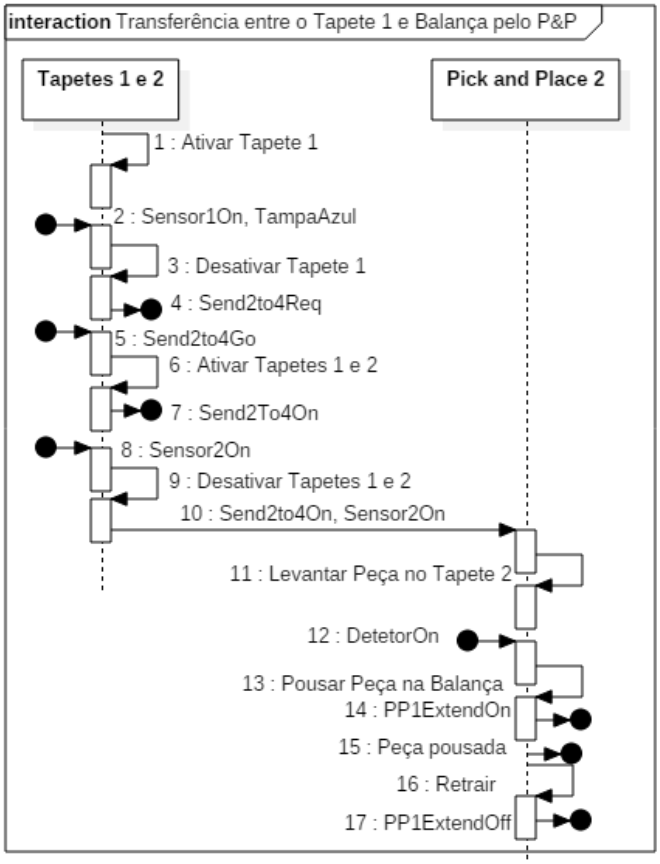

Ilustração 5.5.7: Transferência entre o Tapete 1 e a balança pelo manipulador 1

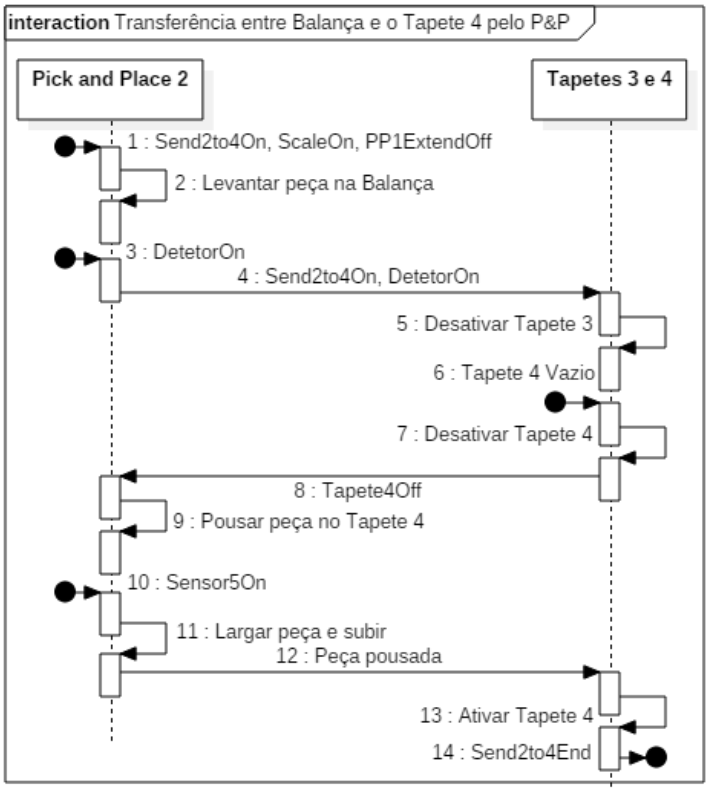

Ilustração 5.5.8: Transferência entre a balança e o tapete 4 pelo manipulador 2

5.5.6. Resultados e Observações

Os componentes do cenário virtual cumpriram a sua função como pretendido. O troço compreendido pelos manipuladores nunca chegou a estar ocupado por mais do que uma peça, o que indica que as restrições impostas pela rede de Petri foram implementadas com sucesso.

A comunicação entre os dois *SoftPLCs Codesys* por variáveis de rede foi desempenhada com facilidade, como seria de esperar, dado serem componentes de *software* da mesma marca.

Este caso de estudo pode ser desenvolvido com a inclusão duma ordem de movimentação mais complexa ou alterável por via de botões ou seletores adicionáveis ao painel de controlo. Podem também ser adicionados mais troços de tapetes transportadores ou manipuladores de modo a criar mais possíveis trajetos de encaminhamento para as peças.

Uma outra possível evolução deste caso de estudo seria a aplicação do encaminhamento simultâneo de várias peças pelos manipuladores *Pick and Place*, com o mesmo sentido de movimentação, sem que se dê a colisão dos equipamentos. Este comportamento é fisicamente possível no ambiente criado, mas não permitido pela rede de Petri especificada.

5.6. Encaminhador por Dois Separadores a Rodízios com Cinco Entradas e uma Saída

Este cenário (Ilustração 5.6.1) é constituído por 5 conjuntos de tapetes (1 a 5) que transportam caixas com destino a um único tapete de saída (tapete 6). Os tapetes de entrada 1, 2 e 3 terminam num cruzamento constituído por um separador a rodízios que encaminha as caixas para um tapete intermédio (tapete 7). Esse tapete, assim como os tapetes 4 e 5 conduzem a um segundo separador que envia as caixas para o tapete de saída.

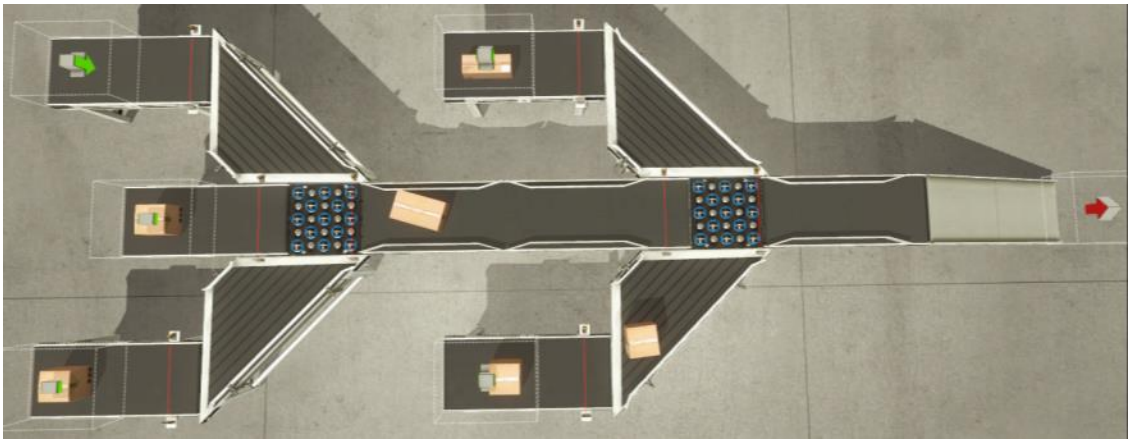

Ilustração 5.6.1: Encaminhador por dois separadores a rodízios com cinco entradas e uma saída

Em termos de configuração física (Ilustração 5.6.2), este caso de estudo é similar aos primeiros cenários desenvolvidos, nos quais se fazia o reencaminhamento de itens por via de mesas transportadoras. Este caso apresenta a particularidade de todas as entradas conduzirem a uma única saída.

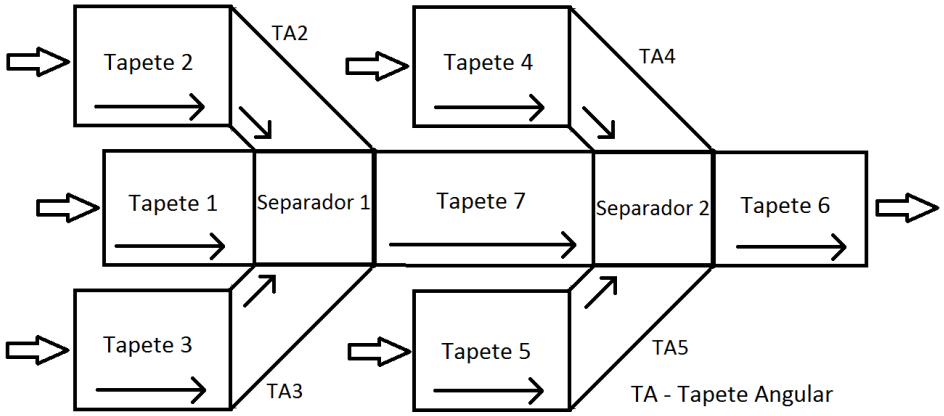

Ilustração 5.6.2: Esquema dos elementos transportadores do cenário

5.6.1. Comportamento

O funcionamento desejado para este caso de estudo é fazer com que todas as caixas convirjam para o mesmo tapete de saída. Pretendeu-se implementar um sistema de seleção baseado em filas de espera do tipo FIFO (*First In First Out*), e não em prioridades pré-estabelecidas como se fez com os casos de estudo anteriores.

Cada interseção, constituída pelos separadores e pelos tapetes angulares de tela, terá portanto a sua própria fila de espera, recebendo uma caixa de cada vez em ordem de chegada.

5.6.2. Modelação por Rede de Petri

A rede de Petri representativa deste cenário virtual encontra-se na Ilustração 5.6.3 e a sua legenda na Tabela 5.6.1. Os tapetes alimentadores podem conter várias caixas em simultâneo, podendo apenas enviar uma para os atuadores seguintes de cada vez. Os restantes atuadores (tapetes e separadores) apenas podem transportar uma caixa num dado momento, pelo que cada um possui dois lugares: um para o estado livre (inicial) e outro para o estado ocupado. O tapete de saída é representado pela transição T17, pressupondo-se que as caixas são garantidamente eliminadas ao entrar nele.

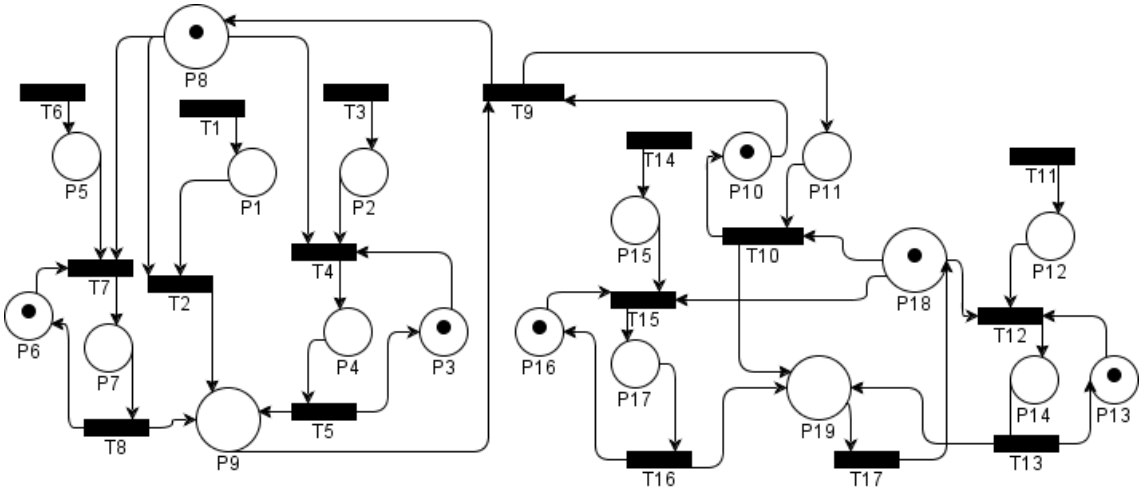

Ilustração 5.6.3: Rede de Petri do cenário virtual

Tabela 5.6.1: Legenda da rede de Petri

P1	Número de caixas no tapete 1	T1	Entrada duma caixa no tapete 1
P2	Número de caixas no tapete 2	T2	Transferência duma caixa do tapete 1 para o separador 1
P3	Tapete angular 2 livre	T3	Entrada duma caixa no tapete 2
P4	Tapete angular 2 ocupado	T4	Transferência duma caixa do tapete 2 para o tapete angular 2
P5	Número de caixas no tapete 3	T5	Transferência duma caixa do tapete angular 2 para o separador 1
P6	Tapete angular 3 livre	T6	Entrada duma caixa no tapete 3
P7	Tapete angular 3 ocupado	T7	Transferência duma caixa do tapete 3 para o tapete angular 3
P8	Separador 1 livre	T8	Transferência duma caixa do tapete angular 3 para o separador 1
P9	Presença duma caixa sobre o separador 1	T9	Transferência duma caixa do separador 1 para o tapete 7
P10	Tapete 7 livre	T10	Transferência duma caixa do tapete 7 para o separador 2
P11	Tapete 7 ocupado	T11	Entrada duma caixa no tapete 4
P12	Número de caixas no tapete 4	T12	Transferência duma caixa do tapete 4 para o tapete angular 4
P13	Tapete angular 4 livre	T13	Transferência duma caixa do tapete angular 4 para o separador 2
P14	Tapete angular 4 ocupado	T14	Entrada duma caixa no tapete 5
P15	Número de caixas no tapete 5	T15	Transferência duma caixa do tapete 5 para o tapete angular 5
P16	Tapete angular 5 livre	T16	Transferência duma caixa do tapete angular 5 para o separador 2
P17	Tapete angular 5 ocupado	T17	Transferência duma caixa para o tapete de saída, após a qual considera-se que foi eliminada
P18	Separador 1 livre		
P19	Presença duma caixa sobre o separador 2		

A rede é reversível, e todas as suas transições são vivas. Com a exceção dos lugares correspondentes aos tapetes alimentadores, nenhum lugar da rede pode apresentar mais do que uma marca. Isto confirma a conservabilidade da rede.

5.6.3. Distribuição do Controlo

O controlo do cenário virtual foi repartido entre dois controladores: o *Omron* CP1L e o *SoftPLC Codesys Controlo Win* V3. O CP1L ficou encarregue do controlo dos separadores, do tapete intermédio 7 e do tapete de saída. Este controlador também foi programado para gerir a lógica de encaminhamento com base nas filas de espera e de interagir com o utilizador através da consola HMI, assim como ler as variáveis do estado do cenário.

O *SoftPLC Codesys* foi feito responsável pelo controlo dos tapetes alimentadores e dos respetivos tapetes angulares. O controlo de cada tapete foi implementado numa tarefa de programa independente, permitindo agir como se cada tapete alimentador fosse controlado por um PLC diferente. As variáveis envolvidas foram listadas na Tabela 5.6.2.

Tabela 5.6.2: Variáveis trocadas entre o cenário e os controladores

Variáveis de entrada:	Tipo de variável:	Controlador:
Factory.Run	Binária	*Omron* CP1L
Factory.Pause		
Factory.Reset		
Sensores difusivos 6, 7 e 8	Três variáveis binárias	
Sensores difusivos 1, 2, 3, 4 e 5	Cinco variáveis binárias	*Codesys Controlo Win* V3
Variáveis de saída:	**Tipo de variável:**	**Controlador:**
Tapetes 6 e 7	Duas variáveis binárias	*Omron* CP1L
Separadores 1 e 2 – Mover em frente		
Separadores 1 e 2 – Reorientar para a esquerda ou para a direita	Quatro variáveis binárias	
Tapetes 1, 2, 3, 4 e 5	Cinco variáveis binárias	*Codesys Controlo Win* V3
Tapetes laterais 2, 3, 4 e 5	Quatro variáveis binárias	
Emissores 1, 2, 3, 4 e 5	Cinco variáveis binárias	

5.6.4. Inclusão duma Consola HMI

Para este cenário virtual, procedeu-se também à criação duma HMI com a consola programável *Omron* NB-5Q (introduzida na secção 3.2.7). A HMI (Ilustração 5.6.4) foi concebida para permitir monitorizar o ambiente virtual, mostrando quais os atuadores da instalação (tapetes ou separadores) que estão ocupados por caixas a cada momento, assim como revelar o registo das filas de espera em cada separador. A interface também possui botões de comando para substituir aqueles que seriam normalmente utilizados num painel dentro do ambiente virtual.

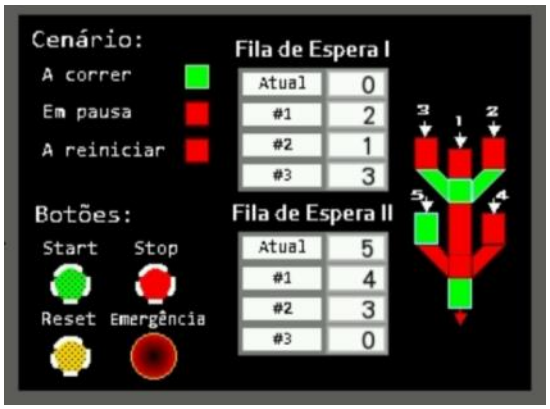

Ilustração 5.6.4: Ecrã da consola HMI

5.6.5. Realização das Comunicações

As comunicações entre os controladores foram efetuadas por via duma ligação *Ethernet/IP*, da qual o *SoftPLC Codesys* e o *Omron* CP1L foram designados, respetivamente, como mestre e como escravo. O CP1L foi ligado à consola NB-5Q por *Modbus* TCP, em que o controlador agiu como servidor e a HMI como cliente. A interação entre o cenário virtual e os controladores foi efetuado por servidor OPC *Codesys*, com o *SoftPLC*, e por *Modbus* TCP, com o *Omron* CP1L.

O diagrama de sequência da Ilustração 5.6.5 mostra como decorreram as trocas de dados entre os controladores e a HMI. O diagrama de componentes da Ilustração 5.6.6 revela a implementação das comunicações em termos de suportes físicos e protocolos.

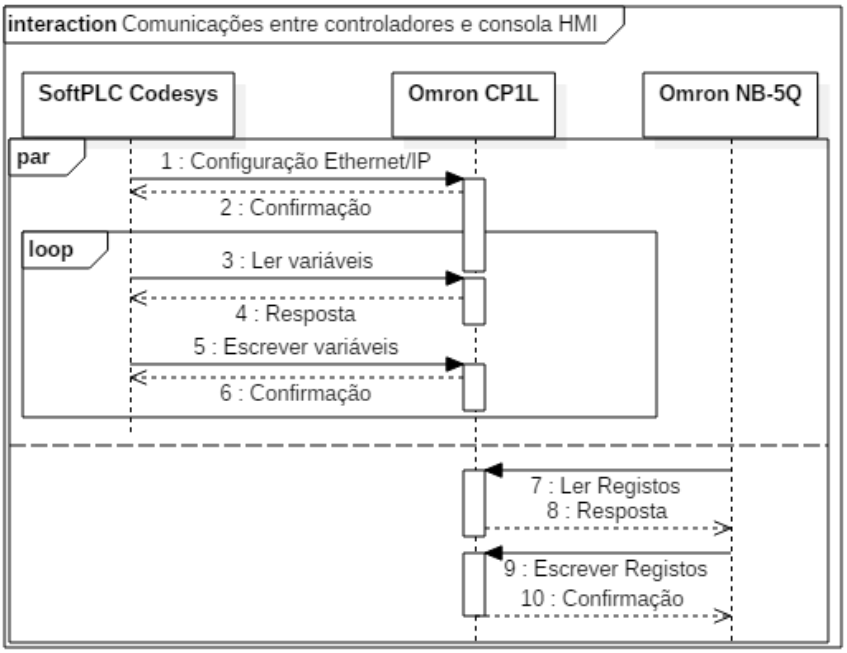

Ilustração 5.6.5: Diagrama de sequência da comunicação entre controladores e a consola

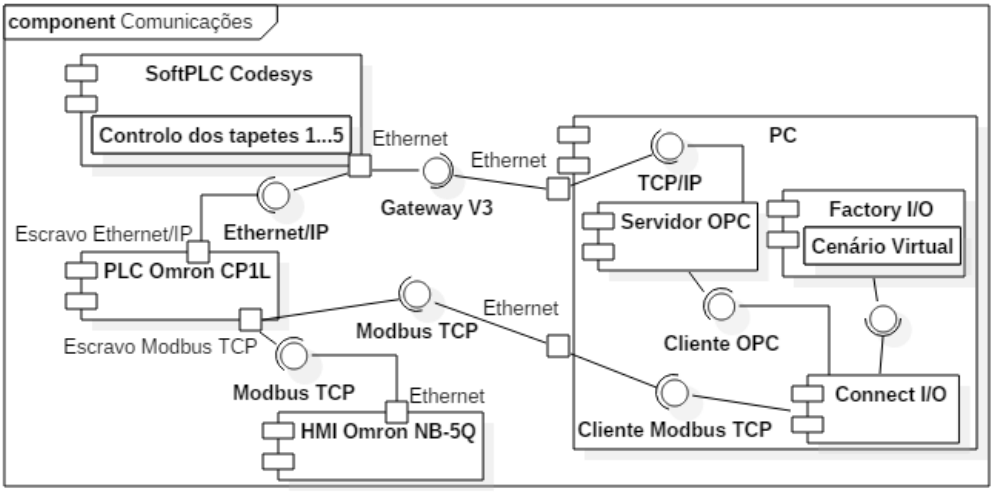

Ilustração 5.6.6: Diagrama de componentes da implementação das comunicações

5.6.6. Coordenação do Controlo

A Tabela 5.6.3 lista as variáveis trocadas entre os controladores lógicos. As variáveis correspondentes ao controlo dos tapetes alimentadores são necessárias para a evolução da atividade da transferência de caixas desde os tapetes até ao separador, representada no diagrama de sequência da Ilustração 5.6.7. O comportamento do tapete alimentador 1 e do tapete intermédio 7 são similares, mas mais simplificados devido ao facto de não existirem tapetes angulares entre eles e os separadores.

A tarefa de controlo correspondente a cada tapete transmite a sua própria variável *ResetConfirm*.

Tabela 5.6.3: Variáveis trocadas entre controladores

Variável:	Tipo de variável:	PLC que a determina:	Significado:
SinalStart	Binária	Omron CP1L	Início de funcionamento
SinalStop			Paragem de funcionamento
SinalReset			Reiniciação de variáveis
Tapete1…5_SendGo			Autorização para transferir caixa
Tapete1…5_SendDone			Fim da transferência
Tapete1…5_SendReq		Codesys Controlo Win V3: Tarefas de controlo dos tapetes alimentadores 1 a 5	Pedido de transferência de caixa
Tapete1…5_SendOn			Transferência de caixa decorrente
Tapete1…5_ResetConfirm			Confirmação da reiniciação de variáveis

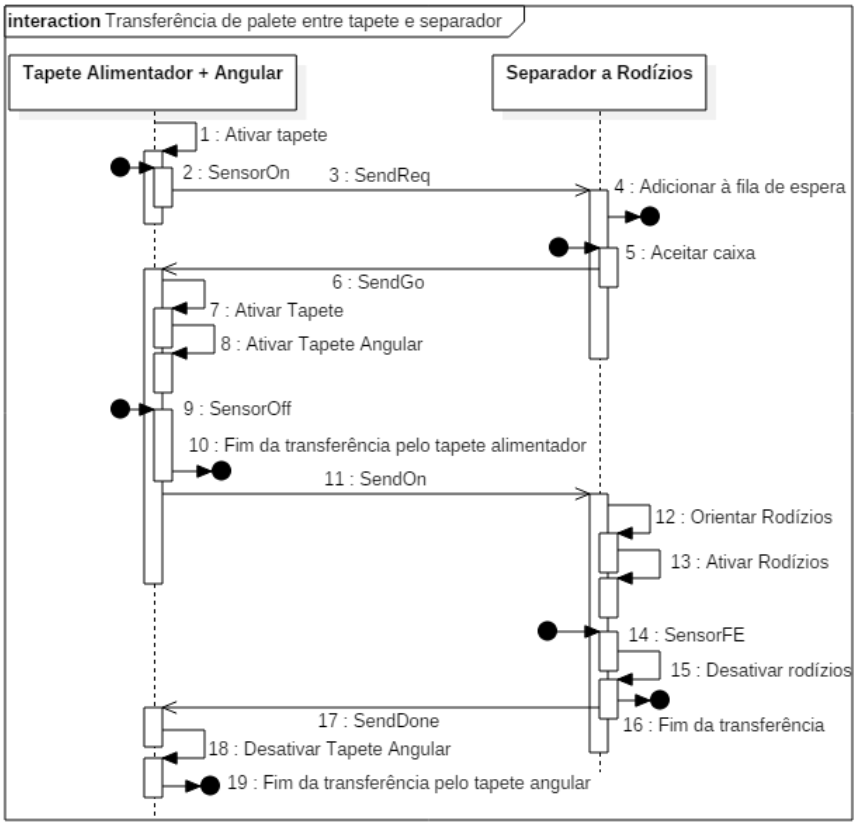

Ilustração 5.6.7: Diagrama de sequência da atividade de transferência duma caixa

O diagrama de atividade da Ilustração 5.6.8 apresenta o funcionamento duma das listas de espera. Os pedidos da parte dos tapetes são adicionados à lista um a um, sendo as caixas aceites na ordem com que os pedidos foram recebidos.

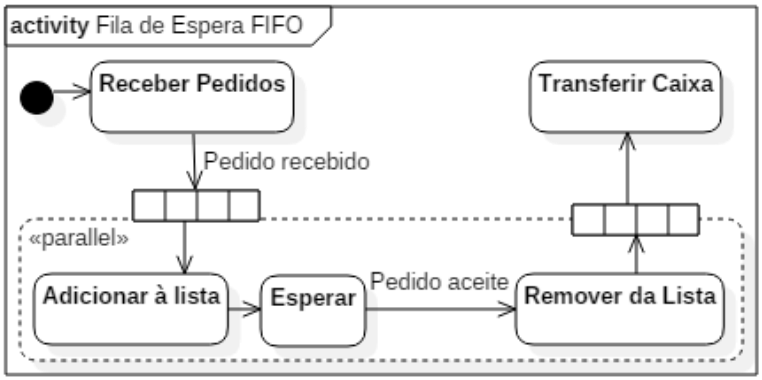

Ilustração 5.6.8: Diagrama de atividade da fila de espera FIFO

5.6.7. Resultados e Observações

Obteve-se o comportamento pretendido para este caso de estudo. A divisão do programa do *SoftPLC Codesys* em tarefas independentes permitiu simular a utilização de um número maior de controladores.

Apesar de não ser particularmente atual, o *Omron* CP1L pôde utilizar protocolos de comunicação mais modernos, estes sendo *Modbus* TCP e *Ethernet/IP*, graças à utilização das *option boards Ethernet*. Os detalhes da comunicação Ethernet/IP foram definidos através do *Codesys Control Win* V3, mestre da ligação.

A inclusão da consola *Omron* NB-5Q concedeu uma interface com o utilizador mais sofisticada, em relação aos casos de estudo anteriores em que se utilizou um painel de operador incluído no cenário.

Este caso de estudo pode ser desenvolvido para adotar diferentes padrões de encaminhamento das caixas, tal como foi referido para os cenários anteriores.

5.7. Encaminhador com Quatro Mesas Rotativas, Quatro Entradas e Quatro Saídas

Este caso de estudo (Ilustração 5.7.1) é o último dos cenários desenvolvidos nesta dissertação. É composto por um conjunto de quatro mesas rotativas adjacentes, ladeado por um tapete alimentador e um de tapete de saída em cada lado. Foi colocado um sensor difusivo na fronteira entre cada par de atuadores adjacentes. Sobre cada mesa rotativa situa-se um conjunto de sinalizadores do tipo *stack-light* (Ilustração 5.7.2).

Ilustração 5.7.1: Encaminhador com quatro mesas rotativas, quatro entradas e quatro saídas

Sobre esta instalação circularão paletes sobre as quais assentarão itens diversos.

Ilustração 5.7.2: Mesa rotativa unida a dois tapetes e outras duas mesas

5.7.1. Comportamento

Pretende-se que seja possível que uma palete originada de qualquer tapete alimentador possa ser reencaminhada para qualquer tapete de saída, por via da circulação sobre as mesas rotativas. As mesas rotativas apenas podem deslocar paletes entre si no sentido horário, visto de cima. A Ilustração 5.7.3 apresenta os possíveis trajetos a descrever por uma palete originada de um dado tapete.

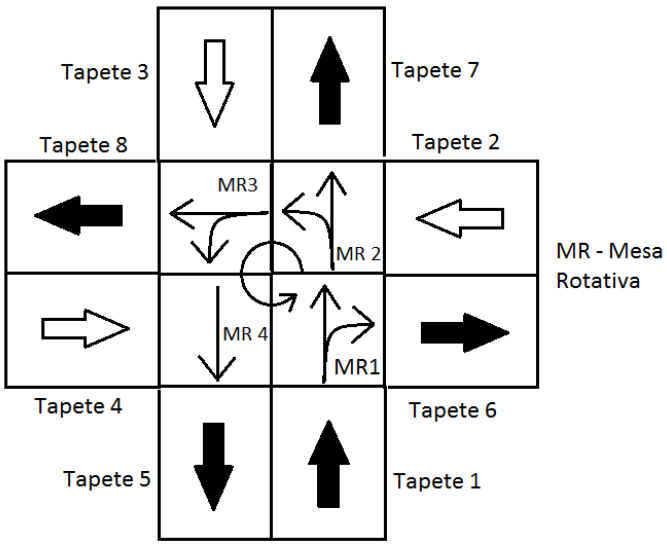

Ilustração 5.7.3: Trajetos possíveis duma palete com origem no tapete 1

Os sinalizadores *stack-light* colocados sobre cada mesa emitirão uma luz diferente, consoante o estado da transferência duma palete por parte do tapete de entrada (Tabela 5.7.1).

Tabela 5.7.1: Sinais do sinalizador *stack-light*

Cor do sinalizador	Situação
Vermelho	A mesa não aceita nenhuma palete do tapete alimentador
Amarelo	A próxima palete a receber pela mesa vem do tapete alimentador
Verde	A mesa aceita a palete do tapete alimentador

5.7.2. Modelação por Rede de Petri

Para a rede de Petri deste cenário, adotou-se uma representação simplificada, de modo a obter um número reduzido de lugares e transições.

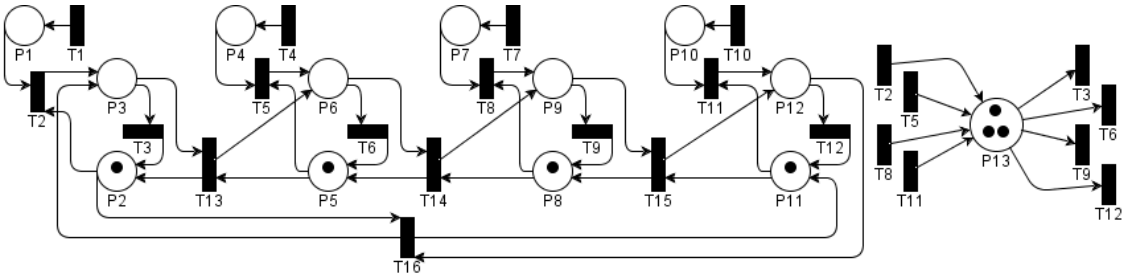

Ilustração 5.7.4 Rede de Petri do cenário virtual

Os lugares P1/4/7/10 correspondem ao número de paletes existentes nos tapetes alimentadores, não impondo um limite sobre esse valor. Os lugares P2/5/8/11 dizem respeito às mesas rotativas, quando estas se encontram livres para receber paletes. Os lugares P3/6/9/12 representam as mesas quando estas se encontram ocupadas.

As transições correspondem todas à transferência de paletes entre dois elementos transportadores. As transições T3/6/9/12 constituem a única representação dos tapetes de saída, pressupondo-se que estes garantidamente eliminam as paletes ao recebê-las. As

transições T13/14/15/16, em conjunto com os lugares que interligam, formam uma arquitetura em anel que ilustra a circulação de paletes sobre as mesas rotativas.

O lugar P13 protege a rede de possíveis situações de impasse, nomeadamente a situação em que todas as quatro mesas se encontram ocupadas, impedindo a transferência de paletes entre si. Este lugar restringe o cenário a um máximo de 3 paletes a circular sobre as mesas. Os restantes detalhes da rede encontram-se na legenda da Tabela 5.7.2.

Tabela 5.7.2: Legenda da rede de Petri

Lugar	Significado	Transição	Significado
P1/P4/P7/P10	Número de paletes no tapete alimentador 1/2/3/4	T1/T4/T7/T10	Tapete alimentador 1/2/3/4 recebe uma palete
P2/P5/P8/P11	Mesa rotativa 1/2/3/4 livre	T2/T5/T8/T11	Mesa rotativa 1/2/3/4 aceita palete do tapete alimentador adjacente
P3/P6/P9/P12	Mesa rotativa 1/2/3/4 ocupada	T3/T6/T9/T12	Mesa rotativa 1/2/3/4 aceita palete da mesa a montante
P13	Número de paletes admissíveis	T13/T14/T15/T16	Mesa rotativa 1/2/3/4 envia palete para o tapete de saída adjacente

Nota-se que o estado de transferência duma palete não foi representado na rede, pelo que, no instante inicial duma transferência, o elemento transportador a montante é considerado livre e o elemento a jusante é considerado ocupado.

5.7.3. Distribuição do controlo

O controlo foi distribuído entre um *SoftPLCs Codesys Control Win* V3, o *Moeller* EC4-200, o Siemens S7-1200 e o *Omron* CP1L.

O *Codesys Control Win* V3 ficou encarregue de controlar as mesas rotativas, um tapete alimentador e um tapete de saída, assim como de gerir as ordens de transferência de paletes entre elementos transportadores. Os restantes controladores ficaram responsáveis pelo controlo de cada um dos restantes pares de tapete alimentador e tapete de saída. As variáveis em causa estão listadas na Tabela 5.7.3.

5.7.4. Implementação das Comunicações

Os controladores comunicaram entre si por uma ligação *Modbus* TCP, da qual o *Codesys Control Win* V3 foi designado cliente e cada um dos restantes controladores participou como servidor. A ligação com o *Connect I/O* foi efetuado por meios já utilizados em casos anteriores: servidor OPC *Codesys* para o *SoftPLC Codesys* e para o PLC *Moeller*, *Modbus* TCP para o CP1L e por via do driver *Siemens* para o S7-1200.

Dado que todos estes tipos de ligação já foram implementados em casos de estudo anteriores, não se concebeu um diagrama de sequência representativo, apresentando-se somente o diagrama de componentes da Ilustração 5.7.5.

Tabela 5.7.3: Variáveis trocadas entre o cenário virtual e os controladores

Variáveis de entrada:	Tipo de variável:	Controlador:
Factory.Run	Binária	*Codesys Control Win* V3
Factory.Pause		
Factory.Reset		
Botão *Start*		
Botão *Stop*		
Sensores de trás das mesas rotativas		
Sensores frontais das mesas rotativas		
Sensores de posição de 0° das mesas rotativas		
Sensores de posição de 90° das mesas rotativas		
Sensores difusivos dos tapetes 4 e 8	Duas variáveis binárias	
Sensores difusivos dos tapetes 1 e 5		*Siemens* S7-1200
Sensores difusivos dos tapetes 2 e 6		*Moeller* EC4-200
Sensores difusivos dos tapetes 3 e 7		*Omron* CP1L
Variáveis de saída:	**Tipo de variável:**	**Controlador:**
Luz do botão *Start*	Binária	*Codesys Control Win* V3
Luz do botão *Stop*		
Sinalizadores das mesas	Doze variáveis binárias	
Tapetes das mesas – sentido positivo	Quatro variáveis binárias	
Tapetes das mesas – sentido negativo		
Rotação das mesas – sentido positivo		
Rotação das mesas – sentido negativo		
Tapetes de rolos 4 e 8	Duas variáveis binárias	
Emissor 4	Binária	
Tapetes de rolos 1 e 5	Duas variáveis binárias	*Siemens* S7-1200
Emissor 1	Binária	
Tapetes de rolos 2 e 6	Duas variáveis binárias	*Moeller* EC4-200
Emissor 2	Binária	
Tapetes de rolos 3 e 7	Duas variáveis binárias	*Omron* CP1L
Emissor 3	Binária	

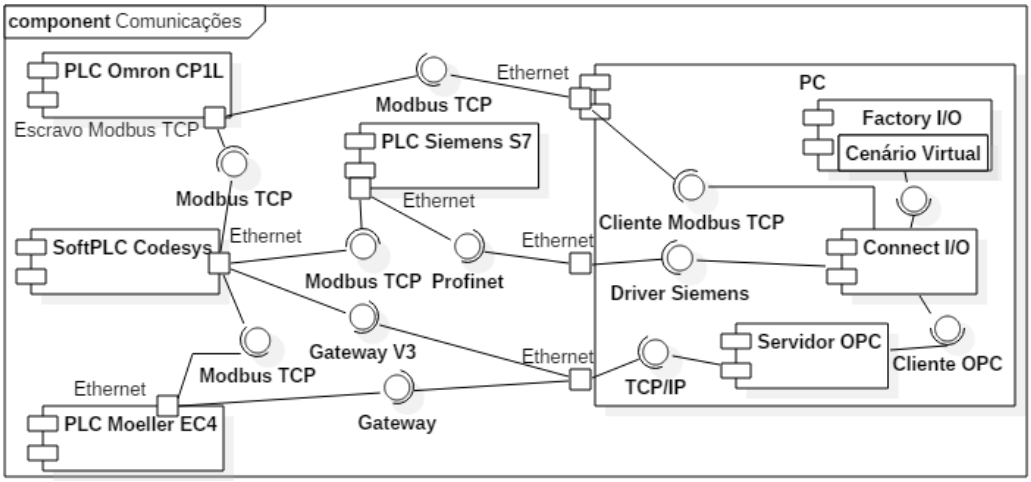

Ilustração 5.7.5: Diagrama de componentes da implementação das comunicações

5.7.5. Coordenação do Controlo

O *Codesys Control Win* foi designado o coordenador do controlo. Foram implementadas as funções de Start e Stop definidas para os restantes cenários.

Foi novamente implementado um algoritmo de fila de espera. Desta vez, a fila de espera regista a origem de cada palete e o trajeto a descrever. O destino de cada palete é decidido pela tarefa de controlo do respetivo tapete alimentador e é comunicado (sobre uma variável do tipo inteiro) à tarefa responsável por gerir a fila de espera. Quando a mesa recebe a palete, o seu destino é traduzido numa *string* que contém a sequência de mesas rotativas que a palete deve seguir para atingir o seu destino. Quando uma palete é transferida entre mesas, esse itinerário é também transmitido.

Dado que os algoritmos de controlo direto são similares em todos os PLCs, as variáveis comunicadas serão descritas a nível das interações entre tarefas, e não entre controladores. As tarefas de controlo dos diferentes elementos transportadores trocam variáveis entre si para executar corretamente a transferência de paletes. Algumas destas variáveis são também transmitidas à tarefa da fila de espera, pois esta é também responsável por monitorizar o estado de ocupação de cada mesa rotativa. As variáveis descritas estão listadas na Tabela 5.7.4, em que o símbolo # representa um qualquer dos atuadores de cada tipo.

Tabela 5.7.4: Variáveis transmitidas entre tarefas de controlo

Variável:	Tipo de variável:	Tarefa que a determina:	Tarefa que a recebe:	Significado:
SinalStart	Binária	Tarefa de Controlo	Restantes tarefas	Início do funcionamento
SinalStop				Paragem do funcionamento
SinalReset				Reiniciação das variáveis
SinalResetConfirm		Restantes tarefas	Tarefa de Controlo	Reiniciação concluída
Tapete#Req		Tapete alimentador	Fila de espera	O tapete espera para enviar uma palete
Tapete#Destino	Inteira		Fila de espera	Destino da palete a enviar
ReceberTapete#	Binária	Fila de espera	Mesa Rotativa	Ordem de receção de palete do tapete
TrajetoTapete#	*String*			Itinerário da palete
MoverTapete#	Binária	Mesa Rotativa	Tapete alimentador	Ordem para o tapete enviar a palete para a mesa
ReceberTapete#Fim				Fim da receção da palete
EnviarTapeteSaída#			Tapete de saída	Pedido para enviar palete para o tapete de saída
EnviarMesa#			Mesa a jusante	Pedido para enviar palete para a mesa seguinte
TrajetoMesa#	*String*		Mesa a jusante	Itinerário da palete a enviar para a mesa seguinte
ReceberMesa#	Binária		Mesa a montante, Fila de espera	Ordem para a mesa a montante enviar a palete
TapeteSaída#Pronto		Tapete de saída	Mesa, Fila de espera	Ordem para a mesa enviar a palete para o tapete
EnviarTapete#Fim			Mesa	Fim da receção da palete pelo tapete de saída

Elaborou-se um diagrama de sequência para ilustrar cada uma das interações de transferência de paletes: entre os tapetes alimentadores e as mesas rotativas (Ilustração 5.7.6), entre mesas rotativas adjacentes (Ilustração 5.7.7) e entre as mesas rotativas e os tapetes de saída (Ilustração 5.7.8). As mensagens perdidas "Mesa Livre" e "Mesa Ocupada" representam a mudança do estado de ocupação das mesas rotativas, visto pela tarefa da fila de espera.

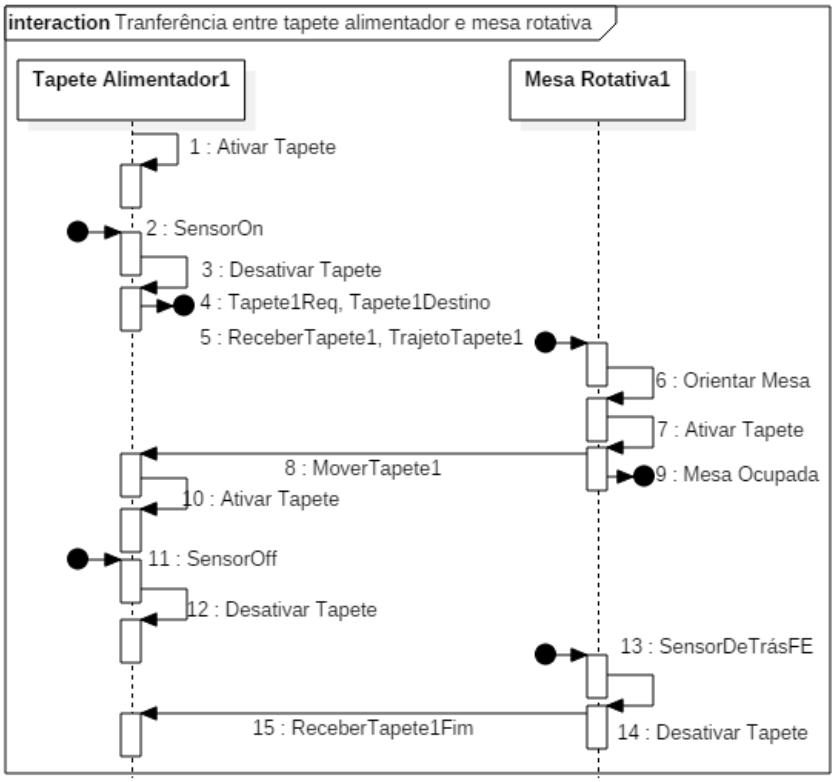

Ilustração 5.7.6: Diagrama de sequência da transferência duma palete entre um tapete alimentador e uma mesa rotativa

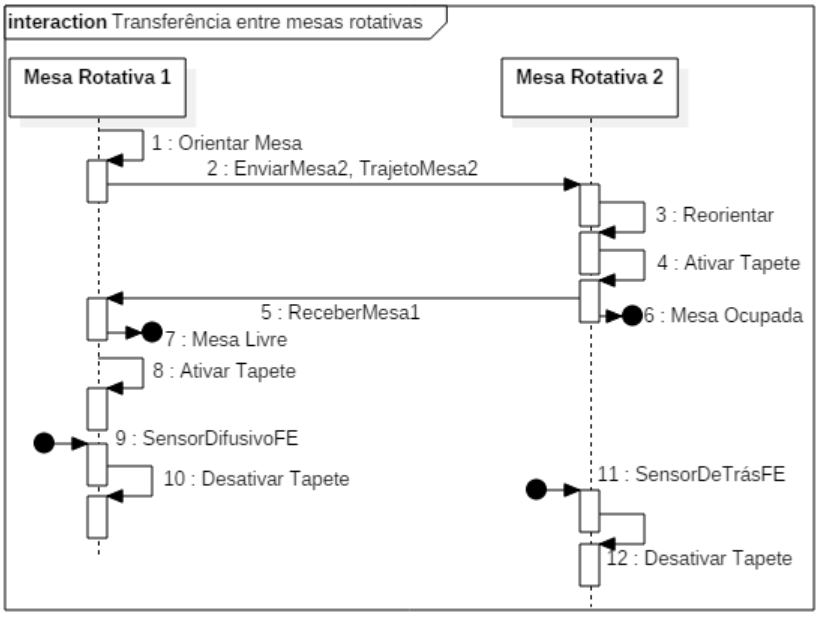

Ilustração 5.7.7: Diagrama de sequência da transferência duma palete entre mesas rotativas

A fila de espera programa é representada pelo diagrama de atividade (Ilustração 5.7.9). Os pedidos são verificados por ordem de envio. Se as primeiras duas mesas do itinerário da palete estiverem livres, o pedido é aceite e removido da lista; senão, é ignorado e é lido o pedido seguinte.

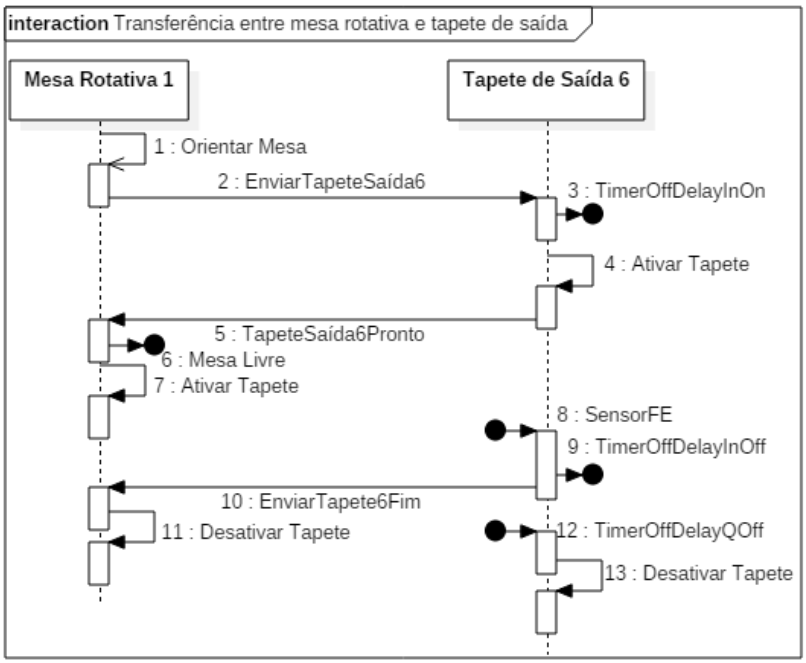

Ilustração 5.7.8: Diagrama de sequência da transferência duma palete entre uma mesa e um tapete de saída

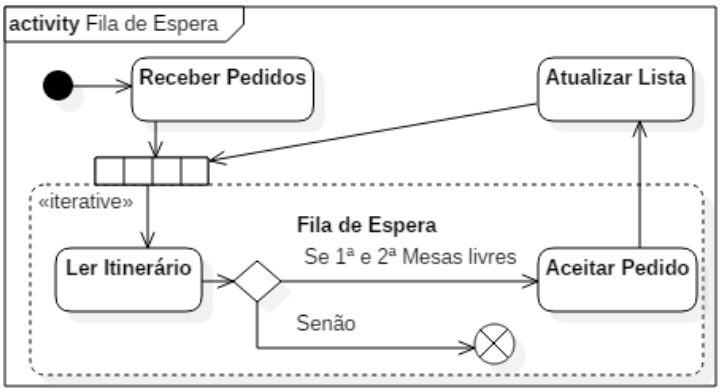

Ilustração 5.7.9: Diagrama de atividade da fila de espera

5.7.6. Resultados e Observações

Devido ao elevado número de controladores usados, a troca de dados entre estes e o cenário virtual foi percetivelmente mais demorada, notando-se um atraso dos atuadores a certos eventos. No entanto, foi possível obter o comportamento desejado para este caso de estudo.

Tanto a fila de espera programada como o algoritmo de encaminhamento foram bem executados, de modo que nunca se observou um impasse do sistema, nem um incorreto encaminhamento das paletes.

Todos os modos de comunicação utilizados tinham sido implementados em casos de estudo anteriores com a exceção da comunicação *Modbus* TCP através do *Moeller* EC4P-200, cuja execução requereu a utilização de bibliotecas de programa especiais. Inicialmente, a comunicação por parte do EC4P-200 não funcionava devidamente, tendo sido necessária a atualização do seu sistema operativo para resolver essa situação.

Dada a complexidade da atividade executada, este caso de estudo pode servir de base para o estudo de otimização doa algoritmos de encaminhamento, no contexto de requisitos temporais e da automação responsiva.

5.8. Conclusões do Capítulo

Os casos de estudo desenvolvidos, ao longo deste capítulo, foram apresentando maior tamanho e complexidade, apesar de todos terem sido baseados em torno de atividades de movimentação de objetos. Isto é observável em vários aspetos, tais como a quantidade de dados trocados entre os controladores e entre estes e o cenário, ou a complexidade dos diagramas de sequência representativos dos vários processos e atividades.

Em consequência disto, a abordagem tomada na representação dos casos de estudo por rede de Petri também sofreu uma evolução: inicialmente, todos os estados do sistema eram representados com lugares individuais, incluindo os estados de transferência de itens entre atuadores. Com a necessidade de reduzir a complexidade destes modelos, as operações de transferência passaram a ser representadas como transições de estado.

Nos cenários tratados inicialmente, a sua implementação foi facilitada pela sofisticação do *software* dos controladores utilizados. À medida que foram sendo utilizados PLCs menos atuais, foi surgindo uma necessidade crescente de recorrer a vários dispositivos de *hardware* adicionais, tais como placas de aquisição, conversores de suportes físicos e módulos de comunicação embebidos ou exteriores aos controladores. Adicionalmente, nalguns casos foi preciso recorrer a atualizações de *software* ou, até mesmo, à sua reversão para uma versão mais antiga.

No entanto, foi possível obter uma coordenação satisfatória dos dispositivos e dos protocolos envolvidos, de modo que as comunicações entre o cenário e entre os controladores foram todas executadas com sucesso.

Capítulo 6 - Conclusões e Trabalhos Futuros

Esta dissertação girou em torno do problema da coordenação de controladores e protocolos de comunicação no âmbito da automação industrial. Numa abordagem a este problema utilizou-se ambientes industriais virtuais gerados através do *Factory I/O* como alvo do controlo distribuído por PLCs.

Esse controlo distribuído foi feito por controladores lógicos de diversas gerações e fabricantes e foi complementado pela implementação de controlo cooperativo entre esses mesmos controladores, através da elaboração de redes de comunicação industrial. Essas redes foram resultado da combinação de diferentes protocolos industriais e suportes físicos.

Considera-se que este trabalho de dissertação constituiu um bom exercício de aprendizagem para o seu autor, permitindo uma compreensão razoável das questões, princípios e conceitos relativos áreas da automação fabril, do controlo distribuído e da comunicação industrial. Para além disso, gerou motivação para o estudo de técnicas e normas de modelação de sistema, em particular as normas UML e *SysML* e as redes de Petri.

Observou-se que cada *software* de programação apresentava as suas particularidades, a nível de interface e a nível de linguagens disponíveis. Também se notou que, com a evolução destas tecnologias, estes *software* começaram a apresentar funcionalidades cada vez mais complexas e em maior número, tornando a área da programação industrial cada vez mais especializada. Por outro lado, os programas mais antigos mostravam-se mais simples e mais fáceis de utilizar, após o utilizador se acostumar a eles.

Considera-se que as maiores dificuldades na realização deste trabalho residiram na tarefa da compatibilização das comunicações entre controladores lógicos, na correta instalação dos elementos de *software* e *hardware* adicionais e na adaptação do autor às especificidades de cada programa e de cada controlador.

Confirma-se que os cenários virtuais *Factory I/O* constituíram uma poderosa ferramenta de treino, não só para a programação de PLCs, como para a prática do controlo distribuído e para o estudo de algoritmos de controlo e técnicas de implementação de modelos.

As normas de modelação utilizadas revelaram-se bastante versáteis, permitindo representar diversos aspetos dos casos de estudo desenvolvidos, apesar de apenas terem sido utilizados quatro tipos de diagramas UML e *SysML*, parte pequena das capacidades destas normas. Em conjunto com essas especificações, utilizou-se as redes de Petri, que mostraram-se úteis para representar o comportamento dinâmico desejado para os cenários desenvolvidos, de forma simples e compacta.

A distribuição e coordenação do controlo dos cenários, exercício principal desta dissertação, foram executadas com sucesso. À medida que foram sendo aplicados controladores sucessivamente mais antiquados, a implementação das comunicações passou a depender cada vez menos das facilidades existentes nos *software* de programação, recaindo na utilização de *hardware* adicional, destinado a melhorar a versatilidade dos PLCs.

Aponta-se que o trabalho desenvolvido nesta dissertação demonstra um vasto potencial de desenvolvimento, quer dentro, quer para além do âmbito do controlo distribuído. Este desenvolvimento pode incluir exercícios tais como:

- Utilização das capacidades dos cenários virtuais *Factory I/O* para simular outros tipos de sistemas controláveis, como por exemplo o enchimento dum tanque de água, de modo a estudar princípios e algoritmos de controlo de natureza diferente aos tratados neste trabalho;
- Execução simultânea de vários ambientes virtuais, com o *Factory I/O*, ou com outros programas de simulação, e realização de coordenação entre eles;
- Coordenação entre um cenário virtual e algum sistema físico real;
- Otimização e sofisticação dos algoritmos desenvolvidos nesta dissertação, de modo a serem mais adequados ao contexto da automação ágil ou responsiva;
- Implementação do controlo de supervisão utilizando outros protocolos e serviços não tratados nesta dissertação, tais como páginas *Web* e envio de mensagens SMS.

Estas sugestões, entre outras possibilidades, poderão constituir trabalhos de treino ou de investigação para alunos do Ramo de Automação de anos futuros.

Bibliografia

1. **Real Games.** *Next-Gen PLC Training Software with FACTORY I/O.* [Online] Real Games, 2017. https://factoryio.com/.

2. —. *Real Games | Educational 3D simulation software.* [Online] Real Games, 2017. https://realgames.co/.

3. **Carvalho, Adriano Moreira da Silva de.** *Integração de Redes de Telemetria em Ambientes Industriais.* 2010. Master Thesis.

4. **Silva, Carlos Miguel Soares da.** *Desenvolvimento de Sistema Distribuído para Controlo e Monitorização de Ensaios Laboratoriais.* 2007. Master Thesis.

5. **Magalhães, António Pessoa de.** *Computação Industrial - Apresentação da disciplina no contexto da Automação Industrial.* 2016.

6. **Derby, Stephen J.** *Design of Automatic Machinery.* s.l. : Marcel Dekker, 2005.

7. **Magalhães, António Pessoa de.** *Automação e Controlo Industrial - Conceitos Básicos e Arquiteturas.* 2016.

8. **Boyer, Stuart A.** *SCADA - Supervisory Control and Data Acquisition.* 3ª. s.l. : ISA - The Instruments, Systems, and Automation Society, 2004.

9. **Lydon, Bill.** Cover Story: Industry 4.0: Intelligent and Flexible Production - ISA. *ISA.* [Online] 2016. https://www.isa.org/intech/20160601/.

10. **Bolton, W.** *Programmable Logic Controllers.* s.l. : Elsevier Newnes, 1996.

11. **Mehta, B. R. and Reddy, Y. J.** *Industrial Process Automation Systems.* s.l. : Elsevier Inc., 2015.

12. International Standard IEC 61131-3. 2003-01. s.l. : International Electrotechnical Commission, 2003.

13. **Anaheim Automation.** HMI | Human Machine Interfaces. *Anaheim Automation.* [Online] 2017. [Cited: Maio 8, 2017.] http://www.anaheimautomation.com/manuals/forms/hmi-guide.php#sthash.2McqS5xo.p9GninQ0.dpbs.

14. **Galloway, Brendan and Hancke, Gerhard P.** Introduction to Industrial Control Networks. *IEEE COMMUNICATIONS SURVEYS & TUTORIALS.* s.l. : IEEE, 2013, Vol. 15.

15. **Mahalik, N. P.** *Fielbus Technology - Industrial Network Standards for Real-Time Distributed Control.* s.l. : Springer, 2003.

16. **Djiev, S.** Industrial Networks for Communication and Control. Reading for "Elements of Industrial Automation" at ELDE.

17. **Can in Automation.** History of CAN technology. *Can in Automation (CiA).* [Online] 2017. https://www.can-cia.org/can-knowledge/can/can-history/.

18. **Corrigan, Steve.** *Introduction to the Controller Area Network (CAN).* Texas Instruments. 2008. Relatório de Aplicação. Publicado em Agosto de 2002, revisto em Julho de 2008.

19. **Real Games.** CONNECT I/O. *Real Games | Educational 3D simulation software.* [Online] 2017.

20. —. Manual. *Next-Gen PLC Training Software with FACTORY I/O.* [Online] 2017. https://factoryio.com/docs/manual/.

21. —. Real Games - Youtube. *Youtube.* [Online] 2017. https://www.youtube.com/user/RealGamesLDA/feed.

22. **3S-Smart Software Solutions GmnH.** *CODESYS - industrial IEC 61131-3 PLC programming.* [Online] 2017. https://www.codesys.com/.

23. **3S-Smart Software Solutions.** CODESYS OPC Server V3 - Installation and Usage.

24. **Siemens.** *S7-1200 Programmable controller System Manual.* 2015.

25. —. *S7-200 Programmable Controller System Manual.* 2008.

26. —. *HMI devices Basic Panels - Operating instructions.* 2012.

27. **Roersch, Peter.** *Programmable Logic Controller easyControl EC4-200.* s.l. : Eaton Industries GmbH, 2010.

28. **Omron.** *CP1L CPU Unit - Operation Manual.* 2010.

29. **ODVA.** *Ethernet/IP ODVA Technology Overview Series.* 2016.

30. **Schneider Electric.** *Global Specialist in Energy Management and Automation | Schneider Electric.* [Online] 2016. http://www.schneider-electric.co.uk/en/.

31. **Westermo.** *Converter RS-232 - RS-422/485 - Installation Manual.* 2003.

32. **Telemecaníque.** *Modicon TSX Micro and PL7 Software.*

33. **Omron.** *NB-series - NB-Desginer Operation Manual.* 2016.

34. **Schuppen, Jan H. van, et al., et al.** Control of Distrbuted Systems - Tutorial and Overview. *European Journal of Control.* Setembro 2011.

35. **Cassandras, Christos G. and Lafortune, Stéphane.** *Introduction to Discrete Event Systems.* s.l. : Kluwer Academic Publishers, 1999.

36. **Zhou, MengChu and DiCesare, Frank.** *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems.* s.l. : Kluwer Academic Publishers, 1993.

37. **Ripps, David L.** *An Implementation Guide to Real-Time Programming.* s.l. : Prentice-Hall, 1990.

38. **Fong, Nga Hin Benjamin.** *Modeling, Analysis, and Design of Responsive Manufacturing Systems Using Classical Control Theory.* 2005.

39. *Dynamic modeling of surge effect and capacity limitation in supply chains.* **Helo, P. T.** 2000, International Journal of Production Research, Vol. 38.

40. **Rumbaugh, James, Jacobson, Ivar and Booch, Grady.** *The Unified Modeling Language Reference Manual.* s.l. : Addison-Wesley, 2005.

41. *OMG Unified Modeling Language ™ (OMG UML).* s.l. : Object Management Group, 2015.

42. Applications of UML - Wikipedia. *Wikipedia, the free encyclopedia.* [Online] Maio 2017. https://en.wikipedia.org/wiki/Applications_of_UML.

43. *OMG Systems Modeling Language (OMG SysML™).* s.l. : Object Management Group, 2015.

44. **Delligatti, Lenny.** *SysML Distilled - A Brief Guide to the Systems Modeling Language.* s.l. : Pearson Education, 2014.

45. **Zhou, MengChu and Zurawski, Richard.** Introduction to Petri Nets in Flexible and Agile Automation. [ed.] MengChu Zhou. *Petri Nets in Flexible and Agile Automation.* s.l. : Kluwer Academic Publishers, 1995, 1.

46. **Girault, Claude and Valk, Rüdiger.** *Petri Nets for Systems Engineering.* s.l. : Springer, 2003.

47. **Xing, Keyi, Hu, Baosheng and Chen, Haoxun.** Deadlock Avoidance Policy for Flexible Manufacturing Systems. *Petri Nets in Flexible and Agile Automation.* 1995, 9.

A. Anexo

Com este anexo pretende-se dar a conhecer em maior detalhe o processo de desenvolvimento seguido, usando um dos casos de estudo apresentados como exemplo ilustrativo.

A.1. Configuração das Comunicações dum Caso de Estudo

Nesta secção são mostradas imagens da configuração das comunicações dos controladores *Codesys Control Win* V3 e *Siemens* S7-1200, utilizados na caso de estudo "Encaminhador por Mesa Rotativa, com Duas Entradas e Duas Saídas Caso 1", documentado na secção 5.2.

A Ilustração A.1.1 apresenta a configuração das variáveis de comunicação com os atuadores e sensores no cenário no *SoftPLC Codesys*. Estas variáveis são transmitidas ao servidor OPC com o qual o *Connect I/O* estabelece comunicação.

GVL

```
1  {attribute 'qualified_only'}
2  VAR_GLOBAL
3      //Variáveis a serem trocadas com o Cenário Factory I/O via servidor OPC Codesys V3
4      In0, In1, In2, In3:BOOL:=0; //Sensores difusivos 1, 2, 3 e 4.
5      Out0,Out1,Out2,Out3:BOOL:=0; //Tapetes 1, 2, 3 e 4.
6  END_VAR
```

Ilustração A.1.1: Definição das variáveis a serem trocadas com o servidor OPC

A Ilustração A.1.2 mostra o mapeamento das variáveis a serem trocadas entre os controladores *Codesys* e *Siemens* quando estes comunicam por *Modbus* TCP enquanto servidor e cliente, respetivamente. Estas variáveis são mapeadas para os registos *Modbus* do *SoftPLC*.

Variable	Mapping	Channel	Address	Type	Unit	Description
		Inputs	%IW0	ARRAY [0..1] OF WORD		Modbus Holding Registers
		Inputs[0]	%IW0	WORD		Sinais vindos do S7-1200
Application.PLC...		Bit0	~~%IX0.0~~	BOOL		Pedido de envio de caixa do tapete 1 para a mesa rotativa
Application.PLC...		Bit1	~~%IX0.1~~	BOOL		Pedido de envio de caixa do tapete 2 para a mesa rotativa
Application.PLC...		Bit4	~~%IX0.4~~	BOOL		Sinal de paragem obrigatória
		Outputs	%QW0	ARRAY [0..1] OF WORD		Modbus Input Registers
		Outputs[0]	%QW0	WORD		Sinais dos sensores a serem reencaminhados para o S7-1200
Application.GVL...		Bit0	~~%QX0.0~~	BOOL		Sensor 1
Application.GVL...		Bit1	~~%QX0.1~~	BOOL		Sensor 2
Application.GVL...		Bit2	~~%QX0.2~~	BOOL		Sensor 3
Application.GVL...		Bit3	~~%QX0.3~~	BOOL		Sensor 4

Ilustração A.1.2: Mapeamento das variáveis no servidor *Modbus* TCP *Codesys*

Na Ilustração A.1.3 estão reveladas a configuração das funções de escrita e leitura de registos *Modbus* TCP, na situação em que o *Codesys* age como cliente.

Name	Access Type	Trigger	READ Offset	Length	Error Handling	WRITE Offset	Length
Channel 0	Read Holding Registers (Function Code 03)	Cyclic, t#100ms	16#0000	1	Keep last Value		
Channel 2	Write Multiple Registers (Function Code 16)	Cyclic, t#100ms				16#0000	2

Ilustração A.1.3: Funções *Modbus* TCP no *Codesys*

Na Ilustração A.1.4 aparece o bloco funcional relativo à operação de leitura de um registo *Modbus* TCP do *Codesys* enquanto escravo, executada pelo S7-1200. Os detalhes da ligação por TCP/IP são compilados na variável estruturada **Communications*.TCPactive_1*.

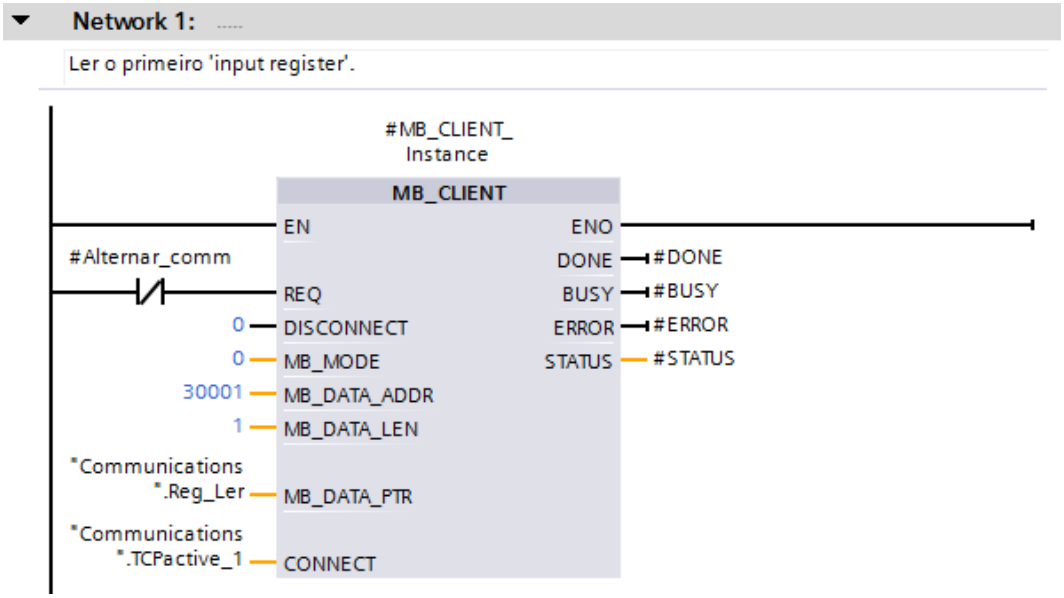

Ilustração A.1.4: Bloco de função *Modbus* TCP de leitura de *input registers* pelo S7-1200

A Ilustração A.1.5 mostra o bloco funcional de configuração do S7-1200 como um servidor *Modbus* TCP. Os detalhes da ligação por TCP/IP são compilados na variável estruturada **Communications*.TCPpassive_1*. O mapeamento das variáveis aos registos *Modbus* é definido pelo endereço "P#M50.0 WORD 2", que corresponde a duas *words* de memória.

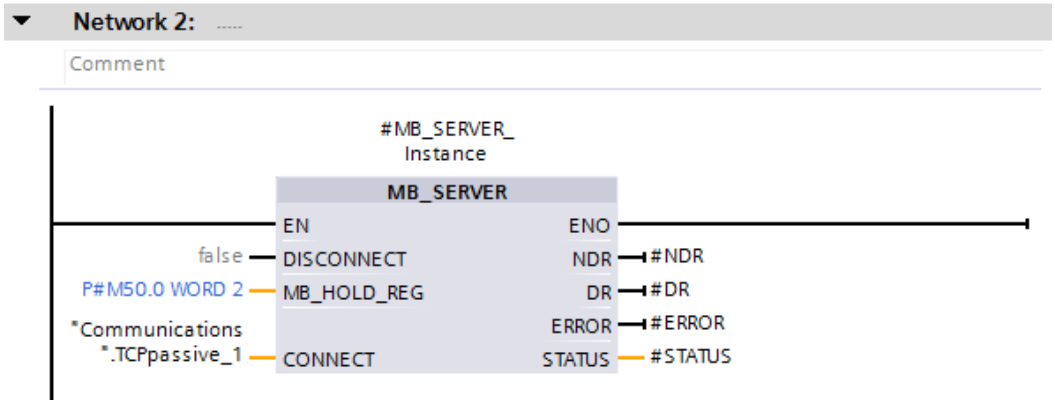

Ilustração A.1.5: Configuração do S7-1200 como servidor *Modbus* TCP

Na Ilustração A.1.6 é revelada a configuração das operações de escrita e leitura das variáveis globais do *Codesys* através da ligação ao servidor OPC *Codesys*. A ligação enquanto cliente OPC DA é configurada pelo bloco funcional *DA Client*. Cada uma das variáveis a ler ou escrever são mapeadas através dos blocos funcionais *OPC Output Bit* e *OPC Input Bit*.

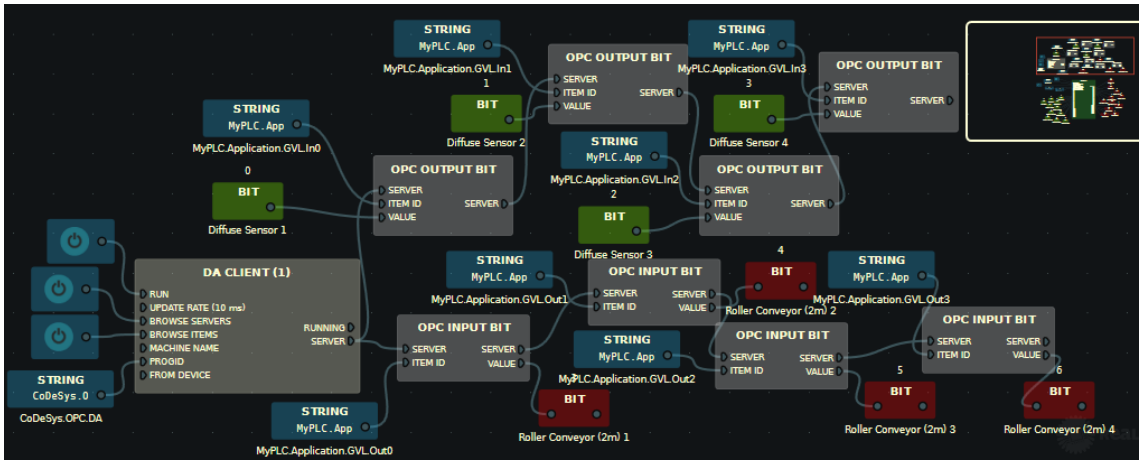

Ilustração A.1.6: Ligação do *Connect I/O* ao servidor OPC *Codesys*

A Ilustração A.1.7 mostra a ligação do *Connect I/O* ao *Siemens* S7-1200. A configuração dos endereços e o mapeamento das variáveis é feito através dum único bloco funcional "S7-1200".

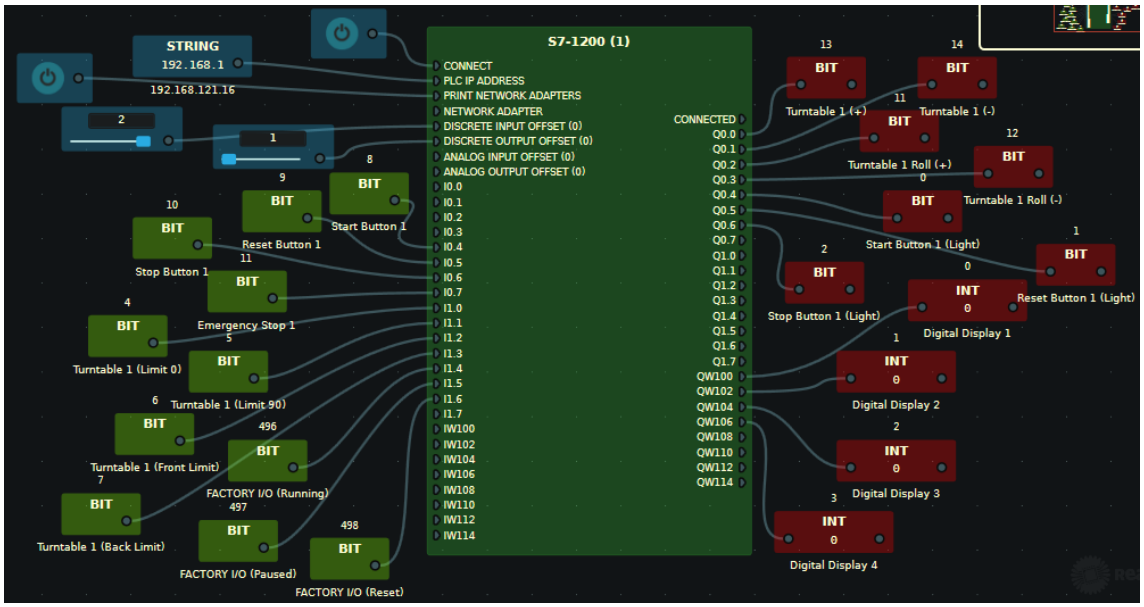

Ilustração A.1.7: Ligação do *Connect I/O* ao *Siemens* S7-1200 por *Ethernet*