

Modelação e simulação de um sistema crossdocking com AS/RS integrado

JOSÉ LUÍS DA CUNHA ALVES

setembro de 2021

Modelação e simulação de um sistema crossdocking com AS/RS integrado

José Luís da Cunha Alves



Mestrado em Engenharia Eletrotécnica e de Computadores

Área de Especialização de Sistemas e Planeamento Industrial

Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

2021

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Unidade Curricular de Tese/Dissertação, do 2º ano, do Mestrado em Engenharia Eletrotécnica e de Computadores

Candidato: José Luís da Cunha Alves, Nº 1141279, 1141279@isep.ipp.pt
Orientação científica: Manuel Fernando dos Santos Silva, mss@isep.ipp.pt



Mestrado em Engenharia Eletrotécnica e de Computadores

Área de Especialização Sistemas e Planeamento Industrial

Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

8 de setembro de 2021

Agradecimentos

A presente Tese/Dissertação não representa apenas um documento de um projeto académico mas também o final de uma das etapas que serão lembradas para o resto da vida. Por isto, devo agradecimentos a várias pessoas que sempre me ajudaram e apoiaram em diversos momentos deste percurso.

Primeiramente, ao orientador desta dissertação, o Professor Doutor Manuel Silva, por todo o apoio, atenção e tempo disponibilizados. A sua visão sobre o tema ajudou-me a seguir o caminho mais correto para encarar este desafio e concluí-lo com sucesso. Aqui deixo o meu sincero agradecimento.

A todos os meus amigos e colegas, por todo o apoio, ajuda, paciência, não só durante esta fase final mas ao longo de todo o ciclo de estudos. Experiências inesquecíveis, que só nós sabemos, marcaram estes últimos anos. Memórias construídas entre as cidades de Viseu e do Porto serão sempre guardadas, e por isso vos agradeço. Mais virão.

À minha família, por garantirem sempre o meu bem estar, quer em alturas menos ou mais complicadas e por serem os principais contribuintes do sucesso que obtive até ao momento e do que está para vir, pessoal e profissionalmente. Por todo o esforço e paciência, um muito obrigado.

Por fim, um agradecimento ao Instituto Superior de Engenharia do Porto, que me acolheu ao longo da minha licenciatura e do mestrado, por todas as experiências e atividades proporcionadas ao longo dos últimos anos.

A todos os mencionados e a outros que apesar de não o terem sido, ficarão na minha memória, obrigado.

Resumo

Atualmente, devido maioritariamente ao avanço dos diversos sistemas tecnológicos e à maior facilidade em adquirir bens, a sociedade tornou-se mais exigente. Devido a este fator, a necessidade das empresas seguirem a mesma trajetória dos consumidores, para que a harmonia entre ambos esteja nivelada, levou a uma mudança na execução de processos e operações. A gestão de cadeias logísticas não foi exceção e surgiu a necessidade da criação de sistemas mais eficientes, tal como os sistemas de *crossdocking*, os quais promovem a movimentação rápida de produtos em que o armazenamento temporário dos mesmos deve ser o mínimo possível. Assim, o principal objetivo passa pela modelação e simulação de um sistema *crossdocking* com um armazém automático integrado, eficiente, oferecendo mais uma solução na área da logística de movimentação de produtos.

No âmbito da presente Tese/Dissertação, apresenta-se o desenvolvimento de todo o trabalho, desde o estudo teórico dos conceitos até às conclusões finais sobre os resultados obtidos. O trabalho passou também por outras fases, devidamente identificadas e de conteúdo relevante. Através de *layouts* definidos, e após uma implementação do modelo no *software* utilizado, a obtenção de resultados, tendo em conta algumas variações a nível de componentes, teve sempre o foco dirigido nos sistemas deste tipo que se encontram em funcionamento em contexto real. Uma aproximação a este tipo de sistemas torna a simulação, embora virtual, bastante mais credível. Várias análises foram executadas para que os resultados gerados sejam abrangentes e completos.

Palavras-Chave: FlexSim, Logística, Cadeia de abastecimento, *Crossdocking*, AS/RS, Armazenamento, Ordens de despacho.

Abstract

Currently, mostly due to the advance of the technological systems, digitalization and the greater ease in acquiring goods, the society has become more demanding. Due to this factor, companies need to start to follow the same line of thinking as consumers, so that harmony between both are stabilized. This can lead to a change in the way how processes and operations are executed. The supply chain management was not an exception and it was necessary to create more efficient systems, such as crossdocking systems, which promote the fast movement of products in a distribution center. In these systems, the products can be stored with the condition that the storage time is minimum and can not exceed a certain period of time. Thus, the main goal is to model and simulate a crossdocking system with an integrated and efficient automatic warehouse. With this model, another solution can be designed and it is meant to help with eventual logistic problems of this type.

This thesis presents the development of the entire work executed, starting in the theoretical study of the concepts, ending with the final conclusions based on the results obtained. The work also went through other phases, all of them identified and with relevant content. Through defined layouts, and after an implementation of the model in the software used, the results were generated taking into account some variations at the level of simulation components, always focused on the existing systems in real context. An approach of this type of systems makes the simulation, although virtual, much more credible and reliable. Several analyzes were performed so that the results generated are comprehensive and complete.

Key Words: FlexSim, Supply Chain, Crossdocking, AS/RS, Storage Systems, Dispatch Orders.

Índice

Agradecimentos	v
Índice	i
Índice de Figuras	iii
Índice de Tabelas	vii
Acrónimos	ix
1 Introdução	1
1.1 Contextualização	1
1.2 Motivação	2
1.3 Objetivos e metodologia	3
1.4 Calendarização	4
1.5 Organização da Tese/Dissertação	6
2 Fundamentação Teórica	9
2.1 <i>Crossdocking</i>	9
2.1.1 Características do crossdocking	13
2.1.2 Tipos de <i>crossdocking</i>	19
2.1.3 Tipos de armazenamento em sistemas <i>crossdocking</i>	20
2.1.4 Calendarização de veículos	21
2.1.5 Indicadores de desempenho	22
2.2 Automated Storage and Retrieval Systems	23
2.2.1 Tipos de AS/RS	24
2.2.2 <i>Projeto</i> do AS/RS	27
2.2.3 Políticas de armazenamento	30
2.2.3.1 Armazenamento baseado em classes	30
2.2.4 Sistema de recolha de pedidos	32

2.2.5	Processamento em lote	33
2.2.6	Sequenciação de pedidos de armazenamento e recuperação	34
2.2.7	Tempo de resposta de um AS/RS	35
2.3	Conclusão do Capítulo 2	37
3	Simulação para Processos Industriais	39
3.1	<i>Softwares</i> de simulação existentes/adequados	39
3.2	Introdução ao FlexSim	43
3.3	Conclusão do Capítulo 3	46
4	Arquitetura do Modelo de Simulação	47
4.1	Arquitetura do sistema de <i>crossdocking</i>	48
4.2	Arquitetura do armazém automático (AS/RS)	51
4.3	Conceitos e considerações finais sobre a arquitetura dos sistemas	55
4.4	Conclusão do Capítulo 4	57
5	Desenvolvimento e Implementação	59
5.1	Implementação do sistema <i>crossdocking</i>	60
5.1.1	Implementação inicial do <i>layout</i> e fluxo desejado	61
5.1.2	Estruturação e desenvolvimento do <i>Process Flow</i>	67
5.1.3	Estruturação e desenvolvimento do <i>Order Process Flow</i>	70
5.2	Implementação do AS/RS	83
5.3	Integração do sistema <i>crossdocking</i> com o AS/RS	88
5.3.1	Enquadramento da integração entre sistemas	88
5.3.2	Aplicação para a integração entre sistemas	90
5.4	Conclusão do Capítulo 5	94
6	Cenários de Atividade e Resultados Obtidos	99
6.1	Cenários de atividade	99
6.1.1	Cenário 1: 6 UA com <i>delay</i> entre ordens	102
6.1.2	Cenário 2: 6 UA sem <i>delay</i> entre ordens	104
6.1.3	Cenário 3: 3 UA com <i>delay</i> entre ordens	106
6.1.4	Cenário 4: 3 UA sem <i>delay</i> entre ordens	108
6.2	Resultados finais	109
6.3	Conclusão do Capítulo 6	113
7	Conclusões	115
7.1	Principais conclusões	115
7.2	Possíveis desenvolvimentos futuros	116
	Referências Bibliográficas	119

Índice de Figuras

1.1	Desenvolvimento faseado da simulação	4
1.2	Gantt <i>chart</i> - plano de trabalhos de TEDI	5
2.1	Aplicabilidade de sistemas <i>crossdocking</i> - adaptado de [1]	11
2.2	Esquema de armazém de consolidação - adaptado de [1]	14
2.3	Esquema de armazém de carga fraccionada - adaptado de [1]	14
2.4	Esquema de armazém misto - adaptado de [1]	14
2.5	Exemplo de forma em I - adaptado de [2]	16
2.6	<i>Crossdocking</i> a dois toques [3]	19
2.7	<i>Crossdocking</i> a múltiplos toques [3]	20
2.8	Classificação de AS/RS - guindastes - adaptado de [4]	25
2.9	Classificação de AS/RS - manuseamento - adaptado de [4]	25
2.10	Classificação de AS/RS - prateleiras - adaptado de [4]	25
2.11	Projeto de AS/RS - adaptado de [4]	28
2.12	Posicionamento de 3 classes em forma de L (parte superior) e em forma retangular (parte inferior) [4]	31
2.13	Estrutura comum de processamento em lote - adaptado de [4]	34
2.14	AS/RS com armazenamento baseado em classes. Classes A,B,C [5]	36
3.1	Página inicial na criação de um modelo no FlexSim	44
3.2	Exemplo de interligação entre o modelo 3D e o <i>Process Flow</i>	44
3.3	<i>Software</i> de simulação 3D orientado a objetos - FlexSim - Exemplo prático 1	45
3.4	<i>Software</i> de simulação 3D orientado a objetos - FlexSim - Exemplo prático 2	45
4.1	Arquitetura do sistema de <i>crossdocking</i> a simular	49
4.2	Representação da interligação entre o <i>crossdocking</i> e o AS/RS	52
4.3	Representação do AS/RS - vista do topo 3D	53
4.4	Representação do AS/RS - vista de perspetiva 3D	54

4.5	Menu dedicado à estrutura de armazenamento de uma <i>rack</i>	54
5.1	Organização do Capítulo 5	60
5.2	Itens utilizados na fase inicial da simulação	61
5.3	<i>A-Connections</i> e <i>S-Connections</i> entre elementos	62
5.4	Criação de itens diferenciados por forma e cor	63
5.5	Definição de parâmetros no DP1	64
5.6	<i>Triggers</i> ‘ <i>On Creation</i> ’ e ‘ <i>On Arrival</i> ’ na distribuição por formato . .	65
5.7	<i>Photo eyes</i> utilizados no modelo	65
5.8	Exemplo de tabela de ordens de despacho importada do Microsoft Excel	67
5.9	Colocação estratégica dos DP1 e DP11	68
5.10	Lógica a implementar no funcionamento inicial do <i>Process Flow</i> . . .	69
5.11	<i>Process Flow</i> generalizado para o sistema <i>crossdocking</i>	70
5.12	Visão geral do <i>Order Process Flow</i>	71
5.13	Fluxo de obtenção de dados da tabela de ordens de despacho	72
5.14	Distribuição estatística utilizada na chegada de ordens de despacho ao sistema	72
5.15	Fluxo de distribuição de ordens de despacho pelas UA destinadas . . .	73
5.16	Propriedades da atividade ‘ <i>Wait for Event: Conveyor Arrival</i> ’	74
5.17	Transformação <i>Processor</i> -> <i>Wrapper</i>	75
5.18	Implementação de robôs de transporte, paletizadores e máquinas de embrulho (<i>wrappers</i>)	76
5.19	Lógica aplicada ao bloco responsável pelo transporte de itens entre UA e <i>combiners</i>	77
5.20	Funcionamento da atividade ‘ <i>Decide: Items in Queue?</i> ’	78
5.21	Aplicação de três robôs no transporte de itens no <i>crossdocking</i>	79
5.22	Alteração de fluxo no bloco 3 do OPF	80
5.23	Alteração de atividades no bloco 3	81
5.24	Associação entre elementos na concretização de ordens de despacho . .	82
5.25	Objetos/instâncias utilizadas na lógica do OPF	83
5.26	Alteração de itens utilizados na simulação	84
5.27	Distribuição de locais de armazenamento por <i>rack</i>	86
5.28	Distribuição de itens armazenados por <i>label</i> numa <i>rack</i>	87
5.29	Colocação estratégica de <i>decision points</i> na distribuição de itens para armazenamento	88
5.30	Localização estratégica do DP9	90
5.31	Criação de um <i>label</i> para definição de prioridade de itens na lista dos mesmos	91
5.32	Ligações existentes na lógica do <i>Process Flow</i> generalizado	92
5.33	Bloco do OPF de recuperação de itens armazenados no AS/RS	93
5.34	Reestruturação do OPF para integração do AS/RS com o <i>crossdocking</i>	94
5.35	Janela <i>pop-up</i> de acompanhamento da lista de itens	94

5.36	Visão geral do modelo da simulação	96
5.37	Visão geral do <i>Order Process Flow</i>	97
6.1	Exemplo de tabela de ordens utilizada na simulação	100
6.2	Resultados do Cenário 1 relativos ao sistema <i>crossdocking</i>	102
6.3	Resultados do Cenário 1 relativos ao AS/RS	103
6.4	Resultados do Cenário 2 relativos ao sistema <i>crossdocking</i>	104
6.5	Resultados do Cenário 2 relativos ao AS/RS	105
6.6	Resultados do Cenário 3 relativos ao sistema <i>crossdocking</i>	106
6.7	Resultados do Cenário 3 relativos ao AS/RS	107
6.8	Resultados do Cenário 4 relativos ao sistema <i>crossdocking</i>	108
6.9	Resultados do Cenário 4 relativos ao AS/RS	109

Índice de Tabelas

2.1	Fatores de decisão - caracterização de AS/RS	29
3.1	Comparação de <i>softwares</i> de simulação	42
5.1	Valores associados ao <i>label</i> ‘Type’ dos itens criados nas <i>sources</i> do <i>crossdocking</i> por cor	62
5.2	Valores associados ao <i>label</i> ‘Type’ dos itens criados nas <i>sources</i> do <i>crossdocking</i> por forma	62
5.3	<i>Label</i> associado aos elementos UA e <i>Robot</i>	80
5.4	<i>Label</i> associado aos paletizadores	81
5.5	Cor correspondente a cada <i>label</i> no armazenamento dedicado	85
5.6	Tabela de tempos de chegada de itens ao sistema	89
6.1	Quatro cenários distintos simulados	102
6.2	Síntese dos resultados obtidos nos quatro cenários simulados	110

Acrónimos

Acrônimo	Descrição	Página
ISEP	Instituto Superior de Engenharia do Porto	1
TEDI	Tese/Dissertação	5
AS/RS	<i>Automated Storage and Retrieval Systems</i>	9
JIT	<i>Just-in-Time</i>	10
EDI	<i>Electronic Data Interchange</i>	12
SCM	<i>Shipping Container Marking</i>	12
UPC	<i>Universal Product Code</i>	12
FTL	<i>Full Truck Loads</i>	13
LTL	<i>Less Truck Loads</i>	13
FCFS	<i>First-Come-First-Served</i>	22
COI	<i>Cube-per-order index</i>	30
OPS	<i>Order Picking System</i>	32
OP	<i>Order Picking</i>	32
VC	<i>Visual Components</i>	40
API	<i>Application Programming Interface</i>	41
3D	<i>Três Dimensões</i>	43
AGV	<i>Automated Guided Vehicles</i>	49
UMP	<i>Unidades Móveis de Paletização</i>	49
PF	<i>Process Flow</i>	56
UA	Unidades Acumuladoras	61
DP	<i>Decision Points</i>	63
PE1	<i>Photo Eye 1</i>	65
PE2	<i>Photo Eye 2</i>	65
OPF	<i>Order Process Flow</i>	66
CS	<i>Conveyor System</i>	67

Capítulo 1

Introdução

O presente documento foi elaborado no contexto da unidade curricular Tese/Dissertação do 2º ano do Mestrado em Engenharia Eletrotécnica e de Computadores, ramo de Sistemas e Planeamento Industrial, do Instituto Superior de Engenharia do Porto (ISEP), tendo como principal objetivo o estudo e a simulação de um sistema *crossdocking* integrado com um armazém automático. Neste documento é apresentado todo o desenvolvimento do trabalho realizado, que engloba uma fundamentação teórica de ambos os temas principais (*crossdocking* e armazéns automáticos), um estudo relativo aos *softwares* existentes no mercado capazes de simular o pretendido, uma descrição sucinta da implementação de ambos os sistemas, seguido de uma demonstração de resultados e testes realizados tendo em conta diferentes cenários. Finaliza-se com uma conclusão do estudo e da simulação executada.

Posto isto, neste capítulo será realizada uma contextualização e motivação do projeto, uma exposição dos objetivos traçados e uma calendarização de modo a exibir a linha cronológica do desenvolvimento do projeto. É também apresentada a organização da dissertação.

1.1 Contextualização

O avanço tecnológico cada vez mais notado nos tempos que correm, leva não só a um maior desenvolvimento de processos e operações mas também a uma maior exigência por parte dos consumidores. A logística tradicional está a transformar-se cada vez mais e, atualmente, um dos maiores desafios remete para a entrega de produtos no menor tempo possível, não só com o intuito de reduzir os custos associados mas também para aumentar a satisfação do cliente, quer este seja individual ou coletivo. A gestão de cadeias de distribuição, conceito mundialmente

conhecido como *Supply Chain Management*, é um aglomerado de processos cada vez mais complexo, que envolve direta ou indiretamente tudo o que se relaciona com um pedido de um cliente. Para que haja uma envolvimento eficiente deste tema, as empresas de distribuição têm a obrigação de acompanhar a exigência dos consumidores e, para isso, recorrem a tecnologias de ponta e a novas metodologias, de forma a satisfazerem o que lhes é requerido. Este é o seu grande desafio.

Sistemas de movimentação rápida de produtos, tal como sistemas de *cross-docking*, têm como objetivo facilitar a distribuição dos mesmos ao mesmo tempo que os custos de armazenamento, de transporte, entre outros, são suprimidos. Porém, nem sempre é possível encaminhar todos os produtos desde a sua entrada num centro logístico diretamente para a zona de expedição e de envio para o destino que lhes foi traçado. Por vezes, é necessário recorrer a um armazenamento temporário, normalmente inferior a 24 horas. Isto não suprime os custos de armazenamento na totalidade, mas é capaz de os reduzir substancialmente. Assim, sistemas como este são atualmente utilizados para fazer face às exigências do mercado e respetivos clientes, tornando-se assim num ativo com extrema importância na área da logística, aliado à evolução tecnológica. Diversos centros logísticos armazenam produtos temporariamente devido a diversos fatores, como por exemplo eventuais atrasos dos veículos de entrada (trazem os produtos para o centro) ou de saída (transportam os produtos do centro para o seu destino), ou a algum problema interno no fluxo de operações. Este armazenamento é praticado de maneiras diferentes, conforme as necessidades e possibilidades dos centros logísticos. Frequentemente, dois tipos de armazenamento são utilizados, dependendo dos recursos de cada centro. Os produtos podem ser armazenados no chão do mesmo, numa área destinada para tal, ou armazenados em sistemas de prateleiras de forma automática, conhecidos como *Automated Storage and Retrieval Systems*, ou armazéns automáticos. Este tipo de armazenamento automático é aplicável a sistemas de *crossdocking*.

De modo a explorar este tema integrado de sistemas *crossdocking* com armazenamento temporário automático, foi desenvolvido um modelo utilizando o *software* FlexSim, concedendo mais uma hipótese para dar resposta a eventuais problemas existentes no mercado e na área da logística e distribuição.

1.2 Motivação

A escolha deste tema para a realização da presente Tese/Dissertação foi impulsionada pelo desafio que o mesmo propõe. Pessoalmente, como aluno do mestrado de Engenharia Eletrotécnica no ISEP, a diversidade de conceitos associados a sistemas e planeamento industrial fornece uma visão abrangente da situação atual em várias áreas. Sendo a logística uma área complexa e em constante desenvolvi-

mento, surgiu a oportunidade de criar, modelar e simular um sistema *crossdocking* com um armazém automático incorporado.

Por outro lado, toda a complexidade associada a estes sistemas, a cada vez maior inserção da tecnologia nos sistemas logísticos, alterando o panorama da logística tradicional e das cadeias de abastecimento, gera uma curiosidade acrescida nestes temas e conceitos. Como consumidor, perceber o funcionamento, os processos e as operações existentes por detrás da movimentação de produtos, é enriquecedor e permite alargar a visão existente sobre logística. O desenvolvimento das cadeias de abastecimento nas últimas décadas, devido à digitalização e à recente facilidade de adquirir produtos *online*, também contribuiu como motivação para um estudo mais aprofundado acerca da forma como os mesmos são distribuídos e os processos que sofrem durante essa operação.

Toda esta curiosidade aliada à possibilidade de modelar e simular um sistema exequível em contexto real, contribuíram como motivação maior para o desenvolvimento desta Tese/Dissertação, oferecendo um conhecimento extra sobre esta área e sobre *softwares* de simulação.

1.3 Objetivos e metodologia

Quanto ao objetivo primordial do trabalho, passa pela modelação e simulação de um sistema de *crossdocking* capaz de armazenar produtos temporariamente recorrendo a um AS/RS, através de um *software* de simulação que permitisse criar um ambiente mais próximo do contexto real. Este objetivo encarrega-se simultaneamente ao incentivo e desenvolvimento de competências nas áreas de simulação e otimização de processos industriais. Devido aos temas a que o presente trabalho se encontra associado, não é considerada uma simulação generalizada a diversos processos industriais, embora se encontre relacionado com a área. No meio da logística, mais propriamente em centros de distribuição, a presente simulação é capaz de se adaptar aos diversos fatores associados e oferecer uma solução válida a eventuais problemas na transição de produtos entre uma origem e um destino.

Visto que dois temas se encontram em estudo, objetivos diferentes se dirigem a cada um e, num momento final, agrupam-se de forma a atingirem um objetivo comum. Relativamente ao *crossdocking*, o objetivo principal do seu funcionamento é a circulação rápida de produtos a partir do momento que entram no sistema até ao momento em que são expedidos, cumprindo as ordens de despacho emitidas. Outros objetivos se encontram ligados, como o estudo dos tipos de *crossdocking* existentes e as suas características mais influentes. Quanto ao AS/RS, o seu objetivo principal passa por apoiar o sistema *crossdocking* no que diz respeito à necessidade de armazenamento temporário de produtos. Caso este tipo de armazém não estivesse presente, os processos seriam bem mais demorados e desorganizados, retirando eficiência ao processo. O estudo de políticas de

armazenamento e de configurações exequíveis para uma dada dimensão revela-se anexado ao objetivo principal. Toda a evolução da integração destes dois sistemas foi gradual e exigente.

No que diz respeito à metodologia usada para o desenvolver do trabalho, esta é distinta por duas etapas, teórica e prática. Numa fase inicial foi necessário o estudo dos diversos conceitos associados, quer a sistemas *crossdocking*, quer a sistemas de armazenamento automático, e de que forma podem ser integrados. Após o estudo dos conceitos, os primeiros passos na utilização do *software* FlexSim revelaram-se de extrema importância, em jeito de entrada para a parte prática da dissertação. Seguidamente a esta familiarização com a plataforma de simulação, uma abordagem ao *layout* de ambos os sistemas foi realizada de forma a perceber qual o mais benéfico e adequado para a simulação, o que levou posteriormente à implementação dos mesmos e ao desenvolvimento de todos os processos e fluxos associados à simulação, de forma faseada como é possível verificar na Figura 1.1. Completado o modelo, testes foram realizados tendo em conta diversas abordagens e concluída a simulação.

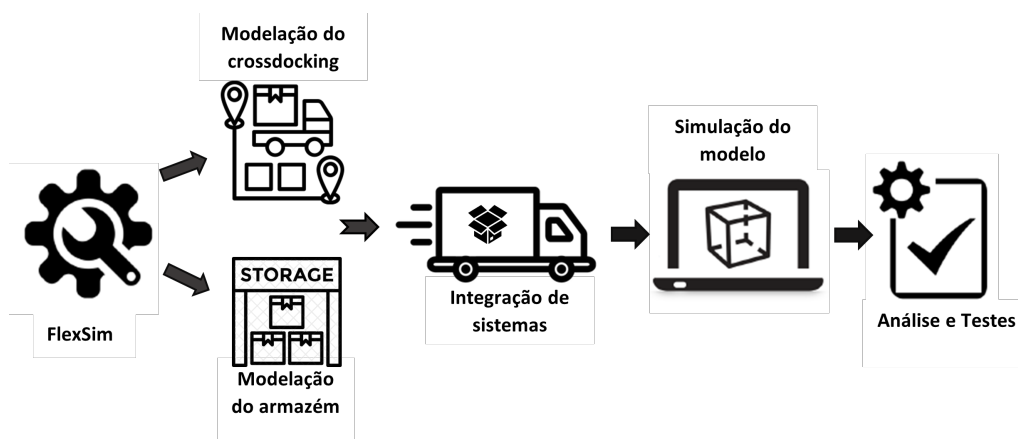


Figura 1.1: Desenvolvimento faseado da simulação

1.4 Calendarização

De forma a calendarizar o desenvolvimento das atividades realizadas que levaram à realização da presente Tese/Dissertação, foi criado um diagrama de Gantt [6]. Este contempla todas as principais atividades desenvolvidas, a sua data de início e de conclusão e uma representação gráfica das mesmas numa linha temporal. O diagrama é representado na Figura 1.2. Como é possível verificar, existe uma divisão mensal e semanal, sendo que o quadro de atividades se encontra dividido à semana. Isto permite identificar em que semana se iniciou e finalizou uma

determinada atividade, as quais estão representadas pela cor azul no diagrama.

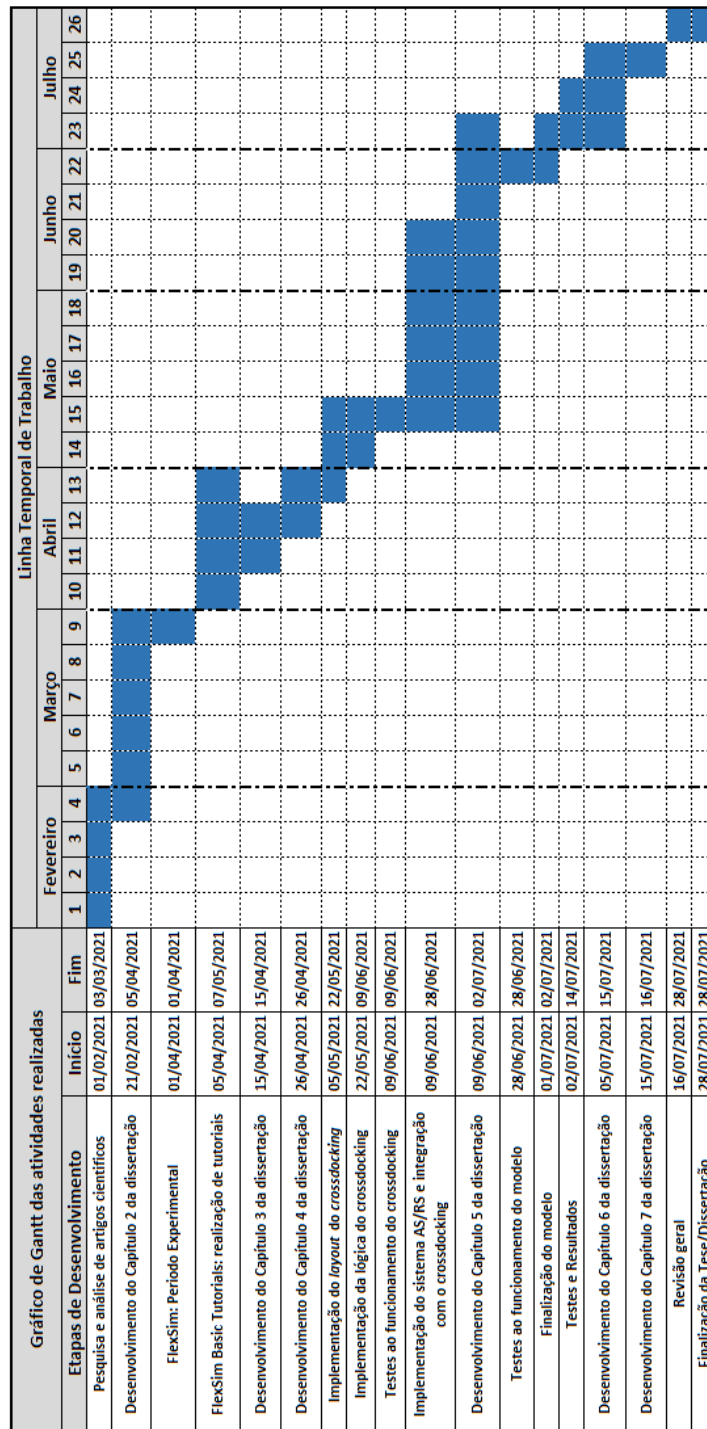


Figura 1.2: Gantt *chart* - plano de trabalhos de TEDI

A presente Tese/Dissertação foi iniciada no dia 1 de Fevereiro de 2021. A

partir desse momento, foram vinte e seis semanas de trabalho, desde o início da exploração do tema até a uma revisão geral de todo o trabalho desenvolvido, quer teórico, quer prático. Várias etapas foram realizadas, sempre com o foco de um projeto equilibrado e desenvolvido de forma gradual. O estudo dos conceitos associados ao tema principal facilitou a implementação do sistema no FlexSim, bem como a realização de tutoriais de funcionamento do mesmo. A parte final dirigiu-se à recolha e análise de resultados, de acordo com parâmetros definidos, o que permitiu retirar as conclusões necessárias para uma avaliação global do trabalho. Por fim, realizou-se uma revisão geral de tudo o que foi desenvolvido e descrito. Desta forma, o projeto terminou no dia 28 de Julho de 2021.

1.5 Organização da Tese/Dissertação

A presente Tese/Dissertação é composta por sete capítulos principais, de forma a que o trabalho realizado seja exposto de forma lógica e gradual, devidamente organizado. Como introdução, o Capítulo 1 apresenta uma contextualização ao que foi desenvolvido e demonstra os objetivos e a linha cronológica em que foi realizado o trabalho.

No Capítulo 2 são explorados conceitos teóricos dos dois temas em estudo e apresentadas as referências bibliográficas que suportam os mesmos. Os conceitos mais relevantes para o desenvolvimento da simulação estão divididos em duas partes, uma relacionada com o *crossdocking* e outra com o armazém automático.

No Capítulo 3 é realizada uma abordagem aos *softwares* existentes no mercado, e uma avaliação às suas propriedades para definir qual o mais apto para executar a modelação e simulação de um sistema *crossdocking* com um armazém automático integrado. Foram exploradas as principais funcionalidades dos *softwares* estudados e nomeado o de utilização mais propícia e acessível.

O Capítulo 4 é referente ao *layout* do modelo, tanto do *crossdocking* como do armazém automático. É exposta a sua disposição, o seu fluxo e os principais recursos que são necessários incluir.

O Capítulo 5 aborda os passos realizados para que a modelação e simulação do sistema evolua. Ou seja, desde o momento em que se deram os primeiros passos na implementação do *layout* até se concluir a simulação, com os sistemas integrados e o fluxo de produtos funcional segundo as ordens de despacho existentes. Considera-se que é o principal capítulo da presente Tese/Dissertação pois descreve os atividades, conceitos e o principal conteúdo desenvolvido na parte prática.

No Capítulo 6 são criados alguns cenários, de forma a perceber a eficiência do sistema tendo em conta a falta de alguns elementos e recursos, qual o seu comportamento e de que forma estas alterações afetam o desempenho dos siste-

mas. Assim, realizam-se vários testes, através dos quais se obtêm resultados do trabalho desenvolvido.

O Capítulo 7 é destinado à conclusão de todo o trabalho desenvolvido, fazendo uma ponte de ligação entre conceitos teóricos e práticos, bem como uma apreciação geral do trabalho.

Capítulo 2

Fundamentação Teórica

O papel da logística nos dias que correm revela-se de extrema importância, sendo um dos principais negócios aliados à digitalização. Isto permite uma maior ligação entre o fornecedor e o cliente final, o qual pretende ter o produto que adquiriu na sua posse no menor tempo possível. Para que tal aconteça, diversos sistemas se interligam com o objetivo de satisfazer o cliente. A movimentação rápida de produtos é um dos principais focos na área logística, sendo o aprimoramento das suas operações e fluxos um ponto de melhoria. A utilização de vários sistemas tecnológicos, cada vez mais avançados, tem-se revelado imprescindível e, para que não ocorram falhas ou para que as mesmas sejam diminuídas ao máximo, é utilizada a simulação dos próprios sistemas em *softwares* apropriados para tal. Porém, uma simulação requer um estudo prévio de conceitos teóricos para que, quando aplicados na prática, se revelem conformes. Assim, o desenvolvimento da simulação de um modelo prático, com o intuito de replicar algo aplicável em contexto real, necessita de fundamentos teóricos para todos os temas que se encontram associados. No caso da presente dissertação, o estudo de temas como *crossdocking* e sistemas de armazenamento automático, nomeadamente *Automated Storage and Retrieval Systems*(AS/RS) é extremamente importante para uma boa simulação.

2.1 Crossdocking

Com o aumento constante da necessidade dos clientes e consumidores em ter bens e produtos ao seu alcance no menor período de tempo, foi necessário encontrar soluções para os movimentar. Tradicionalmente, os centros de distribuição funcionam como armazéns, onde recebem, armazenam e expedem os bens para os respetivos clientes ou consumidores. Atualmente esta metodologia ainda é

utilizada, porém, como o armazenamento de produtos tem custos associados, novas metodologias surgiram para os diminuir e, se possível, suprimi-los. Surge o *crossdocking*. É um método de distribuição com o intuito de minimizar significativamente, ou mesmo acabar com os custos de armazenamento.

Crossdocking é uma estratégia logística bastante utilizada por diversas empresas dos mais variados setores económicos, aplicada nos armazéns e centros de distribuição. O seu principal objetivo passa por receber produtos provenientes dos veículos de entrada (*inbound trucks*) e encaminhá-los diretamente para o seu local de expedição, onde se encontram os veículos de saída (*outbound trucks*), sem que haja, idealmente, armazenamento intermédio, aumentando assim a eficiência das operações [1, 3]. A chave para o sucesso passa por ter o menor período de permanência possível nas instalações de receção e expedição dos produtos. Quanto menor esse período menos tempo de armazenamento temporário é necessário. Na prática, o ideal poucas vezes se verifica, sendo assim necessário um armazenamento temporário limitado e minimizado ao máximo, por exemplo através de um *Automated Storage and Retrieval Systems*. Veículos de entrada por norma transportam produtos de um determinado fornecedor, os quais ao serem colocados num sistema de *crossdocking* são consolidados juntamente com produtos provenientes de outro veículo e, possivelmente, de outro fornecedor. Esta consolidação permite que produtos de diversas origens sejam agrupados e distribuídos para diferentes destinos finais.

No setor dos transportes o principal objetivo passa por movimentar um produto desde o seu fornecedor ao cliente ou consumidor com o mínimo de danos, custo e tempo possível. As estratégias a utilizar têm um grande impacto no negócio, o que levou muitas empresas a atualizarem os seus centros de distribuição tipicamente tradicionais para sistemas *crossdocking*. Para além de acompanharem a evolução da tecnologia e inovação, possibilitou-se uma melhoria dos seus processos internos, criando assim uma grande diferença entre estes dois tipos de sistemas. Centros de distribuição tradicionais, por norma, têm inventário disponível para enviar aos seus clientes ao invés de um sistema *crossdocking* em que o objetivo é precisamente o oposto. Ter inventário reduzido ao máximo ou simplesmente não ter, focando-se em processos de entrega *Just-In-Time* (JIT) [7]. Comparando estes dois sistemas, é possível identificar diversas vantagens [3] do *crossdocking*:

- redução de custos (custos de armazenamento, custos de posse do inventário, custos de manuseamento);
- diminuição do tempo de entrega através da simplificação de processos;
- serviço de cliente melhorado;
- redução do espaço de armazenamento;

- risco de perda (a identificação dos produtos em cada sistema diminui o risco de perda consideravelmente).

Sistemas logísticos como o *crossdocking* tornaram-se uma estratégia importante no que diz respeito à melhoria do desempenho dos serviços de distribuição, à redução do inventário existente nos armazéns e possibilitaram uma maior capacidade de resposta a pedidos efectuados por parte dos clientes e consumidores. É uma estratégia que ao ser implementada pode servir propósitos diferentes, consoante o objetivo definido. Visto que o *crossdocking* acelera a movimentação de produtos entre a sua origem e destino, a sua implementação pode surgir da necessidade de minimização do tempo de espera na entrega dos mesmos. A questão económica também se fixa como um propósito com bastante influência nos tempos atuais, sendo que um sistema de *crossdocking* possibilita uma redução de custos (custos de *stock* e distribuição) caso seja bem aplicado. É necessário um investimento inicial que terá um risco associado, porém uma boa aplicação irá gerar resultados positivos a longo prazo [8].

Existem fatores cruciais no momento da decisão com vista a questionar se é preferível um sistema *crossdocking* ou um sistema de distribuição tradicional [1]. Entre eles a taxa de procura do produto e o custo de saída por unidade em *stock*. Considera-se que a taxa de procura de um produto pode ser estável/constante ou instável/flutuante. Já o custo de saída por unidade em *stock* pode ser reduzido ou elevado. A Figura 2.1 demonstra a relação existente entre estes fatores e indica qual o sistema preferencial.

		Taxa de procura do produto	
		Estável/Constante	Instável/Flutuante
Custo de saída de unidade em <i>stock</i>	Elevado	<i>Crossdocking</i> pode ser implementado com sistemas próprios e ferramentas de planeamento.	Preferível uma distribuição tradicional.
	Reduzido	Preferível utilizar <i>crossdocking</i> .	<i>Crossdocking</i> pode ser implementado com sistemas próprios e ferramentas de planeamento.

Figura 2.1: Aplicabilidade de sistemas *crossdocking* - adaptado de [1]

Como é perceptível, o *crossdocking* é mais valioso quando a procura é estável

e o custo de saída de unidade em *stock* reduzido. Estável porque para ter um bom funcionamento é necessário haver um balanço entre a carga que entra e a que é expedida e um custo de saída reduzido pois como um sistema *crossdocking* minimiza a existência de inventário, a probabilidade de não ter *stock* disponível é elevada.

O tamanho do edifício onde se pretende instalar o sistema é também um fator a ter em conta e pode estar diretamente relacionado com a taxa de procura dos produtos. Isto porque quanto maior for a procura maior o fluxo de veículos e cargas no sistema, o que requer um espaço que consiga satisfazer o negócio. Para além disso, dois fatores críticos de sucesso nas operações de *crossdocking* são o fluxo físico dos produtos e o fluxo de informação relativa aos mesmos [9]. As operações de movimentação dos produtos, tais como a receção, redistribuição e expedição, representam o fluxo físico, ao qual está associado um fluxo de informação. A variedade dos produtos e o seu volume dificultam a gestão de processos, tornando-se assim num dos principais fatores de risco. De forma a reduzir a imprevisibilidade, no momento da implementação, surge um aspeto adicional de extrema relevância e que influencia diretamente o bom funcionamento do sistema. A carga que chega ao edifício proveniente dos fornecedores deve-se encontrar devidamente identificada, com o seu conteúdo previamente conhecido e com indicação do local a que se destina, através de um documento eletrónico de aviso prévio de envio. São três simples informações que irão permitir uma melhor coordenação e gestão de carga. Para leitura deste tipo de informações são maioritariamente utilizados três tipos de tecnologias de informação imprescindíveis [3, 10]. *Electronic Data Interchange* (EDI), onde há uma troca estruturada de dados eletrónicos entre ou dentro de empresas através de uma rede de dados. *Shipping Container Marking* (SCM), para identificar o contentor ou a carga que entra no sistema e também o *Universal Product Code* (UPC), mais conhecido como código de barras, com a missão de identificar individualmente cada produto.

Tendo estes fatores previamente definidos, antes de se estabelecerem processos, características e outros métodos, é necessário formular as necessidades de cada situação em específico. A implementação de um sistema de *crossdocking* tem que servir os propósitos determinados numa fase embrionária. Visto que a implementação deste sistema pode nem sempre ser vantajosa, é necessário estruturar os principais pontos da mesma. Assim sendo, propõem-se três decisões principais, fundamentadas através de perguntas chave em cada uma delas [3, 9].

1. Decisões Estratégicas

- a) ‘Onde ficará colocado o *crossdock*?’
- b) ‘Qual o melhor *layout* para o *crossdock*?’

2. Decisões Táticas

- a) ‘Como é que os produtos irão circular de forma a minimizar os custos e ao mesmo tempo satisfazer a procura?’

3. Decisões Operacionais

- a) ‘Qual a rota do *crossdock*?’
- b) ‘Qual o melhor tipo de transporte a utilizar dentro do *crossdock*?’

São decisões sequenciais, em que o *input* inicial para a tomada de decisões estratégicas é dado não só pelo propósito mas também pelos fatores referidos. À medida que as perguntas chave são analisadas e respondidas, a sua resposta, ou seja, o seu *output*, serve de *input* para as perguntas a responder no tipo de decisão a tomar em seguida e assim sucessivamente.

2.1.1 Características do crossdocking

O armazenamento tradicional tem dois principais objetivos no armazenamento de cargas. Melhorar o serviço ao cliente ao ter disponível a carga ou produto que necessita perto de si e poupar nos custos de transporte ao utilizar *Full Truck Loads* (FTL). O conceito de FTL remete para um modo de envio em que um veículo carrega uma remessa dedicada, ou seja, o transporte é exclusivamente reservado para uma remessa. Já o seu oposto, *Less Truck Loads* (LTL), indica que o veículo está disponível para transportar mais do que uma remessa. Para consolidar estes objetivos e obter poupanças económicas no transporte, foram idealizados três tipos de armazéns. Armazéns de consolidação, armazéns de carga fraccionada (mais conhecidos como *break-bulk warehouses*) e armazéns mistos [1].

- Armazém de consolidação (Figura 2.2): o material recebido advém de diversos fornecedores em diversos carregamentos LTL. É feita a consolidação dos produtos ou cargas no armazém e são posteriormente expedidos para o seu destino final em carregamentos FTL em remessas a granel (remessas sem acondicionamentos).
- Armazéns de carga fraccionada (Figura 2.3): utilizam uma lógica inversa relativamente aos de consolidação. Os produtos são recebidos em granel através de carregamentos FTL e consolidados nas instalações. De seguida são expedidos em LTL para clientes localizados nas proximidades.
- Armazéns mistos (Figura 2.4): as remessas chegam consolidadas de diversos fornecedores através de carregamentos FTL, onde são divididas e reconsolidadas de forma a criar novas remessas. Cada uma das novas remessas FTL é enviada para o cliente. Este tipo de armazém é uma combinação entre os armazéns de consolidação e de carga fraccionada pois utiliza princípios de ambos.

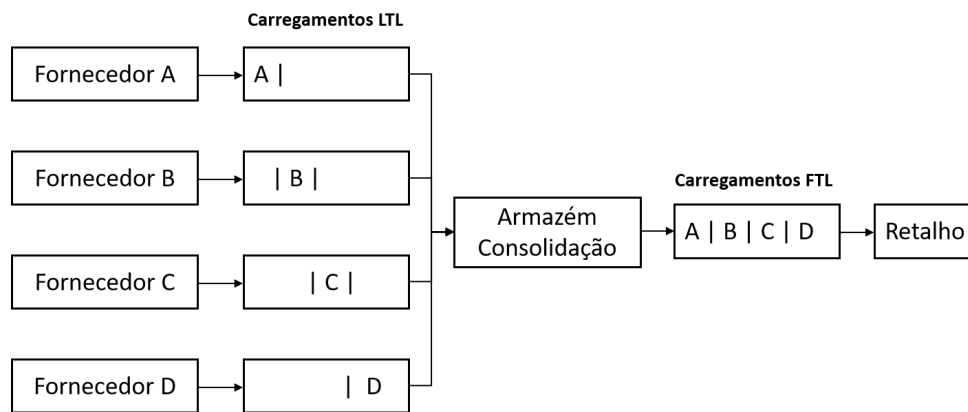


Figura 2.2: Esquema de armazém de consolidação - adaptado de [1]

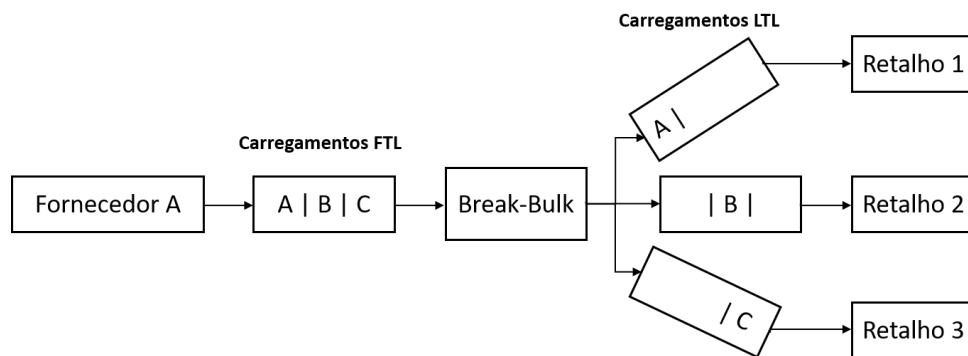


Figura 2.3: Esquema de armazém de carga fraccionada - adaptado de [1]

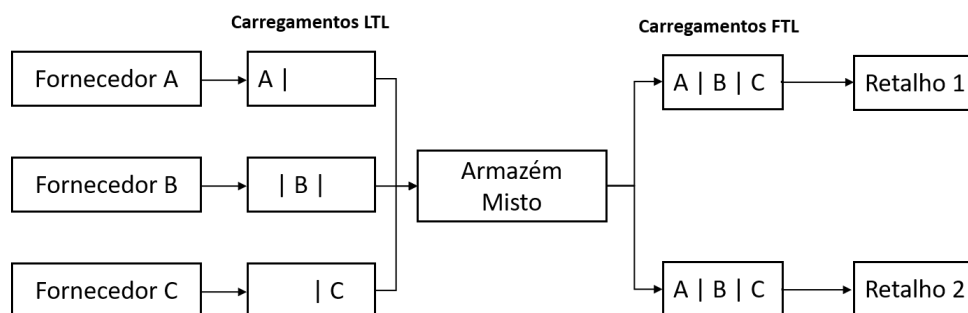


Figura 2.4: Esquema de armazém misto - adaptado de [1]

Em qualquer um dos três tipos de armazéns tradicionais, os produtos permanecem nas instalações durante um longo período de tempo antes de voltarem a ser expedidos, sendo esta a principal diferença entre armazenamento tradicional e *crossdocking*. Conceptualmente, um sistema *crossdocking* é equivalente a um armazém misto, todavia os produtos circulam pelo armazém rapidamente e não

ficam armazenados, evitando a criação de inventário. Em termos de remessas, o *crossdocking* dá prioridade a carregamentos FTL devido a questões maioritariamente económicas. Isto é, caso a remessa tenha sido recebida através de FTL, como a carga que transporta é maior, o tamanho do lote também será, consequentemente, maior. Isto permite uma redução dos custos de configuração do sistema.

As características de sistemas *crossdocking* podem ser utilizadas para diferenciar os vários tipos existentes. Assim, três tipos principais de características se destacam: físicas, táticas e de fluxo de bens/produtos (ótica operacional).

Características **físicas** têm em conta parâmetros como o número de docas, a forma do *crossdocking* e o tipo de transporte interno. São parâmetros que se pressupõem fixos durante um longo período de tempo [9, 11].

- Número de docas: para que seja possível receber os produtos provenientes das demais origens e expedi-los para os respetivos destinos, um sistema *crossdocking* tem que ter em consideração o número de portas ou docas existentes, através das quais os produtos entram e saem do sistema. Normalmente contém entre 6 a 200 docas, exceto alguns sistemas mais raros e complexos que poderão ter até 500 docas. Por norma o número de docas existentes é definido tendo em conta o número de destinos que o sistema irá servir. Caso cada destino seja representado por uma doca, então o sistema poderá optar por ter o número de docas igual ao de destinos. Se para um determinado destino houver um elevado fluxo de produtos poderá ser necessário mais do que uma doca.
- Forma: são conhecidos sistemas *crossdocking* com formas em I, L, T, U, H e E. É uma das características mais influentes devido ao impacto que causa no funcionamento do sistema. Influencia diretamente na suavidade de processos de carga, descarga e movimentação dos produtos. Segundo um estudo ([11]), a forma que apresenta um melhor desempenho e é mais comum é a representada pela letra I ([2]), presente na Figura 2.5. Esta forma apresenta maiores benefícios comparativamente com as restantes maioritariamente devido ao curto tempo de viagem entre docas de receção (*inbound trucks*) e de expedição (*outbound trucks*). Quanto menor o tempo de viagem menor será o custo variável dedicado à viagem dos produtos entre docas. Esta forma também apresenta um maior número de docas convenientes em comparação com as outras formas. Ou seja, há uma menor distância entre docas, sendo as centrais consideradas mais convenientes.

A desvantagem associada à forma I passa pela variação do número de docas. Ao aumentar o número de docas, esta forma tende a perder eficiência pois o diâmetro do *crossdocking* aumenta, o que faz com que os produtos possam

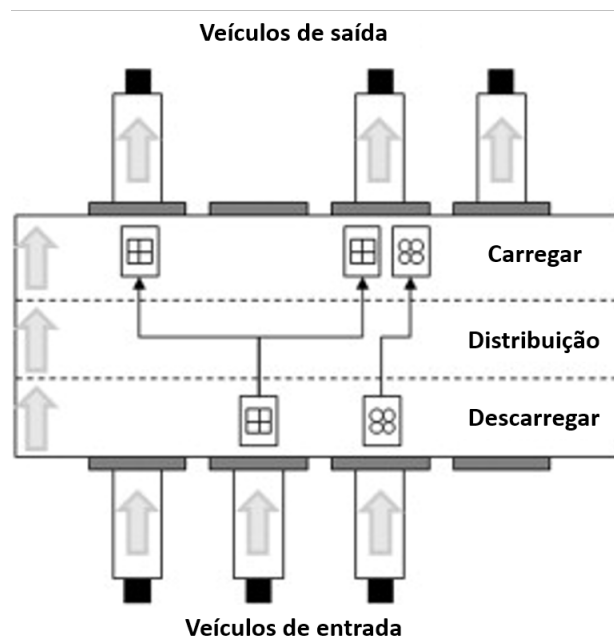


Figura 2.5: Exemplo de forma em I - adaptado de [2]

ter que viajar entre a doca de recepção e a de expedição mais do que o previsto [11].

- Transporte interno dos produtos: pode ser diferenciado por dois tipos de movimentação, manual ou automática. No que diz respeito a uma movimentação manual, pode ser efectuada pelo Homem ou por recurso a empilhadores. Já a movimentação automática é efectuada por tapetes rolantes.

A definição do *layout* é um fator influenciado pela estrutura do próprio edifício.

Quanto às características **táticas**, por vezes são confundidas com decisões táticas. A sua relação é imposta na fase de implementação do *crossdocking* visto que as decisões tomadas irão afetar o funcionamento do mesmo. Essas decisões irão ser traduzidas em características após a implementação. Quatro tipos de características de nível tático se destacam. O modo de serviço, o direito de preferência (*pre-emption*), capacidade de armazenamento temporário e a capacidade de recursos internos. O modo de serviço tem como objetivo determinar os graus de liberdade na atribuição de veículos na entrega e recolha de produtos às docas correspondentes, ou seja, está diretamente relacionado com o tipo de docas a utilizar. Divide-se em três tipos: modo exclusivo, modo misto e modo de combinação [3].

- Modo exclusivo: tal como indica o nome, apresenta docas exclusivas para recepção e docas exclusivas para expedição de produtos. Ou seja, os veículos

que chegam ao edifício com o intuito de descarregar os produtos só o poderão fazer em determinadas docas e o mesmo acontece para veículos com o intuito de carregar os produtos para os entregar no seu destino. Outra abordagem para este tipo de serviço prende-se por alocar uma origem ou destino fixo a uma ou várias docas, sendo caracterizadas por doca de origem exclusiva ou doca de destino exclusivo.

- Modo misto: não há restrições relativas às docas. Qualquer uma pode funcionar como doca de receção ou de expedição de produtos.
- Modo de combinação: funciona como uma agregação do modo exclusivo e do modo misto. Por outras palavras, este modo aceita determinadas docas a funcionar com um serviço exclusivo enquanto que as restantes funcionam como um serviço misto.

O direito de preferência (*pre-emption*) baseia-se na interrupção de um processo que se encontra a ocorrer, como por exemplo o descarregamento ou carregamento de uma carga num determinado veículo, e surgir um novo com direito de preferência sobre o mesmo, numa determinada doca [3, 7]. Ou seja, é necessário parar o processo de modo a que o veículo com direito de preferência que chegou por último ao *crossdocking* possa realizar as suas atividades. Quando estas se derem por concluídas, o veículo sobre o qual se interromperam os processos anteriormente irá retomar e finalizar a sua atividade. Isto só é possível se no sistema este direito for permitido.

No que concerne à capacidade de armazenamento temporário, várias questões estão em causa. Normalmente, quando um veículo não está disponível para recolher produtos que estejam prontos para ser expedidos (ou seja, acabados de serem entregues por outro veículo), estes serão armazenados numa área própria durante um curto período de tempo, no chão do armazém. Esta área tem um limite de capacidade dependendo do espaço à mesma destinada, caso contrário a sua capacidade era considerada infinita. Algo comum nos sistemas *crossdocking* mais evoluídos, nomeadamente nos automatizados, é a existência de armazéns automáticos. Para além dos benefícios que oferecem, a ligação entre os dois sistemas funciona de forma eficaz. Exemplo disso são sistemas *crossdocking* com AS/RS incorporados. Desta forma os produtos que não podem ser expedidos de imediato após entrarem no sistema ficam armazenados corretamente, evitando eventuais incómodos de armazenamento e restrições de espaço. Um caso especial é o tipo de produtos a armazenar. No caso de produtos alimentares congelados, não é permitido que fiquem armazenados temporariamente no *crossdocking*, o que leva a que para estes determinados bens a capacidade associada seja nula.

A capacidade dos tapetes em sistemas automatizados ou o número máximo de trabalhadores disponíveis num sistema manual pode ser considerada limitada ou ilimitada.

As características de **fluxo de bens** ou operacionais [3] nem sempre são controladas pelo gestor do *crossdocking* pois podem ser impostas por intervenientes externos ao mesmo, como por exemplo os clientes, fornecedores e até empresas de transporte. A movimentação ou fluxo de bens/produtos é definida de acordo com parâmetros como o tempo de chegada, o tempo de partida e a intercambiabilidade do produto.

- tempo de chegada: os veículos podem chegar de forma concentrada ou desfasada. Caso o façam de forma concentrada e cada um deles possa ser descarregado a qualquer altura, não se detetam restrições e assume-se que todos se encontram disponíveis desde o início do horizonte de planeamento para serem processados. Se a chegada dos veículos estiver sujeita a restrições, como por exemplo tarefas a desempenhar noutros lugares, então o padrão de chegada é definido por veículo e respetivas tarefas subsequentes. Esta abordagem é aplicada tanto a veículos que cheguem para entrega ou recolha de produtos. O funcionamento do *crossdocking* é bastante influenciado pelo tipo de padrão de chegada a praticar [12].
- tempo de partida: tal como no tempo de chegada, pode ou não haver restrições associadas. Isto é, podem partir assim que o serviço esteja completo. Caso existam restrições (como algum compromisso agendado), o veículo, quer seja de entrega ou recolha, terá uma *deadline* a cumprir. Se não a cumprir existe a possibilidade de algum produto ou conjunto de produtos não ser expedido nesse carregamento e ter que aguardar pelo próximo para o mesmo destino.
- permutabilidade do produto: este parâmetro remete para a existência ou não de produtos permutáveis. Um produto é permutável quando pode substituir outro numa carga. Geralmente a carga manipulada numa doca não é permutável. Os produtos são pré-distribuídos, ou seja, dedicados a um destino ou veículo de recolha específico. Isto indica que a informação sobre o veículo ou destino do produto é conhecida antes dos mesmos entrarem no *crossdocking*. No entanto, em diversas situações os produtos são considerados permutáveis. A carga já não é definida por produto mas sim pelo seu tipo, o que permite que a expedição seja feita por tipo de produto e não por produtos específicos. Consequentemente, quando se verifica a existência de permutabilidade, realizam-se novas atividades de valor acrescentado como a rotulação dos produtos.

2.1.2 Tipos de crossdocking

Os diversos sistemas de *crossdocking* podem ser diferenciados através de diferentes abordagens. Utilizando as características referidas anteriormente, é possível diferenciar sistemas através da sua forma, número de docas, fluxo de bens, entre outros. Porém, é também possível fazer uma distinção através do número de toques ou fases existentes entre o produto e o *crossdocking* [3, 12].

Crossdocking a um toque, *crossdocking* a dois toques ou *crossdocking* a múltiplos toques, são operações básicas de um prestador de serviços de logística, e estão relacionados com o manuseamento de cargas. Estas operações permitem distinguir os sistemas existentes devido à relação direta com o seu funcionamento [3, 9].

- *crossdocking* a um toque: este tipo de sistema é considerado o tipo de *crossdocking* 'puro'. Como já referido, a metodologia ideal deste tipo de sistema passa por receber os produtos e movimentá-los diretamente para o veículo de expedição definido, sem qualquer tipo de armazenamento intermédio. Os produtos apenas são 'tocados' uma vez, no processo de movimentação.
- *crossdocking* a dois toques ou de uma fase (Figura 2.6): os produtos são recebidos e colocados num local específico, conhecido como 'área de *picking*'. Este processo é considerado o primeiro toque. De seguida são movimentados e carregados para o veículo de recolha, onde se efectua o segundo toque.
- *crossdocking* a múltiplos toques ou de duas fases (Figura 2.7): o primeiro toque ocorre na receção e colocação dos produtos numa área destinada e de seguida serão acondicionados (segundo toque). Assim que este processo seja concluído, serão movimentados e carregados para o veículo de recolha, onde ocorre o terceiro toque, e expedidos para o seu destino.

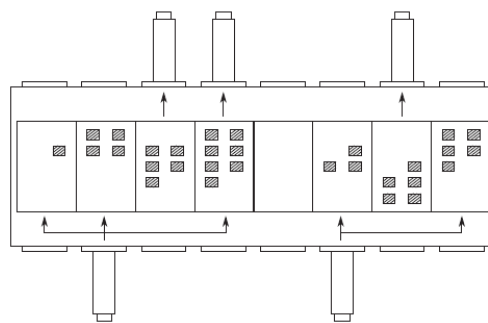


Figura 2.6: *Crossdocking* a dois toques [3]

A utilização destes três tipos de *crossdocking* varia bastante consoante a disponibilidade de espaço existente e com o objetivo do negócio. Os sistemas a

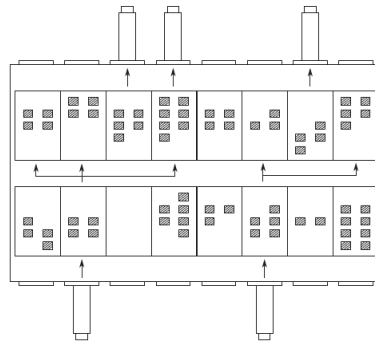


Figura 2.7: *Crossdocking* a múltiplos toques [3]

um toque, considerados *crossdocking* ‘puro’, são raramente utilizados devido ao facto da necessidade de armazenamento temporário da maioria dos produtos. Isto pode-se dever a diversos fatores tais como a calendarização de veículos, o que possibilita que alguns veículos de recolha não se consigam coordenar com veículos de entrega e vice-versa.

2.1.3 Tipos de armazenamento em sistemas *crossdocking*

Tal como mencionado, sistemas de *crossdocking* ‘puro’ são raros e difíceis de implementar devido à falta de sincronização entre os veículos de chegada e de partida. Apenas com um planeamento de calendarização perfeito, sem intervenientes externos provocadores de atrasos, seria possível o sistema funcionar de forma ideal. Tendo isto em conta, os produtos necessitam de ficar armazenados temporariamente no *crossdocking* de forma a serem expedidos assim que possível. Este armazenamento não é linear e varia de sistema para sistema. De forma a distinguir os tipos de armazenamento, estes são classificados de acordo com o tipo de movimentação das cargas [12].

- movimentação contínua: aplica-se ao *crossdocking* tradicional, onde a carga é enviada do fornecedor e recebida pelo cliente sem sofrer processos de armazenamento.
- movimentação híbrida: a carga recebida no *crossdocking* é dividida. Uma parte é expedida para os clientes e a outra é armazenada de forma a perfazer pedidos completos, criando inventário quando estritamente necessário.
- movimentação de distribuição: as cargas recebidas são divididas e cria-se um compasso de espera até haver volume para completar um veículo em modo FTL.

2.1.4 Calendarização de veículos

A carga destinada a um *crossdocking* necessita de ser recolhida e entregue em diversos locais após a sua consolidação no sistema. Tanto a recolha como a entrega de produtos pode ser vista como um processo de roteamento de veículos e, por vezes, é considerado parte integrante dos processos do *crossdocking*.

O principal objetivo, de entre diversas abordagens, passa por alcançar a rota dos veículos de recolha e entrega de produtos que permita minimizar a soma dos seus custos fixos e de transporte [13]. Idealmente os veículos teriam que chegar ao *crossdocking* simultaneamente para que não existam tempos de espera entre a entrega e recolha de produtos. De forma a contornar as restrições associadas a estes processos, surgiram algumas metodologias relativas à atribuição de docas a veículos cuja chegada ao sistema está iminente, criando-se assim problemas de transexpedição. Perante vários métodos e abordagens utilizados para contornar este problema, todos têm um objetivo em comum. Aumentar a produtividade e maximizar o desempenho de processos de entrega e recolha de produtos pelos veículos, de modo a minimizar o *makespan*, ou seja, o intervalo de tempo desde o início da entrega (descarga) de produtos do primeiro veículo até ao final da recolha (carga) de produtos pelo último veículo.

Caso o número de docas existentes no *crossdocking* seja superior ao número de veículos a entrar no processo, a atribuição dos mesmos simplifica-se atribuindo cada veículo a uma doca. Porém, habitualmente isto não se verifica, sendo o oposto o mais comum. De forma a que o fluxo de veículos seja organizado, utilizam-se métodos de calendarização, tais como [12]:

- calendarização a curto prazo: designam-se docas conforme o fluxo existente, sendo que o gestor ou plataforma logística contém informações prévias sobre o veículo de entrega. Pode ser realizada diariamente assumindo que as informações relativas ao dia sejam conhecidas.
- calendarização a médio prazo: a designação das docas é realizada tendo em conta destinos específicos. A título de exemplo, considera-se que todas as cargas provenientes do fornecedor X são entregues na doca 1 e as cargas destinadas ao local Y são expedidas na doca 10. É um método eficiente quando estão contratos temporais envolvidos no negócio.
- calendarização combinada (a curto e médio prazo): adequa-se a modos de serviço misto, tal como foi referido na subsecção 2.1.1, ao abordar as características táticas de sistemas de *crossdocking*. Ou seja, algumas docas consideram-se exclusivas a veículos provenientes do fornecedor X enquanto que as restantes são designadas a curto prazo.

- tempo real: o gestor ou plataforma do *crossdocking* designa as docas aos veículos à medida que estes vão chegando. É utilizado o método *First Come First Served* (FCFS), em que o primeiro a comparecer será o primeiro a ser processado, criando assim uma fila de espera sequencial.

2.1.5 Indicadores de desempenho

Tal como em todos os sistemas, é necessário definir indicadores que possam avaliar o desempenho e o sucesso das operações executadas. O desempenho de um sistema *crossdocking* pode ser avaliado de diferentes maneiras através de indicadores distintos, relacionados com operações específicas, fluxo de veículos e com os colaboradores [9].

Deste modo são utilizados indicadores como:

- nível de inventário: visto que o objetivo deste tipo de sistemas é minimizar a existência de inventário, a monitorização do número total de produtos armazenados num intervalo de tempo definido no horizonte de planeamento é o indicador mais lógico de avaliação.
- horas de trabalho: o número total de horas de trabalho é um indicador importante visto que, para um sistema *crossdocking* manual, representa custos elevados. Para um *crossdocking* automático, o número de horas de trabalho é relativo ao período de utilização dos tapetes de transporte.
- carga de trabalho: o número de horas de trabalho para sistemas manuais deve ser dividido de forma justa pelos colaboradores, embora seja um aspeto difícil de avaliar quantitativamente.
- distância de viagem do produto: a distância total percorrida pelos produtos dentro do sistema é um fator de extrema relevância a ser avaliado. Quanto maior a distância percorrida maior será o custo e o tempo de trabalho.
- tempo total de permanência do produto: um dos objetivos principais do sistema passa por aumentar a rotatividade dos produtos, logo o tempo total de permanência dos produtos no *crossdocking* é um fator importante a ter em conta.
- Número de toques: o número de toques, tal como referido anteriormente, define o tipo de sistema de que se trata. Cada toque representa um custo, e o custo está diretamente relacionado com o desempenho.
- produtos não expedidos: indicador que pode ser medido através do número de pedidos perdidos, ou seja, o número de produtos que não foi possível expedir, representando assim lucro perdido.

- tempo total de carga e descarga: de maneira a acelerar o fluxo de produtos é necessário libertar as docas o mais rápido possível para que o próximo veículo possa ser processado. Este indicador mede a taxa de utilização de cada doca para que seja possível, posteriormente, encontrar soluções para diminuir o tempo de entrega/recolha.
- desvio do tempo de processamento dos veículos: por norma, o tempo de chegada, partida e eventuais *deadlines* encontram-se definidos. De forma a garantir que tal horário seja respeitado, são medidos os atrasos e as antecipações dos veículos de entrega e recolha. Este indicador não é influenciado diretamente pela gestão do *crossdocking* mas sim por fatores externos que forçam os veículos a não cumprir o planeamento.
- congestionamento de veículos: com a minimização do tempo de viagem dos produtos, existe a possibilidade das tarefas de carga/descarga da mesma área se agruparem e causar um congestionamento de veículos. Isto origina um desaceleramento do processo geral que se encontra a decorrer, tendo assim o congestionamento um forte impacto no desempenho do sistema. É um indicador que pode ser contornado com medidas proativas embora não existam métodos diretos de correção.
- taxa de utilização das docas: indicador relacionado com o tempo total de carregamento e descarregamento dos veículos de entrega e de recolha.
- *makespan*: caso seja importante finalizar as operações o mais rápido possível, monitorizar o *makespan* é uma opção, sendo que este é o tempo entre o início da primeira operação e a finalização da última.

Todos os indicadores abordados podem ser utilizados para medir o desempenho de sistemas *crossdocking*, de acordo com as necessidades existentes, pontos e focos de melhoria e com o tipo de processos que se pretendem monitorizar.

2.2 Automated Storage and Retrieval Systems

Num sistema *crossdocking*, por vezes é necessário um local de armazenamento temporário de produtos. Pode ser apenas um espaço destinado a armazenamento (como, por exemplo, numa área delimitada no chão de uma doca) ou então um armazém incorporado no sistema, posicionado de forma estratégica. De maneira a automatizar e melhorar a eficiência de processos e operações, uma das melhores opções passa pela implementação de um armazém automático para soluções logísticas, conhecido como AS/RS. [14, 4].

AS/RS consideram-se uma combinação de equipamentos e controlos automáticos que permitem manusear, armazenar e recolher produtos com grande velocidade e precisão, sendo que estes são armazenados em posições específicas.

Consequentemente, são bastante utilizados em empresas industriais de modo a facilitar a gestão económica dos produtos e economizar espaço, tempo e equipamentos [5, 15]. Usa um tipo de armazenamento com um caminho fixo, que contém máquinas/veículos a circular nas suas linhas entre matrizes fixas de prateleiras de armazenamento [4]. De forma a ser possível interligar veículos com prateleiras e locais de recolha e recuperação de produtos, um AS/RS contém diversos elementos, entre os quais se destacam:

- prateleiras (*racks*): estruturas com locais próprios para armazenamento de produtos.
- guindastes/transelevadores/gruas (*cranes*): máquinas de armazenamento e recuperação de produtos totalmente automatizadas. Podem agarrar, largar ou mover os produtos de forma autónoma. Funcionam como elo de ligação entre os locais de armazenamento e os pontos de *input* e *output*.
- corredores: espaços entre as prateleiras, onde se movem os guindastes.
- pontos de *input* e *output*: locais onde os produtos que chegam são recolhidos para armazenamento (*input*) e onde os produtos recuperados de armazenamento são enviados para expedição (*output*);
- posições de seleção: se existirem, os colaboradores, caso necessário, removem um produto individual de uma carga recuperada antes que a mesma seja enviada de novo para o sistema.

A modelação de um sistema *crossdocking* deve ser pensada de acordo com os requisitos necessários ao bom funcionamento do negócio, tendo sempre em conta a sua eficiência. Assim, decisões estratégicas, táticas e operacionais devem ser inevitavelmente tomadas. Destes tipos de decisões surgem as características que mais se adequam ao sistema, o melhor tipo a utilizar, de que forma será feita a movimentação de cargas e se existirá armazenamento temporário. Ao longo do desenvolvimento da simulação deste tipo de sistemas, as várias decisões serão tidas em conta de acordo com os requisitos definidos.

2.2.1 Tipos de AS/RS

As várias opções existentes de AS/RS podem ser classificadas recorrendo a três parâmetros adaptáveis ao tipo de negócio e necessidades existentes. Para tal, os fatores abordados são alusivos ao tipo de operações realizadas pelos guindastes (Figura 2.8), ao manuseamento de cargas (como é feito e o tipo de cargas existentes) (Figura 2.9) e o tipo de prateleiras existentes, e processos com as mesmas relacionados (Figura 2.10).

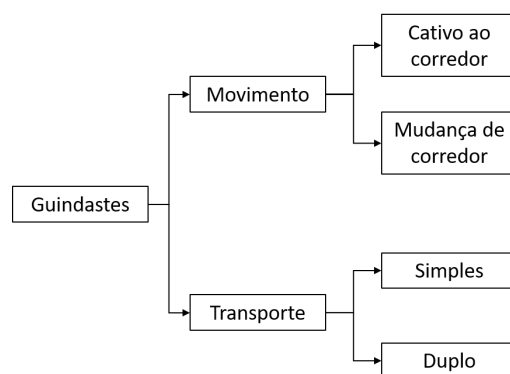


Figura 2.8: Classificação de AS/RS - guindastes - adaptado de [4]

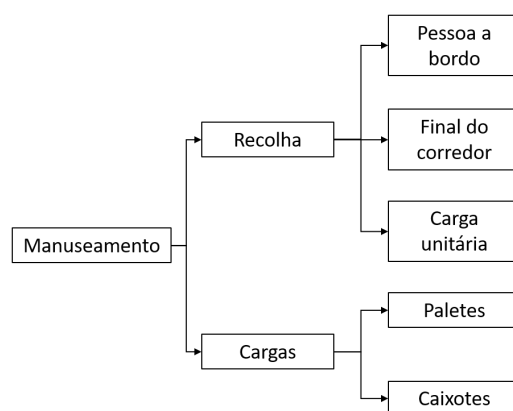


Figura 2.9: Classificação de AS/RS - manuseamento - adaptado de [4]

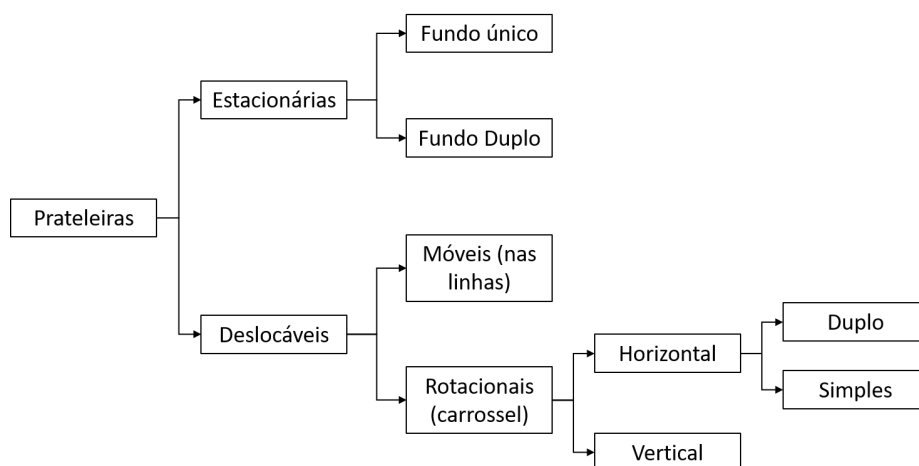


Figura 2.10: Classificação de AS/RS - prateleiras - adaptado de [4]

1. Guindastes (*cranes*)

- a) movimento: pode ser considerado cativo ao corredor quando apenas lhe é permitido circular num corredor específico. Por outro lado, é possível circular entre outros corredores, dependendo da estruturação das linhas do armazém, o que possibilita também um menor número total de guindastes existentes no sistema, algo que pode ser justificado caso o número de pedidos recebidos não justifique a existência de um guindaste por corredor [16].
- b) transporte: pode ser de comando simples ou duplo. Simples caso possa transportar apenas uma carga de cada vez, e duplo caso transporte duas cargas de cada vez. Existe também a possibilidade de se considerarem guindastes com transporte múltiplo (mais do que duas cargas de cada vez), mas é considerada uma parametrização rara.

2. Manuseamento (*handling*)

- a) recolha: pode ser efectuada de três modos distintos. Com uma pessoa a bordo do guindaste, o que em vez de armazenar/recolher uma carga, cria-se a possibilidade de armazenar/recolher apenas um item de uma determinada carga. O modo 'final do corredor' é mais comum, sendo que o guindaste ao recuperar uma carga posiciona-a numa estação de trabalho localizada estrategicamente no final do corredor para que um *picker* (seleccionador) retire a quantidade de produtos necessários da carga, sendo que, de seguida, o guindaste volta a armazenar a carga. Se a recolha for realizada por carga única, o guindaste armazena e recupera a carga como um todo, na quantidade total (algo que acontece regularmente com paletes).
- b) cargas: usualmente são diferenciadas em dois tipos. Paletes e caixotes (*bins*), sendo que as paletes, por norma, têm um tamanho mais considerável enquanto que os caixotes são conhecidos como *miniloads* (mini-cargas) [15].

3. Prateleiras (*racks*)

- a) estacionárias: não há deslocação de prateleiras, ficando constantemente imóveis no seu local [17]. Podem ser distinguidas pelo tipo de fundo que contêm. Caso seja de fundo único apenas fornece espaço para uma carga enquanto que tendo fundo duplo consideram-se duas posições de armazenamento, o que permite que duas cargas possam estar contempladas no mesmo local, por norma uma em frente à outra. Só é possível armazenar/recolher cargas da segunda posição caso a primeira não se encontre ocupada (tendo em conta que a segunda

posição é a mais difícil de aceder). Prateleiras de fundo duplo são consideradas vantajosas quando a variedade das cargas é relativamente baixa e a taxa de rotatividade das mesmas elevada.

- b) deslocáveis ou amovíveis: são prateleiras passíveis de deslocamento. Prateleiras classificadas como móveis deslocam-se consoante um caminho definido. Prateleiras rotacionais ou num sistema carrossel [17] podem ser classificadas como horizontais ou verticais, tendo em conta fatores físicos como o espaço disponível para a instalação de um AS/RS. São adequadas para armazenamento de produtos de tamanho reduzido e médio em diferentes níveis.

O tipo de AS/RS mais básico e de utilização comum contém características mais simples e mais fáceis de gerir. Normalmente em cada corredor está presente um guindaste cativo, ou seja, apenas atua nesse corredor específico e só pode transportar uma carga unitária de cada vez (transporte simples). O seu manuseamento é feito por carga única, como por exemplo paletes. Quanto às prateleiras, são estacionárias e de fundo único, sendo que a carga é acessível pelo guindaste. Neste tipo de AS/RS não há pessoas envolvidas. Toma o nome de *single unit-load aisle-captive* [4], ou seja, carga de unidade única de corredor cativo.

Sistemas mais complexos sofrem variações tendo o sistema AS/RS básico descrito como ponto de partida, como por exemplo a possibilidade de adicionar mais do que um guindaste por corredor, aumentar a capacidade das prateleiras para fundo duplo, permitir que um guindaste circule em diversos corredores, entre outras variações possíveis.

2.2.2 Projeto do AS/RS

De maneira a retirar o maior desempenho possível de um AS/RS, quer para a situação atual do negócio, quer para futuros desenvolvimentos, é necessário criar um desenho que o permita e que evite eventuais excessos de capacidade e diminua picos de fluxo de produtos e cargas [4]. Deve ter em consideração a existência de sistemas adjacentes, como é o caso do *crossdocking*.

Ao interligar um sistema *crossdocking* com um AS/RS, as cargas chegam ao ponto de recolha (*input*) e de recuperação (*output*) através de um tapete rolante automático. Ou seja, são depositadas ou recolhidas pelo AS/RS no local correspondente. A conexão entre sistemas pode representar atrasos e consequentemente diminuir a eficiência das operações. Assim sendo, no momento da tomada de decisão sobre o projeto a implementar, é necessário ter em conta as características dos sistemas adjacentes.

Assim, o projeto obtido através de dois fatores de extrema importância. O projeto físico e as políticas de controlo, tal como é perceptível na Figura 2.11. O

aspecto físico deve ser tido em conta, onde se define o tipo de AS/RS a utilizar e quais as características que irá apresentar, como é o caso do número de corredores e o tipo de prateleiras. As decisões relativas a estes parâmetros são tomadas conforme dados históricos e previsões, características da carga, orçamento disponível, entre outros.

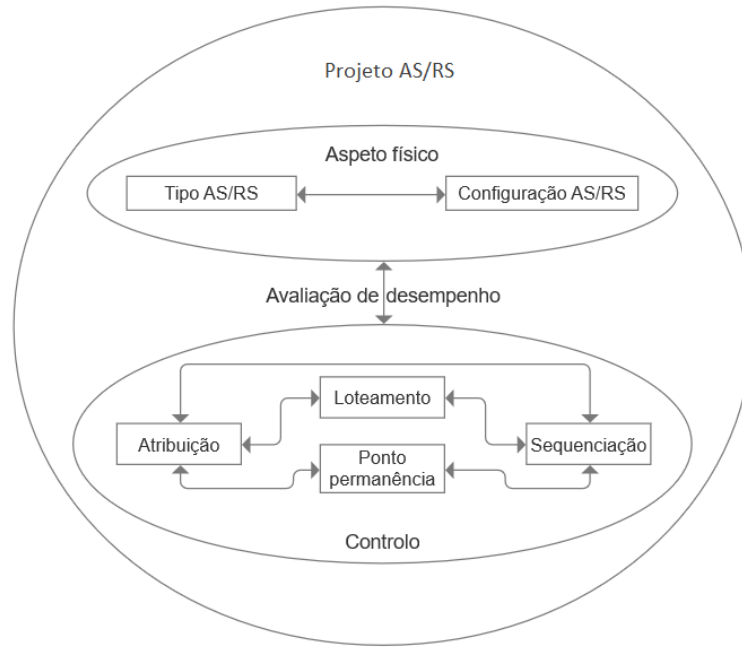


Figura 2.11: Projeto de AS/RS - adaptado de [4]

No fator associado ao controlo do AS/RS, é necessário implementar políticas de atribuição de armazenamento aos locais disponíveis, controlo de lotes ou loteamento, sequenciação de pedidos e também pontos de permanência de produtos (relacionado com o tempo de espera). A interligação entre estes dois fatores será um indicador de desempenho do sistema. Na análise dos parâmetros destinados ao controlo, é possível enumerar várias decisões de modo a moldar o sistema [4] verificando-se um grupo das mesmas na Tabela 2.1.

Tabela 2.1: Fatores de decisão - caracterização de AS/RS

Configuração	Número de corredores Altura das prateleiras Tamanho dos corredores Número de pontos de recolha/recuperação Capacidade de pontos de recolha/recuperação Número de guindastes por corredor
Atribuição	Método de armazenamento Número de classes de armazenamento Posição das classes de armazenamento
Loteamento	Tipo de loteamento (estático ou dinâmico) Tamanho do lote (por capacidade ou por tempo) Regras de seleção de ordens
Sequenciação	Restrições (por exemplo, prazo de validade) Tipo de operação (comando simples ou duplo) Método de sequenciação
Ponto de permanência	Ponto estático ou dinâmico Localização

2.2.3 Políticas de armazenamento

Um AS/RS pode ser definido de acordo com cinco políticas de armazenamento [18] utilizadas regularmente, de modo a atribuir cargas a locais de armazenamento em prateleiras, sendo elas:

- armazenamento dedicado: cada tipo de produto ou carga é armazenado num local específico, onde o reabastecimento desse produto ou carga é efectuado sempre no mesmo local. Embora tenha vantagens associadas, como o armazenamento de cargas cujo peso é superior ao normal nas extremidades inferiores das prateleiras, leva a que locais dedicados se encontrem reservados até para cargas sem *stock*.
- armazenamento aleatório: todas as localizações disponíveis têm a mesma probabilidade de atribuição de cargas.
- armazenamento de local aberto mais próximo: o primeiro local disponível é o selecionado para o armazenamento da carga. Por norma, os locais mais próximos do *input* e *output* do sistema encontram-se mais lotados.
- armazenamento com base na rotatividade de cargas: relaciona o local de armazenamento com a taxa de procura de uma determinada carga. Os produtos em que a taxa é superior encontram-se armazenados em locais mais próximos do *input* e *output*. Um pressuposto importante para este tipo de armazenamento é o conhecimento prévio da frequência de rotatividade das cargas. Devido ao tema de rotatividade de cargas estar implícito nesta política, é utilizada uma regra nomeada de *cube-per-order index* (COI) [19]. Esta regra é definida tendo em consideração a proporção existente entre o espaço de armazenamento e o número de solicitações da carga em questão.
- armazenamento baseado em classes: permite uma redução dos requisitos de espaço e reposições periódicas enquanto mantém os ganhos de eficiência, sendo considerado um tipo de armazenamento dinâmico. Será abordada com maior foco na sub-subsecção seguinte.

2.2.3.1 Armazenamento baseado em classes

O método de armazenamento baseado em classes relaciona-se diretamente com a taxa de procura de uma carga ou produto. É normalmente gerida através da regra de despacho COI. As prateleiras existentes são divididas em porções, criando assim as ditas classes. Uma classe é um bloco de locais de armazenamento, o que leva a que o tamanho das mesmas e a distribuição dos produtos seja projetado corretamente. Este tipo de armazenamento é referido como armazenamento ABC, devido ao facto de habitualmente existirem 3 classes (uma correspondente a cada

letra). A classe ‘A’ corresponde a produtos de movimentação rápida enquanto que a classe ‘C’ aos de movimentação mais lenta, sendo a classe ‘B’ um intermédio. Dentro de cada classe podem ser aplicadas políticas de armazenamento variadas, sendo a mais comum o armazenamento aleatório [5].

A principal vantagem encontrada neste método de armazenamento passa pelo aumento da eficiência, devido ao armazenamento de cargas de movimentação rápida mais próximo dos pontos de *input* e *output* do sistema, enquanto que ao mesmo tempo são aplicados os reduzidos requisitos de espaço para o armazenamento e a flexibilidade do armazenamento aleatório.

Ao formular e implementar uma política de armazenamento baseado em classes, é necessário enfrentar três decisões fulcrais [4].

- divisão de zonas: determinação do número de classes.
- modelação das zonas: em termos de tamanho existe a necessidade de determinar a capacidade de cada classe de forma a perceber o número de cargas/produtos que cada uma poderá albergar.
- posicionamento da classe: determinar onde irá ser colocada cada classe.

De acordo com um estudo [18], o posicionamento estratégico das classes está relacionado com o tipo de transporte efectuado pelo guindaste, de comando simples ou duplo. Uma configuração em forma de L, com limites quadrados para cada classe, é uma solução ótima ao trabalhar com transportes de comando simples e perto de ótima com comandos duplos. Por outro lado, também foi estudada a existência de classes em forma retangular com tipo de transporte de comando simples [20][21]. A Figura 2.12 representa ambas as formas indicadas.

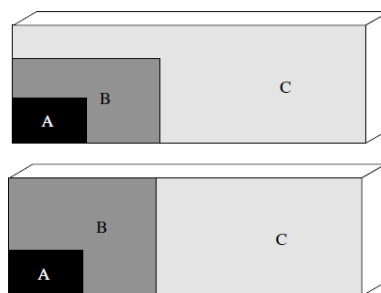


Figura 2.12: Posicionamento de 3 classes em forma de L (parte superior) e em forma retangular (parte inferior) [4]

2.2.4 Sistema de recolha de pedidos

Visto que se trata de um sistema de armazenamento, num AS/RS as operações são executadas tendo em conta os pedidos que recebe. De modo a facilitar esse processo, a implementação de um sistema de recolha de pedidos ou *Order Picking System* (OPS), como é regularmente nomeado, tornou-se de extrema relevância. Um OPS pode ser definido como a recuperação de cargas/produtos dos seus locais de armazenamento de forma a corresponder tanto à procura de clientes externos como internos. *Order Picking* (OP) ou pedido de recuperação deve ser abordado de forma sequencial, um de cada vez.

Um OPS é classificado de duas maneiras distintas de acordo com o seu modo de funcionamento [5, 14, 22].

- *Picker to Part* ou *Man to Goods*: quando o selecionador de pedidos viaja até ao local de recolha do pedido. Por norma ocorre em sistemas de armazenamento cujo processo é manual ou não totalmente automatizado.
- *Part to Picker* ou *Goods to Man*: as cargas provenientes dos pedidos recebidos viajam automaticamente para o selecionador. Sistemas automatizados utilizam frequentemente este tipo de funcionamento, sendo o AS/RS um sistema que representa esta classificação e vai ao encontro da tendência crescente que se verifica atualmente para a automação de operações de armazenamento e redução os custos de mão de obra associados ao processo.

A respeito das atividades de recolha de cargas/produtos, algo com influência direta no desempenho do AS/RS é o tipo de veículos em uso nos corredores [5, 14]. O tipo de cargas a transportar pelo veículo que realiza o procedimento (guindastes) podem ser classificadas como:

- *unit load*: as cargas/produtos são movidas e armazenadas por veículos capazes de mover e armazenar apenas uma unidade de carga (paletes), o que aponta para a realização de apenas uma paragem por ciclo.
- *less than unit load*: neste tipo de sistemas o número de paragens varia conforme os pedidos existentes, dentro da capacidade disponível. O veículo já é capaz de lidar com cargas de unidades múltiplas simultaneamente. Ou seja, é possível proceder à recuperação de várias cargas num único ciclo, visto que há capacidade de visitar vários locais no armazém antes de retornarem à posição dos pontos de *input* e *output*.

Em sistemas OPS de *unit load* [14], tanto para comandos de transporte simples ou duplos, é possível intercalar o tipo de operação que o veículo realiza, de acordo com modelos analíticos que envolvem parâmetros de vários níveis. Isto é, no

mesmo ciclo há a possibilidade de combinar a operação de armazenamento com a de recuperação de uma carga de forma oportunista.

Ao realizar o planeamento de OPS, surgem alguns problemas relacionados com o projeto do AS/RS e gestão da seleção, tanto para OPS do tipo *picker to Part* como para *Part to Picker* que se diferenciam em dois níveis, de políticas e de operações. São ambos agrupados em três principais estratégias operacionais [5]:

- atribuição de local de armazenamento e políticas de preenchimento;
- consolidação do pedido - transformação de pedidos de clientes numa OP;
- roteamento e sequenciação - identificação da rota do veículo no armazenamento ou recolha de carga.

Quanto à avaliação de desempenho do sistema em causa, existem diferentes critérios, sendo o principal o tempo de ciclo do *picking*, ou seja, o tempo de viagem. Este critério remete para a quantidade de tempo esperada em que o veículo realiza uma lista de operações de armazenamento e recuperação. De uma forma geral, a maior contribuição para o tempo de ciclo de um OPS é dada por três fatores.

- tempo administrativo de espera do veículo nos pontos de *input* e *output* no final de cada ciclo.
- tempo de processamento ou tempo que o veículo demora a extrair a carga do seu local de armazenamento e a documentar a operação realizada.
- tempo de viagem entre locais de armazenamento e recuperação.

Tendo conhecimento destes fatores, surge uma importância extrema de reduzir o tempo de viagem o que faz com que seja necessário medir a distância média percorrida por ciclo e transformá-la num indicador de desempenho significativo e generalizado. A duração temporal de um ciclo encontra-se diretamente relacionada com as configurações e parametrizações de um OPS.

2.2.5 Processamento em lote

O processamento em lote está associado à capacidade de transporte de cargas de um guindaste num AS/RS. O seu principal objetivo passa por combinar diversas OP para que sejam executadas num único ciclo [5]. Apresenta uma grande vantagem uma vez que o tamanho do ciclos para satisfazer um lote de pedidos é inferior à soma dos tamanhos de ciclo de execução de pedidos individuais, embora necessite de um maior esforço de perceção acerca de qual carga pertence a uma determinada OP. O tamanho do lote é limitado à capacidade do guindaste.

De forma a obter soluções para grandes problemas de acordo com tempos de ciclo aceitáveis, diversas heurísticas se baseiam e guiam por três passos principais [5].

- inicialização de um primeiro lote através da seleção de uma *seed* (semente);
- alocação de pedidos ao lote;
- definição de critério de paragem para determinar quando as OP do lote estão completas.

Uma regra importante a ter em consideração passa por garantir que uma OP individual não possa ser dividida entre vários lotes mas sim recuperada como um todo. Tais heurísticas resumem-se num esquema de forma a estruturar o processo (Figura 2.13).

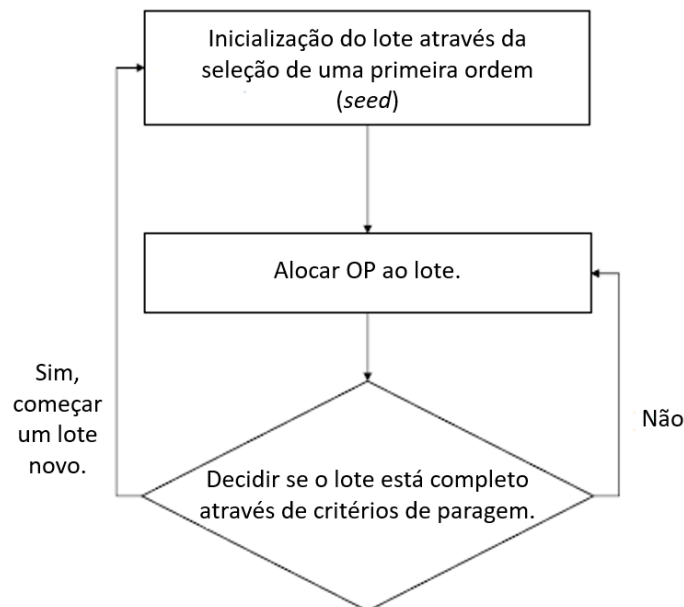


Figura 2.13: Estrutura comum de processamento em lote - adaptado de [4]

De forma a cumprir a estruturação apresentada, existem diversas regras para cada um dos passos. Regras de seleção de uma primeira ordem, regras de alocação de OP e critérios de paragem.

2.2.6 Sequenciação de pedidos de armazenamento e recuperação

De maneira a sequenciar as OP num AS/RS, é necessário definir princípios de seleção. Normalmente, em sistemas de armazenamento, os pedidos são satisfeitos

através do princípio *First-Come-First-Served* (FCFS). Isto é, num sistema de armazenamento, a primeira carga a chegar ao ponto *input* será a primeira a ser armazenada, e assim sucessivamente, criando-se uma sequenciação [4]. O mesmo acontece no *output*, onde o primeiro pedido de recuperação será o primeiro a ser executado. No que concerne à recuperação de pedidos, é necessário ter uma atenção extra aos prazos de vencimento das cargas, sendo necessário abordar a situação de forma mais profunda e, por vezes, utilizar princípios diferentes.

Por norma, os métodos de sequenciação encontram-se associados a AS/RS que trabalham com processos de manuseamento de recolha de carga unitário (*unit-load*) e são posteriormente adequados a outros tipos de AS/RS.

Em AS/RS com manuseamento de carga única, como já foi referido na Subsecção 2.2.1, são utilizados dois tipos de comandos de ciclo de transporte: comando simples (ou é realizado armazenamento ou recuperação de carga) e comando duplo (possibilidade de realizar um armazenamento e uma recuperação). Para ambos os tipos é possível aplicar diferentes métodos de sequenciação dinâmica mantendo os princípios de funcionamento, para além do FCFS [14].

- menor tempo de conclusão: o pedido a ser satisfeito em primeiro lugar é o que apresenta um menor tempo de conclusão.
- vizinho mais próximo [23]: pares de pedidos de armazenamento e recuperação são escolhidos de acordo com a distância entre os respetivos locais, de modo a minimizar o tempo de ciclo.
- menor caminho extra [23]: o local de armazenamento é escolhido tendo em conta a menor distância a percorrer para satisfazer o pedido de armazenamento (caminho extra) enquanto viaja para executar um pedido de recuperação.

2.2.7 Tempo de resposta de um AS/RS

No que concerne ao tempo de resposta de um AS/RS, alguns fatores de diversos níveis devem ser considerados de forma a analisar o desempenho do sistema. A interligação de processos em execução deve ser analisada, permitindo avaliar o desempenho, de modo a perceber onde se concentram eventuais problemas e oportunidades de melhoria. Esta interligação faz com que existam diversas correlações de processos dinâmicos a serem executadas simultaneamente, tais como a inicialização de movimentos de armazenamento e recolha, sequenciação de OP, processamentos por lote, entre outros [5].

De um modo geral, a análise deste tipo de sistemas de armazenamento tem como base a atividade de separação de cargas (armazenamento e recuperação), concentrando-se assim em parâmetros mais específicos. A Figura 2.14 é apenas

representativa, com um AS/RS em que a política de armazenamento correspondente é baseada em classes (A,B,C) e está construído num formato em L. Nos corredores, é possível identificar as prateleiras e os guindastes, bem como o tapete automático que transfere os produtos para os pontos de *input* e *output*.

Estão identificadas variáveis de modo a facilitar a percepção dos fatores a analisar.

- p : comprimento;
- p' : comprimento de área dedicado à classe A;
- q : altura;
- q' : altura de área dedicada à classe A.

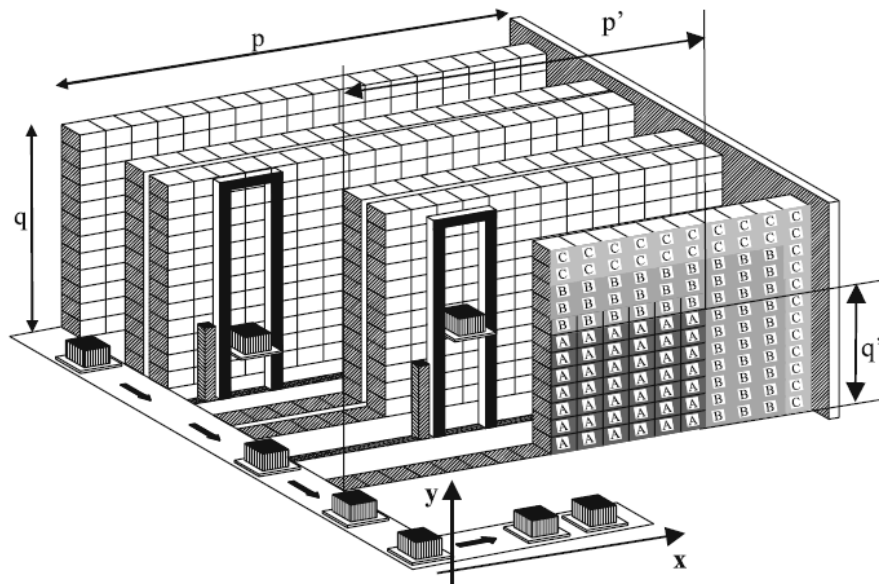


Figura 2.14: AS/RS com armazenamento baseado em classes. Classes A,B,C [5]

Deste modo, os principais fatores com influência no tempo de resposta de um AS/RS são:

- área (a): diretamente relacionado com a capacidade de armazenamento do sistema e é proporcional ao número de locais capazes de recolha de cargas.
- classes (A,B,C): fator que lida com a dimensão física de cada classe de armazenamento. A notação A,B,C e os valores correspondentes a cada letra indicam a porção da capacidade de armazenamento em percentagem

de cada classe. Por exemplo, caso $(A,B,C)=(0,10;0,25;0,65)$, então 10% da carga mais movimentada pertence à classe A (primeira classe de movimento e que se encontra localizada mais próximo do local de *input* e *output*). A mesma lógica é aplicada para as restantes classes.

- *dimension ordinate* ou ordenação de dimensão: corresponde ao número de linhas da lista de OP e está associado com a missão de recuperação de cargas pelo guindaste. Este inicializa e termina a sua viagem no ponto de *input* e de *output*. Sempre que o mesmo se encontra saturado, em termos de capacidade, dirige-se ao local de *input* e de *output*, liberta a carga com a qual viajava e inicia uma nova viagem.
- capacidade: relaciona-se com a capacidade do guindaste e com o número médio de cargas armazenadas e recuperadas em cada paragem durante o seu tempo de viagem (número médio de paragens por ciclo).
- forma $(p + q)$: rácio entre o comprimento e a altura das prateleiras.
- K : rácio entre o comprimento e a altura dedicada à classe A $(p',q'$, respetivamente), na sua forma retangular. Este rácio varia ao ser calculado para as restantes classes visto que têm uma forma em L.

2.3 Conclusão do Capítulo 2

Os sistemas *crossdocking* possuem diversas propriedades capazes de causar impacto caso sejam manipuladas, começando nas decisões estratégicas, táticas e operacionais que devem ser tomadas previamente e acabando no tipo de sistema a desenvolver. A variação das suas características, dos seus modos de funcionamento e do número de toques associados são os fatores mais relevantes no que diz respeito à manipulação deste tipo de sistemas.

O estudo de todos os fatores associados a sistemas AS/RS considera-se crucial para uma implementação lógica e eficiente no desenvolver da presente dissertação de mestrado, na medida que serão implementados conceitos, estratégias e políticas estudadas na simulação executada. O desenvolvimento de um sistema AS/RS deverá ser pensado de forma a melhorar todos os processos associados, tal como o seu projeto. Os fatores de decisão a ter em consideração equivalem a políticas e estratégias a implementar, tal como as políticas de armazenamento, o tipo de lotes a transportar para armazenamento e recuperação, e a sequenciação de restrições, de forma a ordenar os fatores mais importantes que um OP deve considerar. Todas as restrições devem ter em conta qual OPS foi implementada devido à diferenciação de processos entre OPS com *Picker to Part* ou *Part to Picker*.

Capítulo 3

Simulação para Processos Industriais

As cadeias logísticas encontram-se cada vez mais desenvolvidas devido a diversos fatores entre os quais se destaca o crescimento exponencial do comércio eletrônico. Assim sendo, este tipo de comércio impulsiona um desenvolvimento nas operações de armazenamento, distribuição e logística a grande velocidade. Todas as cargas passam por diversos processos de recolha, armazenamento, consolidação e expedição antes de serem entregues no seu destino final. Em torno deste conjunto de processos, a concorrência com as demais empresas presentes no mercado aumenta e foca-se maioritariamente na diminuição dos prazos de entrega, utilização de tecnologia de ponta automatizada e processos eficientes no que diz respeito ao manuseamento de cargas [24].

De forma a facilitar a perceção do funcionamento e eventuais implementações inovadoras para melhorar a eficiência de processos (tal como *layouts*, novos métodos de distribuição, entre outros), a simulação com componentes visuais é um fator de extrema relevância na antecipação de operações logísticas com vista a melhorar a eficiência. Em sistemas de movimentação rápida de cargas, como é o caso dos sistemas *crossdocking*, a simulação é crucial, bem como na implementação de armazéns automáticos como AS/RS, como é o caso do tema da presente dissertação.

3.1 Softwares de simulação existentes/adequados

De forma a antecipar o funcionamento de determinados processos industriais, é realizada a simulação dos mesmos através de *softwares* dedicados. Segundo Mandlik e Borkar [25], a simulação é uma ferramenta de trabalho poderosa

e muito importante em diversos processos operacionais, como por exemplo na produção, distribuição e processamento de bens ou cargas. Tem um elevado grau de importância devido às vantagens que apresenta, entre as quais a possibilidade de realização de análises de custo reduzido, com segurança e celeridade. Para além disso, é uma ferramenta utilizada de forma estratégica tanto na implementação de sistemas como no acompanhamento e melhoria dos seus desempenhos.

Após uma pesquisa intensiva foi encontrada uma panóplia de *softwares* vocacionados para a simulação de processos industriais e logísticos. De forma a perceber qual o mais adequado à realização da simulação de um sistema de *crossdocking* com um AS/RS incorporado, foi necessário ter em conta diversos parâmetros de naturezas diferentes. O fator da acessibilidade é bastante importante neste tipo de decisões dado que nem todos os *softwares* disponíveis oferecem os chamados ‘*free-trials*’ (licenças grátis). Por outro lado, outros fornecem licenças que permitem explorar os seus conteúdos e a sua posterior utilização. Para além disso, outras características são necessárias para este tipo de simulação. Ao longo da pesquisa foram detetadas características e parâmetros relevantes, o que levou a uma comparação entre alguns dos *softwares* identificados, sendo que outros não apresentavam informação suficiente para retirar conclusões. Posto isto, os *softwares* de simulação escolhidos para comparação foram os seguintes:

- AnyLogic [26];
- Flexsim [27];
- SIMIO [28];
- SIMUL8 [29];
- Visual Components (VC) [24].

Analisando cada *software* e a informação disponível nas respetivas plataformas, alguns parâmetros foram selecionados para proceder a uma comparação de forma a perceber qual se adequa melhor à execução da simulação pretendida. Com a ajuda de algumas plataformas digitais [30], foi possível comparar os *softwares* de acordo com os parâmetros existentes, algo que está representado na Tabela 3.1.

Prévio a esta comparação, foi construída uma primeira tabela que explica sucintamente a que corresponde cada parâmetro de comparação. Alguns destes não necessitam de explicação extra para além da sua denominação, porém, outros requerem informação adicional para se tornarem mais perceptíveis. Assim sendo:

- imagem 3D: este parâmetro indica se o *software* apresenta a simulação e os seus elementos através de imagem 3D, sendo que alguns *softwares* apenas utilizam visualização em 2D.

- API (*Application Programming Interface*): ou interface de programação de aplicações, funciona como uma plataforma de comunicação dentro do *software*. Permite que se desenvolvam rotinas ou funções de programação para a aplicação em questão. Ao efectuar determinados movimentos recorrendo a diversos elementos visuais, o *software* percebe a sua lógica e efectua o que o utilizador pretende simular, de acordo com as propriedades também estabelecidas pelo mesmo. Simplifica todos os processos relacionados com o utilizador, expondo apenas os objetos, elementos ou as ações que o mesmo necessita. Resumidamente, uma API é um conjunto de códigos de programação que permitem a transmissão de dados entre um produto de *software* e outro, facilitando a troca de informações e o manuseamento por parte do utilizador.
- biblioteca de conteúdos: local ou plataforma que oferece material de aprendizagem sobre a utilização generalizada do *software* bem como pormenores mais específicos. Tutoriais, *e-books*, resumos, casos de estudo e de aplicação, entre outros.
- manipulação direta: tipo de interação que envolve uma representação contínua do objeto/elemento em estudo, de forma rápida, reversível e com possibilidade de acrescentar ações de incrementação. O principal objetivo deste parâmetro é permitir a manipulação dos objetos/elementos.
- visualização de dados: representação gráfica de informações e dados através de elementos visuais como tabelas, gráficos, entre outras ferramentas de visualização. Isto permite compreender melhor os dados, valores obtidos, discrepâncias e as suas tendências ao longo da simulação.
- análise do desenho do projeto: processo que se enquadra na tomada de decisão no qual ferramentas analíticas que englobam fundamentos matemáticos, estatísticos e de engenharia são utilizadas para analisar o modelo do processo a simular e perceber se pode ser aplicado em contexto real.
- modelação dinâmica: possibilita a alteração de diversas propriedades ao longo da simulação, tal como a variação do tempo ou a alteração do comportamento de um objeto/elemento/operador se assim se proporcionar.
- controlo de processos: possibilidade de monitorizar e ajustar um processo de forma a obter um *output* desejado ou próximo do desejado.
- dados em tempo real: obtenção e visualização de dados relativos à simulação que decorre e aos seus processos em tempo real. Permite perceber eventuais problemas num processo intermédio e até mesmo ajudar a identificar mais rapidamente o *bottleneck* do modelo.

- análise de risco: possibilita uma análise aos resultados possíveis e uma avaliação do impacto dos mesmos, de forma a calcular o risco, permitindo uma tomada de decisão mais apurada.
- importar/exportar dados: importar e exportar dados de outras plataformas e *softwares*.
- base de dados de indústria: existência de uma base de dados de elementos pré-feitos vocacionados para a simulação de processos industriais. É um tipo de base de dados que facilita a integração de processos, permitindo ao utilizador ter noção de que elemento se trata e se corresponde ao que o mesmo pretende utilizar na simulação. O facto de conceitos relacionados com o tema da indústria estarem também definidos no *software* através deste tipo de base de dados também se torna uma grande vantagem.

Tabela 3.1: Comparação de *softwares* de simulação

Parâmetros	AnyLogic	FlexSim	SIMIO	SIMUL8	VC
Imagem 3D	X	X	X		X
API	X	X	X	X	X
Biblioteca de conteúdos	X	X	X		
Manipulação direta	X	X	X	X	
Visualização de dados	X	X		X	X
Análise de projeto	X	X	X		X
Modelação dinâmica	X	X	X	X	X
Controlo de processos		X		X	
Dados em tempo real		X		X	
Análise de risco		X	X		
Importar/exportar dados	X	X		X	X
Base de dados de indústria		X	X	X	

Analisando a Tabela 3.1, é possível verificar que o *software* FlexSim, de acordo com os parâmetros selecionados, mostra ser o mais completo para simular um sistema de *crossdocking* com um AS/RS incorporado, tendo sido este o escolhido para a realização da presente dissertação. Um fator bastante importante na escolha do *software* passa pela existência de tutoriais, *webinars*, entre outros materiais didáticos que se revelam bastante úteis no desenvolvimento do modelo. Para além disso, o FlexSim permite uma inicialização grátis e, segundo diversos *feedbacks* existentes em inúmeras plataformas, o seu apoio ao cliente é bastante recomendado.

Juntando a todos estes fatores, o facto do ISEP providenciar uma licença que permite uma simulação mais cómoda e completa, eliminando restrições de utilização possivelmente existentes, também foi um facto de pesar na decisão da escolha do *software* recair no FlexSim.

3.2 Introdução ao FlexSim

A primeira versão do FlexSim foi lançada no mês de Fevereiro do ano 2003. A partir desta data, o seu desenvolvimento tem sido constante, tornando-se um *software* cada vez mais evoluído e completo. A possibilidade de simulação de diversos tipos de sistemas é uma realidade, sendo que é utilizado maioritariamente para simulação de processos de fabrico, de logística, distribuição e de saúde. É um *software* cada vez mais utilizado para automatizar o desenvolvimento de modelos de simulação em três dimensões (3D), relacionados com todos estes temas. Para criar simulações conformes e o mais aproximadas da implementação que se espera em contexto real, as suas principais características são:

- objetos/elementos padrão: através de uma biblioteca que possui elementos padrão, em que cada um possui uma lógica própria e desenvolvida pelo próprio FlexSim, é possível imitar os elementos utilizados nos processos reais.
- ferramentas para desenvolvimento de lógica: este tipo de ferramentas são extremamente importantes, pois permite construir uma lógica na associação de elementos do modelo, criando um sistema com inteligência própria, capaz de tomar decisões. A lógica é construída através de blocos, tal como um fluxograma, e pelas associações entre eles. Com isto, o *software* oferece a possibilidade de visualizar a simulação em 3D ao mesmo tempo que é possível acompanhar o estado dos produtos, ou qualquer outro tipo de item utilizado, em cada bloco de atividade que desempenha.
- controlos de fácil utilização: para proporcionar ao utilizador uma simulação facilitada e mais interativa, os controlos e os elementos são movidos e inseridos no sistema através de controlos de agarrar e soltar.
- análise de desempenho: devido à existência de ferramentas especializadas na análise dos fluxos e processos implementados durante a simulação, os resultados podem ser gerados através de gráficos e tabelas que analisam todos os dados requeridos em tempo real de simulação. Desta forma, a análise a determinados fatores é mais completa, quer em termos visuais, quer em termos de dados.

De forma introdutória à utilização do *software*, a criação de modelos é feita com relativa facilidade. Ao iniciar o FlexSim, uma *interface* é apresentada ao utilizador, onde o mesmo tem a possibilidade de escolher se pretende criar um modelo novo, abrir um modelo já existente, definir as suas preferências, licenças, entre outros módulos, como mostra a Figura 3.1. O utilizador, dependendo do módulo que escolher, é reencaminhado para a etapa seguinte. Caso escolha a

criação de um novo modelo, na Figura 3.1 “*New Model*”, será encaminhado para um espaço onde poderá inicializar as suas experiências e simulações.



Figura 3.1: Página inicial na criação de um modelo no FlexSim

Para a construção do modelo 3D, o FlexSim oferece uma biblioteca de elementos (“*Library*”) dividida em várias categorias, de acordo com o que o utilizador necessita para o seu projeto. Ao abrir a janela referente à ferramenta que define a lógica do modelo, nomeada *Process Flow*, outra biblioteca, também dividida por categorias, será apresentada com outro tipo de elementos. Neste caso, os elementos passam a ser considerados atividades. A Figura 3.2 representa um exemplo de interação entre o que é implementado no modelo e no *Process Flow*, sendo que este controla os acontecimentos do modelo. O exemplo gerado tem como objetivo a criação de itens através de uma fonte, que são enviados para diversas filas, que por sua vez são transportados para outras filas através de robôs. As atividades representadas do lado direito da imagem controlam os processos associados ao funcionamento do modelo. Conceitos e definições associadas a estes elementos e lógicas serão abordados no Capítulo 4.

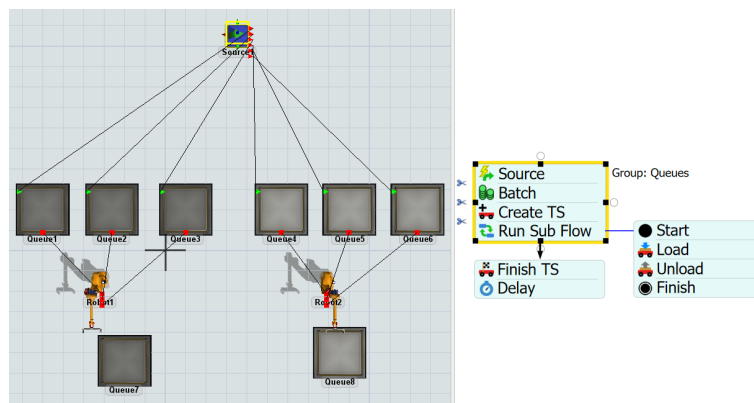


Figura 3.2: Exemplo de interligação entre o modelo 3D e o *Process Flow*

O conjunto de elementos e propriedades do *software*, as ligações entre os mesmos e as dinâmicas de funcionamento são adquiridas ao longo do desenvolvimento de vários modelos teste, representativos de diversos processos. O FlexSim oferece tutoriais referentes a diversos temas, os quais ajudam a desenvolver várias competências importantes. A Figura 3.3 representa um exemplo de modelo de simulação simples de transporte de cargas/produtos através de tapetes rolantes, o qual foi realizado numa fase embrionária da presente dissertação, através da realização de tutoriais de formação disponibilizados pelo FlexSim.

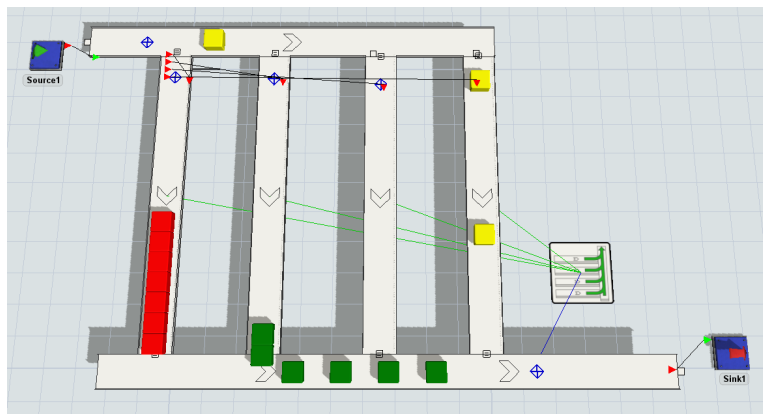


Figura 3.3: *Software* de simulação 3D orientado a objetos - FlexSim - Exemplo prático 1

A Figura 3.4 representa um modelo simples de simulação de um armazém automático, outro dos principais temas da presente dissertação. Através deste modelo é possível prever um crescimento do tamanho e das funcionalidades que o armazém poderá conter graças aos elementos e às funcionalidades que o *software* proporciona.

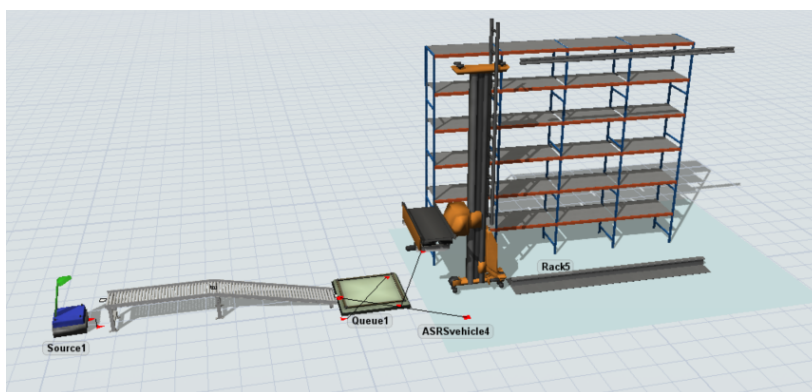


Figura 3.4: *Software* de simulação 3D orientado a objetos - FlexSim - Exemplo prático 2

3.3 Conclusão do Capítulo 3

Através desta análise foi possível perceber quais os *softwares* existentes no mercado para realizar este tipo de simulação e quais os que disponibilizam os parâmetros e as licenças necessárias para tal. A comparação realizada entre eles permitiu adquirir uma maior visibilidade sobre as valências que cada um apresenta e facilitar o momento de decisão na escolha do *software* a utilizar, bem como as suas vantagens.

O FlexSim revelou-se o mais completo e acessível. No capítulo seguinte procede-se à abordagem da arquitetura proposta para a simulação do modelo de *crossdocking* com um AS/RS incorporado.

Capítulo 4

Arquitetura do Modelo de Simulação

O presente capítulo tem como objetivo apresentar e explicar a arquitetura utilizada, quer para a simulação do sistema *crossdocking*, quer para a do armazém automático (AS/RS) a implementar. Assim sendo está dividido em duas secções, uma referente a cada área de simulação.

A arquitetura do modelo de simulação a executar deve ser adequada e pensada de forma a cumprir os objetivos que rodeiam o tema da presente dissertação. Desta forma, foi necessário definir um *layout* que englobe diversos elementos e que seja capaz de funcionar de forma eficiente. Para além disso, a possibilidade de ser implementado em contexto real deve sempre ser considerada.

Serão apresentadas duas arquiteturas distintas, uma correspondente a cada sistema a simular. Evidentemente que a arquitetura de um sistema *crossdocking* irá ser totalmente diferente da de um AS/RS, visto que são sistemas de naturezas e objetivos completamente diferenciados. Neste capítulo não serão especificados quais os elementos a utilizar no *software*, apenas será apresentada a lógica da simulação, o *layout* a utilizar e outras considerações relevantes.

Quanto ao funcionamento base do modelo, como é perceptível ao longo da fundamentação teórica (Capítulo 2 da presente dissertação), o objetivo do *crossdocking* passa pela movimentação rápida de itens de um local para outro, dentro do centro logístico. Assim, estes chegam num veículo de transporte de entrada (*inbound trucks*), são inseridos no sistema e distribuídos para outros veículos (*outbound trucks*) que os levarão para o seu destino final. Esta distribuição é feita através de ordens de despacho previamente definidas. Criando um paralelismo com o existente em contexto real e ao mesmo tempo uma ponte de ligação

com o descrito ao longo do Capítulo 2, os veículos que chegam ao sistema podem transportar apenas itens de um tipo e/ou de um fabricante específico. Estes, ao entrarem no sistema *crossdocking*, serão combinados e agrupados com itens provenientes de outros veículos de entrada, de tipos e/ou fornecedores diferentes. Posto isto, serão embalados e enviados para veículos de saída, situados na zona de expedição do sistema, com o objetivo de transportá-los para o seu destino final. Entre a chegada e a saída dos itens do sistema *crossdocking*, os mesmos podem necessitar de ser armazenados temporariamente. Algo muito comum nos centros logísticos atualmente é a existência de armazéns automáticos. Desta forma, a implementação de um sistema AS/RS fornece a possibilidade de armazenamento de itens, tornando o modelo mais completo. Os itens não utilizados para cumprir alguma ordem serão armazenados e a recuperação de itens ocorre quando uma ordem de despacho necessita de algum, ou alguns, dos itens diferentes, para ser concluída e os mesmos não se encontrarem no sistema. Esta é considerada a lógica principal a implementar na presente simulação, aproximando-se o mais possível do contexto real.

4.1 Arquitetura do sistema de crossdocking

Quanto à arquitetura do sistema *crossdocking*, foi definido um *layout* considerado eficiente e de possível implementação em contexto real.

Tal como explicado no Capítulo 2, um sistema de *crossdocking* baseia-se na movimentação rápida de cargas ou produtos. Estas chegam ao edifício em causa através de veículos de mercadorias. Sendo assim, é necessário que as cargas sejam recolhidas dos veículos de entrega, processadas pelo *crossdocking* e enviadas para os veículos de expedição correspondentes, que as entregarão ao seu destino. Assim, numa primeira fase, as cargas dão entrada no sistema no momento que são descarregadas dos veículos de entrega para tapetes rolantes transportadores. Posto isto, numa segunda fase, é necessária a criação de uma estação de controlo de forma a perceber se uma determinada carga já deu entrada (estado de receção), o que também permite identificá-la e perceber o percurso que tem pela frente no *crossdocking*. Poderá ter um destino traçado previamente ou encaminhar-se para o armazém automático de forma a ficar armazenada durante um curto espaço de tempo. A segunda fase é de extrema importância visto que é nela que as cargas são identificadas e iniciam o seu processo de distribuição de forma a chegarem convenientemente ao seu destino, sem eventuais erros de leitura.

Um dos requisitos que o *layout* deve apresentar passa pela interligação com um armazém automático. Ou seja, terá que haver uma ligação a esse sistema adjacente que seja capaz de transportar determinadas cargas ou produtos para armazenamento e outra ligação capaz de os voltar a introduzir no *crossdocking*. Este requisito representa a terceira fase.

Na fase seguinte, as cargas devem ser transferidas para um local apropriado, idealmente uma unidade acumuladora dinâmica que contenha um posicionamento otimizado. Ou seja, as cargas devem ser colocadas em locais adequados onde se irão acumular de forma programada. Diversas possibilidades existem nesta fase, sendo que agrupar as cargas destinadas a um determinado destino ou a uma rota pré-definida sejam as mais comuns. Num sistema *crossdocking* existem múltiplas unidades acumuladoras de forma a permitir uma organização melhorada e mais eficiente dos processos de separação de cargas e atribuição das mesmas aos veículos de expedição correspondentes.

Transferidas as cargas para as diversas unidades acumuladoras, a próxima fase passa por voltar a transferi-las para um local de paletização. Ou seja, serão enviadas para um local onde serão agrupadas em paletes de acordo com a metodologia de expedição utilizada. Durante este processo é possível considerar que existe uma envoltória e proteção das paletes com algum tipo de material adequado. A transferência pode ser efectuada de diversas formas. Caso se opte por uma simulação/implementação não totalmente robotizada, pode ser realizada através de operadores. Ao invés disso, de forma a automatizar ao máximo o sistema, pode ser realizada por *Automated Guided Vehicles* (AGV), por Unidades Móveis de Paletização (UMP) ou até mesmo por simples tapetes rolantes transportadores. O *layout* utilizado está representado na Figura 4.1.

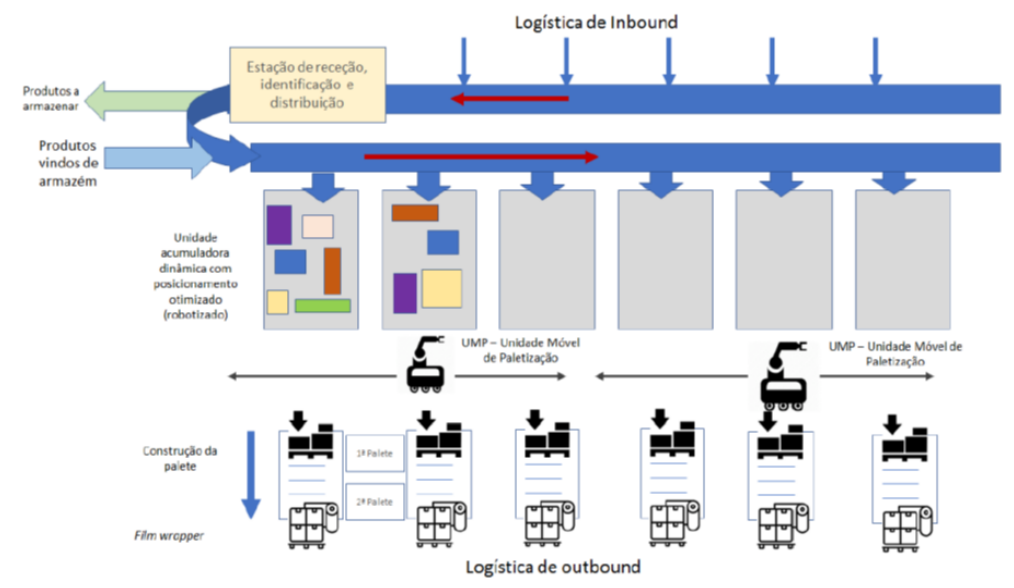


Figura 4.1: Arquitetura do sistema de *crossdocking* a simular

Estando as cargas agrupadas e colocadas em paletes, estas serão carregadas para os veículos de expedição correspondentes de forma a chegarem ao seu destino da forma mais otimizada possível. Todas as fases indicadas estão representadas

na Figura 4.1, onde é possível perceber o fluxo das cargas que entram no sistema, os passos seguintes até serem expedidas para os seus destinos e todo o *layout*. De salientar que o *layout* apresentado na Figura 4.1 foi facultado por uma empresa sediada em Portugal, que se encontra a desenvolver este tipo de sistemas.

Relativamente aos principais elementos do *software* FlexSim [27] para cumprir os parâmetros do que foi apresentado na Figura 4.1, numa fase inicial consideraram-se:

- *source* ou origem: utilizado para criar os itens de fluxo que irão percorrer o modelo ao longo da simulação. É possível definir o tipo de itens que serão criados, bem como a sua forma, cor, tamanho, entre outras propriedades. Em contexto prático, uma *source* funciona como um *inbound truck*, ou seja, um veículo de entrada. Também será utilizada uma *source* com o objetivo de criar paletes para agrupar diversos itens provenientes do *crossdocking* com destino a um veículo de saída.
- *conveyors* ou tapetes de transporte: elementos utilizados para o deslocamento de produtos entre outro tipo de elementos ou entre dois locais.
- *queue* ou fila: elemento utilizado para armazenar itens de fluxo quando um dos elementos no processo seguinte ainda não é capaz de os aceitar. Na presente arquitetura funciona como uma unidade acumuladora.
- *combiner* ou combinador: é um elemento utilizado para agrupar vários itens de fluxo. Pode unir estes itens permanentemente ou pode empacotá-los para que possam ser separados posteriormente. Este elemento tem como objetivo agrupar os itens de fluxo que constituem uma ordem de despacho.
- *decision points* ou pontos de decisão: pontos colocados estrategicamente ao longo do fluxo com o objetivo de decidir o próximo passo do item que o atravessa, nos tapetes de transporte. Assim que um item atravessa um destes pontos, um ou mais eventos pré-definidos irão ser ativados de forma a encaminhar um item para um trajeto requerido ou realizar algum tipo de operação no fluxo no momento desta ocorrência.
- *photo eyes*: funcionam como um sensor de deteção de itens. Podem ter diversas utilidades, nomeadamente para separar itens, para criar um espaço entre eles de forma a não seguirem para a etapa seguinte demasiado juntos, prejudicando o fluxo e possibilitando um eventual choque entre os mesmos.

Comparativamente com tudo o que foi apresentado ao longo do Capítulo 2, mais concretamente na Secção 2.1.1, o sistema de *crossdocking* a desenvolver é considerado um sistema a múltiplos toques. Como justificação, não só pelo facto

de existir a possibilidade de armazenamento de itens temporário, os itens são enviados para o fluxo e, após serem distribuídos por diversas UA correspondentes a uma ordem de despacho, são ainda combinados num paletizador e acondicionados num *wrapper* antes de seguirem para expedição. Quanto à sua forma, considera-se que este é um sistema em I com movimentação interna de produtos automática e modo de serviço exclusivo. Isto porque utiliza docas apenas para entrada de itens (*inbound*) e docas apenas para saída dos mesmos (*outbound*). Em formato de síntese, o *crossdocking* implementado contém as seguintes características:

- o número de docas é reduzido de modo a facilitar a percepção do sistema. Utilizadas cinco docas de entrada e três de saída;
- forma em I;
- movimentação interna de produtos automatizada;
- modo de serviço exclusivo;
- sistema a múltiplos toques;
- tipo de armazenamento híbrido (carga é dividida, havendo itens que viajam diretamente para a expedição e itens que seguem para armazenamento temporário).

Concluindo, considera-se que a arquitetura do *layout* apresentada é plausível e que a mesma se adequa à simulação, indo ao encontro dos objetivos da presente dissertação. Para além das propriedades do sistema apresentadas, várias outras possibilidades poderiam ser implementadas. A próxima secção, que demonstra a arquitetura do armazém automático (AS/RS) a simular, considera-se uma extensão ao sistema *crossdocking*.

4.2 Arquitetura do armazém automático (AS/RS)

De maneira a retirar o máximo de eficiência da simulação de um AS/RS integrado num sistema de *crossdocking*, o *layout* do mesmo foi definido de forma progressiva. Ou seja, foram definidas inicialmente algumas propriedades que, com o avançar da simulação e das necessidades de armazenamento, fossem expostas a uma mudança, quer para um aumento da sua capacidade, quer para a redução da mesma. Como foi referido no Capítulo 2, o tempo de resposta de um AS/RS varia maioritariamente conforme o comprimento e altura das prateleiras. Estes fatores têm influência em vários sistemas o que indica que a eficiência do mesmo poderá estar diretamente relacionada com o seu tamanho.

Visto que este sistema não é independente, ou seja, encontra-se conectado com o *crossdocking*, a sua origem será uma das saídas do mesmo, e a sua saída será

uma das entradas no fluxo de itens, diretamente para um tapete de transporte, como é visível na Figura 4.2, onde se apresenta um esquema representativo da interligação entre os dois sistemas.

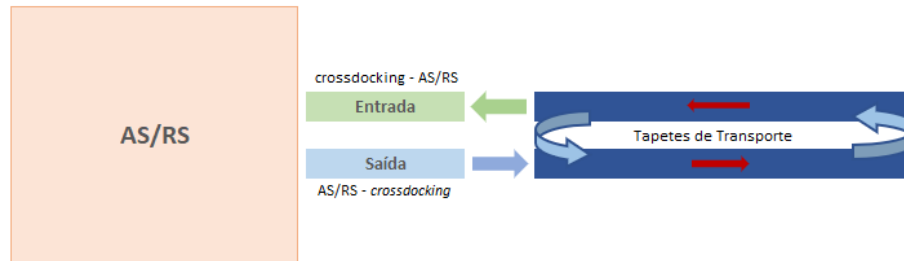


Figura 4.2: Representação da interligação entre o *crossdocking* e o AS/RS

De maneira a que o AS/RS fosse capaz de acompanhar o *crossdocking* em termos de capacidade de fluxo de itens, ou seja, de armazenar os itens provenientes do *crossdocking* sem que haja falta de espaço para armazenamento, foi definido que o armazém deveria conter três corredores, o que indica a utilização de quatro prateleiras (*racks*). Uma delas irá constar no modelo para caso haja alguma incongruência ou um excesso de itens armazenados não esperado, servindo assim de *rack* de segurança. Nesses corredores, seriam utilizados veículos especializados para este tipo de sistemas, nomeados de *ASRS Vehicles*. Para que os itens entrem e saiam do sistema de armazenamento, várias formas de execução destes dois processos foram pensadas. É possível criar estações de *input* e *output* simultâneas, o que indica que apenas uma estação pode realizar ambas as tarefas (recolha e recuperação de itens) e enviar os mesmos para o *crossdocking*. Outra hipótese é criar estações de *input* e *output* independentes, sendo que a primeira é responsável pela recolha de produtos para serem armazenados e a segunda pela recuperação de itens e reenvio dos mesmos para o sistema *crossdocking*.

No que respeita aos elementos a utilizar na arquitetura do AS/RS:

- *conveyors* ou tapetes transportadores: realizam o transporte dos itens desde o momento em que saem do *loop* do *crossdocking* até darem entrada numa *queue* específica de recolha de itens para armazenamento.
- *queues*: tal como para o sistema *crossdocking*, a *queue* funciona como uma unidade de espera de itens. Enquanto não podem ser recolhidos para armazenamento, é neste elemento que se encontram. O mesmo acontece para a recuperação dos mesmos. No momento em que itens que se encontrem armazenados sejam requeridos por uma ordem de despacho, serão recuperados do armazém, colocados numa *queue* e de seguida enviados para o *crossdocking*. Fazendo uma ligação com o que foi explicado no Capítulo 2, as *queues* servem de estações de *input/output* do sistema de armazenamento.

- *racks* ou prateleiras: é um objeto de armazenamento. A principal especialização deste elemento é a definição das propriedades das *slots* de acordo com as necessidades de armazenamento e o espaço disponível para a sua implementação.
- *ASRSVehicle*: é um tipo essencial de transporte projetado especificamente para trabalhar com *racks*, sendo um executor de tarefas. Desloca-se para a frente e para trás, ao longo do corredor formado pelas *racks*, armazenando e recolhendo itens de acordo com as ordens recebidas. Para conseguir alcançar todas as *slots*, possui movimentos de alcance, elevação e deslocamento que podem ocorrer simultaneamente, aumentando a eficiência do processo.

De forma a ser possível visualizar o *layout* do sistema pretendido, foi efectuada uma simulação no *software* FlexSim de um AS/RS com três corredores e seus respectivos *ASRSVehicles* e com estações designadas apenas para *input* e *output*, tal como é possível verificar na Figura 4.3 e Figura 4.4. Estas figuras servem de enquadramento do que se pretende através de duas vistas distintas, uma do topo e outra de perspectiva.

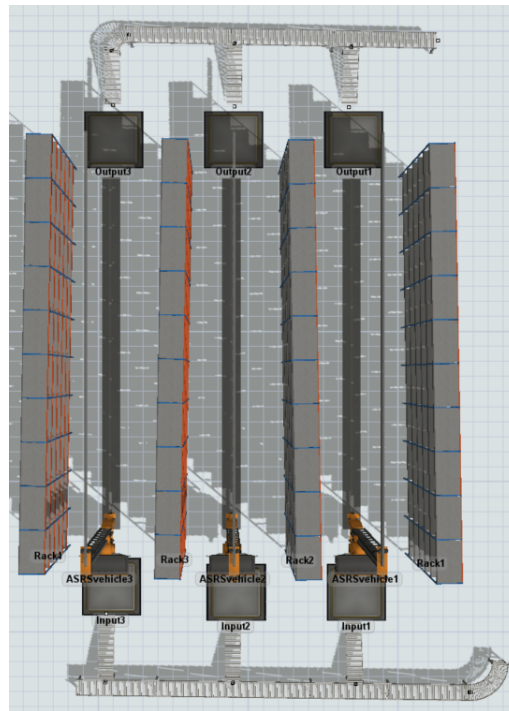


Figura 4.3: Representação do AS/RS - vista do topo 3D

Este sistema será colocado numa posição lateral ao *crossdocking* de forma a criar uma interligação realista entre ambos, tendo em conta as restrições de espaço eventualmente associadas. Na Figura 4.3, as *queues* situadas na parte inferior do

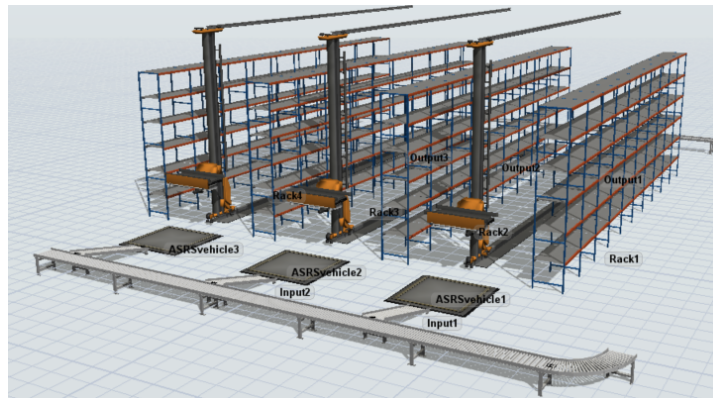


Figura 4.4: Representação do AS/RS - vista de perspectiva 3D

sistema representam os *inputs* e as colocadas na parte superior representam os *outputs*. É possível perceber que são utilizados três *inputs* e três *outputs*, um destinado a cada corredor.

Ao implementar um armazém, um dos principais fatores a ter em conta é o seu espaço útil. Ou seja, qual o espaço destinado exclusivamente ao armazenamento de produtos e se esse espaço é capaz de satisfazer as necessidades existentes. Diretamente relacionado com isto, o tamanho das *racks* a utilizar deve ser definido de modo a conseguir armazenar os produtos que entram no sistema. O *software* oferece um menu próprio para que seja possível definir o número de baías disponíveis por cada nível de armazenamento, as dimensões de cada nível e de cada *slot* e oferece também a possibilidade de definir *slots* que não necessitam de ser utilizadas. A Figura 4.5 representa o menu de uma *rack* com as medidas que serão utilizadas no modelo a implementar.

 A screenshot of a software interface for configuring a rack. At the top is a grid with 7 rows (labeled 1 to 7 on the left) and 10 columns (labeled 1 to 10 at the bottom). Below the grid is a control panel with the following elements:

- Selected:** All Bays All Levels All Slots
- Number of Bays:** 10 (input field)
- Number of Levels:** 7 (input field)
- Slots Per Bay:** 2 (input field)
- Bay Width:** 2.50 m (input field)
- Level Height:** 1.40 m (input field)
- Slot Width:** 1.25 m (input field)
- ☒ Storable Slots
-

Figura 4.5: Menu dedicado à estrutura de armazenamento de uma *rack*

Como é perceptível, contêm sete níveis em termos de altura e dez níveis de comprimento (baías). Cada baía possui duas *slots* de armazenamento com as medidas indicadas na Figura 4.5:

- *Bay Width* ou largura da baía: 2,50 metros;
- *Level Height* ou altura de cada nível: 1,40 metros;
- *Slot Width* ou largura da *slot*: 1,25 metros.

Quanto à definição de políticas de armazenamento, o FlexSim oferece diversas hipóteses de armazenamento em cada *rack*. No caso do modelo a simular, visto que serão quatro *racks* em uso, é possível utilizar uma política de armazenamento diferente em cada uma. Mas, para que não surjam conflitos no armazenamento de itens, apenas uma política deve ser definida. Assim sendo, através do parâmetro pertencente às propriedades das *racks*, *Slot Assignment Strategy*, podem ser aplicadas as seguintes políticas de armazenamento:

- *Matching Labels*: os itens serão armazenados em *slots* da *rack* destinadas apenas ao seu *label*;
- *Random Bay, Level and Slot*: os itens serão armazenados em qualquer baía, nível ou *slot* desde que disponíveis;
- *Random Bay, Level and Slot with Space*: os itens serão armazenados em qualquer baía, nível ou *slot* desde que disponíveis e com espaço suficiente para os mesmos;
- *First Slot with Space*: os itens serão armazenados na primeira *slot* livre;
- *By Bay, Level and Slot Rank*: os itens serão armazenados de forma ordenada, por baía em primeiro lugar, de seguida por nível e, por fim, por *slot*, tendo em conta uma classificação atribuída previamente a cada baía, nível e *slot*;
- *By Bay, Level and Slot ID*: os itens serão armazenados de forma ordenada, por baía em primeiro lugar, de seguida por nível e, por fim, por *slot*, de acordo com a sua identidade.

4.3 Conceitos e considerações finais sobre a arquitetura dos sistemas

Para um melhor enquadramento de alguns conceitos cruciais na utilização do FlexSim, foi desenvolvida uma subsecção destinada exclusivamente à explicação

dos mesmos. Desta forma, é possível adquirir informação sobre o conteúdo explorado nos capítulos seguintes, sendo os principais conceitos:

- **label:** para a criação de itens no *software*, através das *sources*, é necessária a utilização de *labels* para os distinguir, ou seja, rótulos. No caso do modelo implementado, procedeu-se à utilização de um *label* para que todos os itens que percorram o sistema estejam identificados com o mesmo. O *label* dos itens influencia todos os processos e decisões ao longo do fluxo. Para que tudo decorra de forma correta, visto que todos terão o mesmo *label* associado, o que os distingue é o seu valor, isto é, um *label* pode ter diversos valores associados de forma a criar uma diferenciação entre itens. No Capítulo 5 será abordada a criação de *labels* e a atribuição dos seus respetivos valores.
- **ligações entre elementos:** as ligações ou conexões entre elementos permitem associá-los entre si, definindo uma lógica de fluxo no modelo. Desta forma é possível fazer com que um item saia de um recurso e siga para outro, a partir de um *Task Executer* ou não. Foram utilizados dois tipos de ligações:
 - *A-Connections*: conexões mais comuns para interligação de elementos de forma a promover as trocas de itens entre si.
 - *S-Connections*: utilizadas para conectar os elementos que executam as tarefas, ou seja, os que realizam o transporte dos itens de um elemento para outro.
- **Process Flow (PF)** : o *software*, para além da possibilidade de criação de um modelo em 3D, oferece também um painel direcionado para a lógica da simulação do modelo construído, intitulado de *process flow* ou fluxo de processos. Permite que os fluxos dos processos a simular se tornem mais fáceis de modelar, pois oferece uma maior organização e simplifica a utilização de códigos de programação através de blocos e, para além disso, permite dimensionar o projeto de acordo com as necessidades.
- **Task Executors:** são um tipo de elementos providenciados pelo FlexSim com o objetivo de movimentar itens entre outros tipos de elementos, ou seja, executam as tarefas de movimentação. Caso seja necessário, por exemplo, movimentar um item entre uma *queue* e um tapete rolante, este tipo de elementos é especializado em carregar o item presente nessa mesma *queue* e transportá-lo para o tapete rolante. Alguns exemplos de *Task Executors* são robôs, *ASRSVehicles* (ambos serão utilizados no modelo a implementar), operadores, entre outros.

- **Task Sequence:** é uma sequência de tarefas criada por um conjunto de atividades no *Process Flow*. Através desta sequência, o *Task Executer* que se encontra associado à mesma, efectuará as diversas tarefas.
- **Token:** é um elemento fictício que atua no decorrer do *Process Flow*. Tem o objetivo de simular um item que se encontra no modelo. Desta forma, tendo um *token* a representar um determinado item, é possível acompanhar o estado do mesmo nas atividades que desempenha.
- **Trigger:** traduzido para português, os gatilhos têm como objetivo desencadear uma ação num determinado ponto do modelo. Por norma, ao colocar um elemento no modelo, é possível associar um *trigger* ao mesmo. Desta forma, no momento de atuação, a ação definida irá ser executada.

Como referido no Capítulo 3, o *software* oferece uma biblioteca de elementos dividida por várias categorias. Para implementar o sistema *crossdocking* e o AS/RS, três categorias revelam-se as de utilização mais importante. Em primeiro lugar, a categoria dos *Fixed Resources*, onde estão presentes as *sources*, *queues*, *sinks*, *combiners*. Em segundo lugar, a categoria dos *Task Executors*, onde se encontram os elementos encarregues por realizar as tarefas (principalmente de transporte), como os *robots* e os *ASRSVehicles*. A terceira categoria mais utilizada é a dos *Conveyors*, onde foram bastante utilizados os próprios *conveyors*, os *decision points* e os *photo eyes*.

4.4 Conclusão do Capítulo 4

A abordagem à arquitetura dos dois sistemas a simular deve ser tida em conta como a fase imediatamente anterior à da implementação no *software*. Foram abordados os dois sistemas em causa de forma separada pois a sua implementação não tem que ser realizada simultaneamente. O sistema *crossdocking* pode ser criado numa primeira etapa e, de seguida, criar o AS/RS. A interligação entre ambos irá criar um modelo completo e integrado. Visto que a ligação entre estes dois sistemas é feita apenas por tapetes transportadores, não é necessário criar uma arquitetura para tal.

No final da implementação, um modelo completo estará a funcionar através da utilização da arquitetura dos dois sistemas em análise.

Capítulo 5

Desenvolvimento e Implementação

O presente capítulo tem o intuito de apresentar o desenvolvimento da implementação do sistema *crossdocking* e do *automated storage and retrieval system* para que seja possível realizar uma simulação, em contexto real, da integração destes dois sistemas e perceber as dificuldades existentes para um funcionamento conforme dos mesmos. Após a fundamentação teórica dos vários conceitos relacionados com os temas principais e da definição aproximada da arquitetura de ambos os sistemas, é possível iniciar a implementação com recurso ao *software* FlexSim. O *layout* de ambos os sistemas foi implementado e, através de diversos componentes e da definição de várias lógicas de fluxo, os modelos foram crescendo, bem como as suas propriedades. Visto que são modelos de características e propósitos diferentes, serão abordados em secções separadas, seguidas de uma secção de integração de modo a criar uma ponte de ligação de funcionamento entre ambos. Após uma análise ao manual de instruções fornecido pelo *software* e os seus tutoriais, o início da utilização e ligação dos mais variados elementos necessários foi feita de forma gradual, realizando testes a cada novo passo de forma a facilitar a perceção do funcionamento e da lógica existente entre os mesmos.

A divisão do presente Capítulo é sequencial, de acordo com o desenvolvimento do modelo de simulação. Ou seja, todos os processos e etapas descritas foram desenvolvidas na prática pela mesma ordem que aqui se encontram. O Capítulo está dividido em quatro Secções. A primeira descreve a implementação do sistema *crossdocking*, quer a aplicação dos elementos no modelo, quer a criação da lógica associada aos mesmos através do *Process Flow*. A Secção seguinte é destinada apenas à implementação do AS/RS. Visto que é necessário existir uma ligação entre os dois sistemas, foi desenvolvida uma Secção com o objetivo de explicar

como ambos se integram. Nesta secção é feita uma reformulação da lógica implementada na Secção relativa ao *crossdocking* com influência na Secção relativa ao AS/RS, como se pode verificar na Figura 5.1. Por fim, a última Secção destina-se a uma breve conclusão do presente Capítulo.

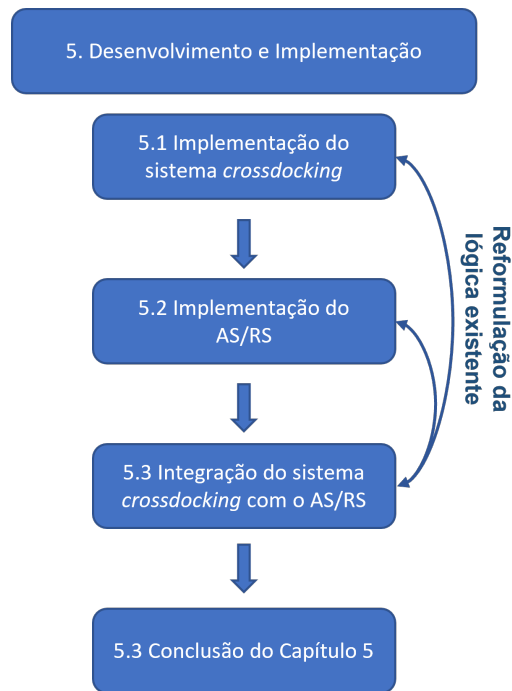


Figura 5.1: Organização do Capítulo 5

Ao longo do presente capítulo utiliza-se a palavra “itens” que tem o mesmo significado que “produtos”, termo maioritariamente utilizado até ao momento no Capítulo 2. Nas figuras, quer relativas ao próprio modelo 3D ou ao *Process Flow*, é maioritariamente utilizada a língua inglesa, devido a conjugar o que foi criado com a língua utilizada pelo *software*.

5.1 Implementação do sistema crossdocking

A implementação do sistema *crossdocking* passa por várias etapas. A implementação do *layout* a simular é o primeiro passo, seguido da estruturação do *process flow* com vista a definir a lógica que controla a simulação vista como um todo. De seguida, é estruturado e desenvolvido um *process flow* menos generalizado, direcionado a objetos. Neste caso, direcionado às ordens de despacho e de que forma é possível satisfazê-las da forma mais eficiente possível, sem ter em consideração qualquer tipo de armazenamento intermédio.

5.1.1 Implementação inicial do layout e fluxo desejado

Tendo em conta a arquitetura do sistema *crossdocking* apresentada no Capítulo 4, a lógica do fluxo a simular passa pela entrada de itens através de uma ou várias *sources* que, usando tapetes rolantes, transportam os mesmos até uma das duas zonas disponíveis. Os itens poderão ser colocados em unidades acumuladoras (UA) (representadas por *queues*) ou enviados para armazenamento (AS/RS). Os produtos que reentram no *crossdocking* provenientes do AS/RS, fazem-no a meio do fluxo.

Foram utilizadas cinco *sources* diferentes, sendo que cada uma representa uma entrada no sistema. Em comparação com circunstâncias em ambiente real, cada *source* pode representar um veículo de entrada. A entrada de produtos no fluxo é chamada de logística de *inbound*. Cada *source* está conectada ao tapete transportador, o qual encaminha os itens desde a sua origem até ao seu destino que, neste caso, ou será uma UA ou o armazém. Inicialmente, de forma a simular a existência de diversos itens, foram criados cinco tipos diferentes, cada um correspondente a uma *source*. Numa primeira abordagem, cada tipo de item pode ser representado por seis cores, exceto os itens provenientes da *source inbound_5* que podem ser representados por sete cores. Isto equivale à criação de trinta e um itens diferentes. Contextualizando, cada cor pode corresponder a itens provenientes de um mesmo fabricante e a diferentes formas às referências que esse fabricante envia para o mercado, neste caso, para o sistema *crossdocking*.

A diferenciação por cor não se considera necessária, porém, de forma a entender melhor o funcionamento do sistema nesta etapa inicial, definiu-se a sua utilização. Os itens utilizados estão representados na Figura 5.2.

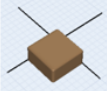
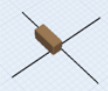
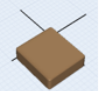
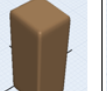
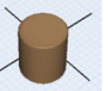
Source	Inbound_1	Inbound_2	Inbound_3	Inbound_4	Inbound_5
Item					
Cores	6	6	6	6	7

Figura 5.2: Itens utilizados na fase inicial da simulação

Um fator de importância elevada relacionado com os itens criados passa pela definição do seu *label*, conceito esclarecido no Capítulo 4. Caso haja uma gralha ou incongruência com este parâmetro, o fluxo poderá ter erros de processamento associado e as ordens de despacho que serão implementadas numa fase mais avançada da simulação não irão ser satisfeitas de maneira conforme.

Foi criado um único *label* ‘Type’ que terá valores associados conforme a cor de cada item, como está representado na Tabela 5.1. O ‘Type value’ indica qual o valor que o *label* toma de acordo com a cor do item.

Tabela 5.1: Valores associados ao *label* ‘Type’ dos itens criados nas *sources* do *crossdocking* por cor

Cor	Vermelho	Verde	Azul	Amarelo	Laranja	Roxo	Preto
‘Type value’	1	2	3	4	5	6	7

Outra hipótese passa por definir o ‘Type value’ de acordo com a forma do item. Isto é, um item criado pela *source Inbound_1* terá um valor de 1 e assim sucessivamente, como demonstra a Tabela 5.2. Neste caso só cinco valores serão criados pois só cinco formas de itens distintas estão definidas na simulação. Aqui também é possível distinguir os itens pela cor e pela forma, simultaneamente. Ou seja, os itens provenientes, por exemplo, da *source Inbound_2* possuem todos a mesma forma e, através da adição de um *trigger*, a mesma cor. Desta maneira, cada formato de item teria uma cor diferente.

Tabela 5.2: Valores associados ao *label* ‘Type’ dos itens criados nas *sources* do *crossdocking* por forma

Source	Inbound_1	Inbound_2	Inbound_3	Inbound_4	Inbound_5
‘Type value’	1	2	3	4	5

Ao longo da implementação do *layout*, de forma a associar os diferentes elementos para permitir que os itens percorram o sistema, foram utilizados dois tipos de ligações, *A-Connections* e *S-Connections*, as quais se encontram descritas no Capítulo 4. Até esta etapa, e ao longo da implementação desta ideia embrionária de funcionamento do sistema, apenas foi necessário utilizar *A-Connections*. Caso nesta fase se encontrassem a ser utilizados *Task Executors*, ou seja, elementos que realizam transporte de itens, é que se usariam ligações do tipo *S-Connections*. A Figura 5.3 representa um exemplo de ambos os tipos de ligações de forma a elucidar a lógica entre elementos.

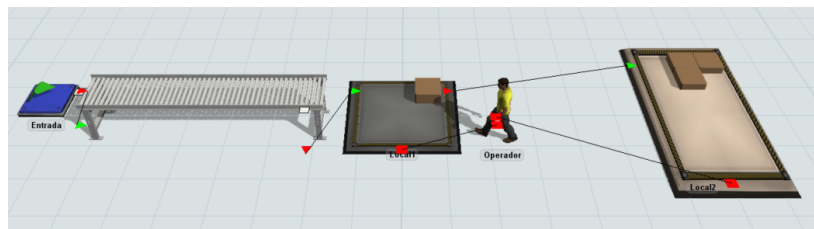


Figura 5.3: *A-Connections* e *S-Connections* entre elementos

Entre os elementos ‘Entrada’, tapete rolante, ‘Local1’ e ‘Local2’, as ligações existentes são do tipo *A-Connections* enquanto que entre o ‘Local1’ e o ‘Operador’ e entre o ‘Operador’ e o ‘Local2’ as ligações existentes são do tipo *S-Connections*. Isto porque o ‘Operador’ é considerado um *Task Executer* devido a ser o mesmo a movimentar os itens entre locais. Neste exemplo o *Task Executer* é representado por um operador, porém pode ser representado por outros elementos, tal como um AGV.

Tendo parte do *layout* implementado, com todas as ligações feitas entre as *source* e os tapetes, foram criadas seis UA de forma a acumularem diversos itens. Uma UA é representada pelo elemento *queue*. De seguida, o objetivo passa por atribuir a uma UA itens de uma determinada cor ou de uma determinada forma, dependendo da abordagem, através da utilização de vários *decision points* (DP). Assim que um item atinja o primeiro DP, este irá analisá-lo e, consoante a sua cor ou forma, atribui o mesmo a uma UA específica.

- **Distribuir itens pela cor:** de forma a que o *software* permita a diversificação de itens e ações no modelo, é necessário proceder à criação de *triggers*. Isto é, no momento em que um item seja criado ou atinja um outro elemento que contenha um *trigger*, desencadeia-se uma ou várias reações no fluxo. Poderá ser alterado o seu *label*, o seu nome, a sua cor, a sua posição ou outros parâmetros. Em cada *source* foi criado um *trigger* ‘On Creation’, com a função ‘Set Label and Color’ (Figura 5.4). De forma a cada *source* criar seis cores diferentes, no campo ‘Value’ os itens e as suas seis cores são definidos de acordo com uma distribuição uniforme.

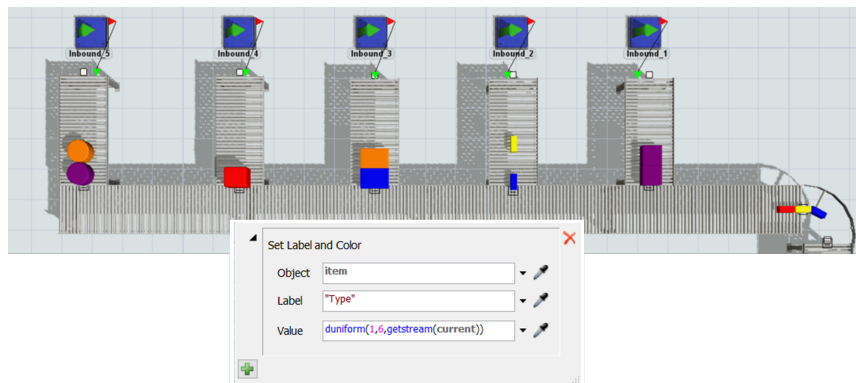


Figura 5.4: Criação de itens diferenciados por forma e cor

De seguida, ao atingirem o primeiro *decision point* ou DP1, os itens serão encaminhados para uma UA de acordo com a sua cor. De forma a simular a presença de produtos para armazenamento, a *source inbound_5* gera uma cor específica (daí serem sete cores diferentes e não seis no *inbound_5*),

que ao chegar ao DP1, este percebe que é um item para armazenamento e encaminha-o para uma *queue* que, nesta fase, simula o efeito de armazém. Os itens irão viajar ao longo do sistema até atingirem o DP1 que determina para qual UA se devem dirigir. No DP1, através da criação de um *trigger* ‘On Arrival’ com a função ‘Set Label’, será atribuída uma UA a cada item assim que estes cheguem à posição do mesmo. Na Figura 5.5 é possível verificar no campo ‘Destination’ a expressão utilizada que envia cada ‘Type’ para uma UA distinta.



Figura 5.5: Definição de parâmetros no DP1

Como é perceptível, independentemente da forma do item, a distribuição foi realizada por cor, sendo que cada UA tem uma cor destinada.

- **Distribuir itens pelo seu formato:** relativamente ao funcionamento do processo e do fluxo, a distribuição de itens pelo seu formato é semelhante. Ao invés de se proceder à criação de itens com seis cores diferentes por cada *source*, apenas se cria uma cor padrão transversal a todos, diferenciando-se apenas pelo seu valor. Quer na criação de itens quer na chegada dos mesmos ao DP1, os *triggers* utilizados mantêm-se, havendo alteração de parâmetros no *trigger* ‘On Creation’.

Assim, os itens criados são diferenciados pelo seu valor. Posteriormente, com a evolução do projeto, o valor associado ao *label* de cada item será responsável pela distinção dos mesmos nas ordens de despacho que entram no sistema. Através disto, será possível enumerar o número de itens necessários, de cada tipo, para satisfazer uma ordem em curso no modelo. A Figura 5.6 demonstra os dois *triggers* utilizados na distribuição de itens pela sua forma. É definido um *trigger* ‘Set Label’ nas *sources*, com um *label* comum (‘Type’) que tem um valor associado (valor difere em cada *source*). O *trigger* ‘Send Item’ é relativo ao DP1 e tem o objetivo de enviar os itens para a UA correspondente ao seu valor de *label*.

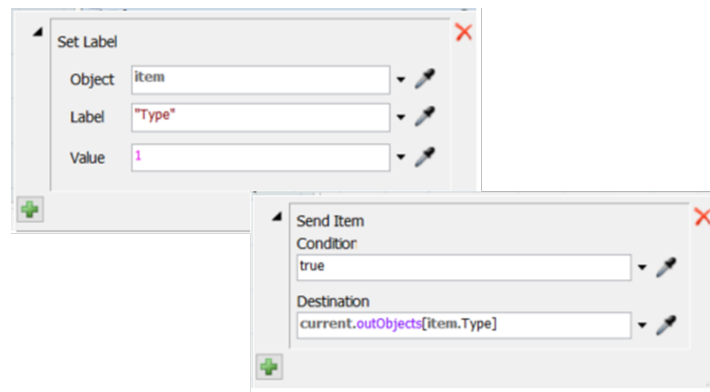


Figura 5.6: *Triggers ‘On Creation’ e ‘On Arrival’* na distribuição por formato

Nesta fase inicial, este processo revelou-se bastante importante, tendo sido útil para uma melhor assimilação das dinâmicas de criação de itens e de distribuição dos mesmos por diversos elementos através de *triggers*.

Foram utilizados também dois *photo eyes*, o *photoeye1* (PE1) e o *photoeye2* (PE2) para que seja controlado o distanciamento entre itens de modo a não choquem entre si no momento de passagem para as UA. Como referido no Capítulo 4, os *photo eyes* funcionam como um sensor de deteção de itens. Neste caso, ambos estão colocados de forma sequencial no tapete rolante, como é perceptível na Figura 5.7.

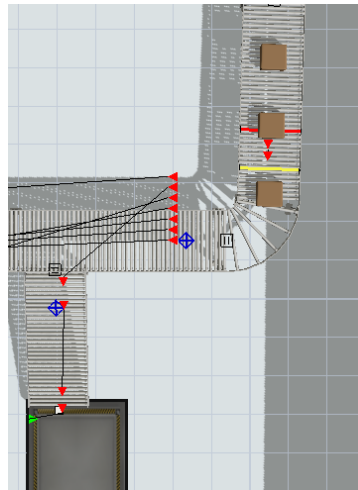


Figura 5.7: *Photo eyes* utilizados no modelo

O PE1 está representado na figura através da cor vermelha e o PE2 através da cor amarela. É também visível que um item está a atingir o PE1 (nomeado item_1) e que o item em frente atingiu o PE2 momentos antes (nomeado item_2). O item_1, ao atingir o PE1, foi obrigado a fazer um compasso de espera de modo

a criar um distanciamento entre ele e o item_2. Só lhe foi concedida permissão para seguir em frente quando o item_2 ultrapassou o PE2. Para possibilitar a existência de um espaço entre estes itens, foram utilizados os seguintes *triggers*:

- PE1: *trigger* ‘On Block’ com a função ‘Acquire Restricted Area’.
- PE2: *trigger* ‘On Clear’ com a função ‘Release Restricted Area’.

Assim, foi possível implementar um espaço entre itens para prevenir uma eventual colisão entre os mesmos nas etapas seguintes do modelo.

Tendo a criação de itens estruturada e a distribuição de produtos através de DP funcional, de acordo com o tipo de itens existentes, o próximo passo é relativo à colocação de itens numa paleta. Assim sendo, foi criada uma *source* intermédia no fluxo, nomeada de ‘*Pallet_source*’, apenas com o objetivo de fornecer paletes para que os itens sejam embalados e expedidos. Estas paletes são fornecidas diretamente para um elemento que fará a combinação dos itens com a mesma. Para a combinação itens-paleta, inicialmente foram utilizados *combiners*, elementos utilizados para agrupar vários itens de fluxo à medida que vão viajando pelo modelo. Visto que o modelo apresenta seis UA, considerou-se plausível a existência de três *combiners*. Através de *A-connections*, a ‘*Pallet_source*’ conecta-se com os *combiners*.

Neste momento da simulação, surgiram diversas dificuldades, não relacionadas com a lógica do fluxo mas sim com a implementação das ideias no *software*. Deste modo, várias alterações e adições foram realizadas ao modelo. As principais questões remetem para a criação de ordens de despacho e de que maneira é que as mesmas seriam cumpridas no formato implementado até ao momento. Portanto, para conseguir ultrapassar estas dificuldades, recorreu-se a uma ferramenta própria do FlexSim, já referenciada, o *Process Flow*. Foram utilizados dois tipos de PF, interligados. Um *General Process Flow*, que permite a criação de um “cérebro” centralizado para todo o modelo. Este tipo de PF são os ideais para criar lógica que necessite de ser globalmente acessível por todos, ou quase todos, os elementos do modelo de simulação. Para além disto, foi também utilizado um *Order Process Flow* (OPF), ou fluxo de processos de recursos fixos, neste caso direcionado às ordens de despacho. Destina-se à criação da lógica de um ou mais intervenientes do modelo, neste caso das ordens de despacho. Permite adicionar lógica extra a qualquer elemento e a criação de uma lógica personalizada. O FlexSim fornece uma biblioteca dedicada apenas à construção de PF com diversos ativos disponíveis, facilitando a criação do fluxograma e as associações ao modelo.

5.1.2 Estruturação e desenvolvimento do Process Flow

Em termos de implementação e simulação, um aglomerado de itens entra no sistema, e é transportado pelos *conveyors*, ou tapetes rolantes, até atingirem uma estação ou um determinado ponto onde serão distribuídos por diferentes UA de acordo com ordens de despacho previamente enviadas. Estas ordens têm como objetivo a definição de qual item irá para um determinado destino. De forma a melhorar a eficiência da simulação e aplicar a lógica mais funcional, o PF tornou-se fundamental. Através de vários fluxogramas, é possível determinar e acompanhar o percurso dos diversos itens existentes no sistema. Paralelamente à simulação no modelo, ao invés de se utilizarem itens, no PF utilizam-se *tokens*. Servem maioritariamente para acompanhar o fluxo do sistema, o que permite monitorizar cada passo executado e, eventualmente, perceber onde existem mais incongruências, bem como detetar o *bottleneck* do sistema, caso exista.

A lógica pensada para o funcionamento do sistema foi estruturada por duas fases distintas. A primeira fase é dedicada à entrada de itens no *conveyor system* (CS), ou sistema de tapetes rolantes, e de que forma é que se irão dividir de modo a satisfazer a lista de ordens existente. A segunda fase é destinada à colocação de cada ordem de despacho na paleta e ao envio para a expedição.

Assim sendo, o primeiro passo passa por importar um tabela desenvolvida através do Microsoft Excel [31]. O FlexSim permite importar tabelas em formato Excel desde que seja estruturada de forma semelhante à suportada pelo mesmo. Neste caso, cada coluna representa uma ordem de despacho e cada linha um tipo de item. Visto que a simulação cria itens de cinco tipos diferentes, serão cinco as linhas existentes, sendo que o que varia é o número de ordens de despacho, ou seja, de colunas. A Figura 5.8 representa oito ordens de despacho, com o número de itens de cada tipo correspondente a cada uma. Considera-se que o número de itens correspondente a cada ordem foi definido aleatoriamente.

	Ordem 1	Ordem 2	Ordem 3	Ordem 4	Ordem 5	Ordem 6	Ordem 7	Ordem 8
Tipo 1	1	1	2	2	3	0	0	1
Tipo 2	2	3	3	2	0	0	2	2
Tipo 3	4	1	1	2	4	0	3	4
Tipo 4	3	4	2	1	0	1	4	3
Tipo 5	1	2	4	0	1	1	0	1

Figura 5.8: Exemplo de tabela de ordens de despacho importada do Microsoft Excel

De seguida, passando à implementação e criação do PF, a lógica deve ser definida previamente, já com uma ideia estruturada. Foram exploradas várias lógicas do ponto de vista teórico. O ideal, para um sistema *crossdocking*, passa por uma calendarização de entrada de itens e de ordens de despacho. Quanto à entrada de itens, simulando um veículo de entrada, este deve comparecer no

sistema para descarga de material a uma hora definida, através da utilização de uma tabela importada a partir do Microsoft Excel, com valores de hora de entrada e dos itens que pretende descarregar. Quanto às ordens de despacho, deveria ser possível indicar a hora do dia em que chegam ou que são emitidas e estabelecer esse horário de acordo com uma distribuição estatística. Porém a lógica delineada para a implementação difere da descrita. A entrada de itens no sistema *crossdocking* será feita de forma faseada. Cada *source* irá enviar para o sistema um tipo de item de acordo com um intervalo de tempo ou de acordo com uma distribuição estatística. Não foi incluída a calendarização na chegada de itens ao sistema. Quanto às ordens de despacho, tal calendarização também não foi incluída, porém, de forma a contornar parcialmente esta falha, foi definido um ‘*Offset time*’ que permite que as ordens cheguem ao sistema desfasadas, com um intervalo de tempo definido a separá-las. Este parâmetro será abordado ao longo do presente capítulo.

Posto isto, a primeira lógica a ser implementada no PF passa pelo registo de itens que entram no sistema com o intuito de satisfazerem uma ordem de despacho. Visto que o fluxo de itens no sistema é elevado, de forma a garantir que os mesmos ficam registados no sistema, devem ser guardados numa lista. Assim que sejam utilizados para uma ordem, serão retirados da mesma lista, deixando de pertencer ao sistema e encaminham-se para a zona de expedição (*outbound*). Para que a criação desta lógica fosse possível, foi necessário proceder à criação de um DP (DP11) e à utilização de um já existente (DP1). A Figura 5.9 demonstra a localização de ambos, a azul, para uma melhor interpretação.

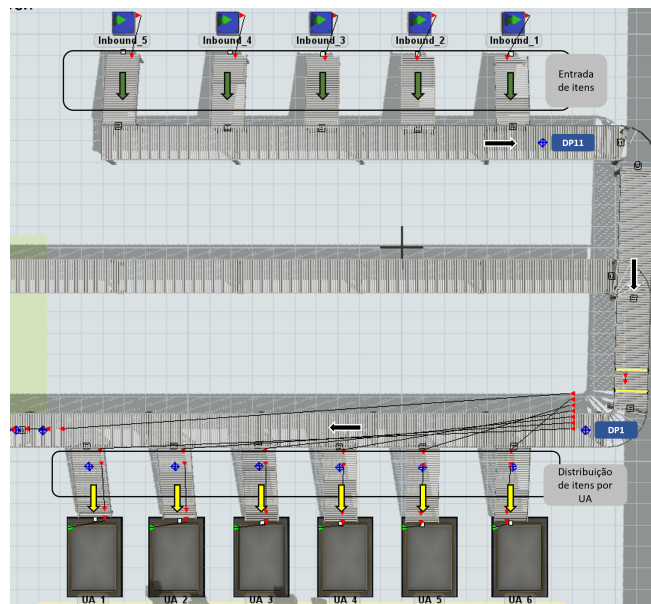


Figura 5.9: Colocação estratégica dos DP1 e DP11

Como é perceptível, após a chegada ao DP1, os itens serão distribuídos pelas UA. A cada UA, assim que o sistema entra em funcionamento, é atribuída uma ordem de despacho, fazendo com que cada uma represente uma ordem.

O DP11 foi colocado estrategicamente no CS de forma a que os itens, assim que sejam emitidos para o sistema pelas *sources*, o atinjam. Nesse momento, são colocados pelo PF na lista criada. Quando tal acontece, no contexto do PF, um item passa a ser representado por um *token*. Continuando a percorrer o CS, o item representado por um *token* no PF, ao atingir o DP1, será retirado da lista, o que indica que foi utilizado para satisfazer uma ordem de despacho. Aqui, o item é enviado para uma UA com vista a preencher os requisitos da ordem que a mesma representa. Esta lógica está representada na Figura 5.10.

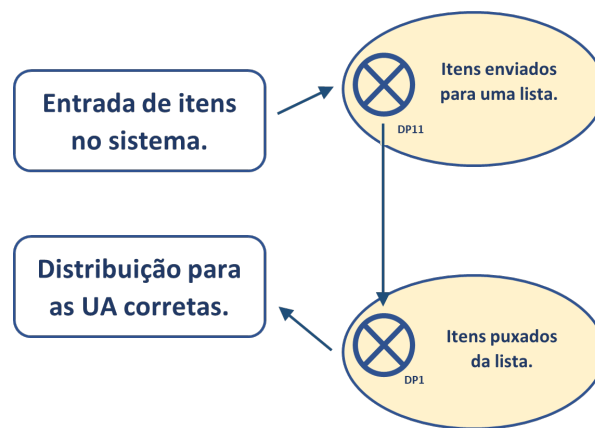


Figura 5.10: Lógica a implementar no funcionamento inicial do *Process Flow*

Tanto o item como o *token* continuam presentes no sistema nas etapas seguintes da simulação. Relativamente às ordens de despacho, é possível aplicar regras de prioridade, porém, visto que todas são consideradas importantes e o objetivo principal do sistema *crossdocking* é a movimentação rápida de produtos, a ordem de execução é definida pela tabela de ordens de despacho. Serão satisfeitas de forma sequencial, cumprindo a sua posição. A ordem presente na coluna número um será a primeira a ser satisfeita, a ordem da coluna número dois a segunda e assim sucessivamente.

Com a lógica inicial definida, a sua implementação foi dividida em dois tipos de PF. Um generalizado a todo o modelo e outro mais específico para as ordens de despacho. De modo a satisfazer o requerido, o PF generalizado é constituído por diversos objetos. “Lista: Entrada de itens no Conveyor”, é um objeto da categoria *shared assets*, onde os *tokens* podem reivindicar ou ser libertados em determinados pontos do fluxo. É nesta lista que os mesmos irão ser introduzidos ao entrarem no sistema e libertados ao entrarem para as UA. Encontra-se interligada a dois outros grupos de fluxo, um orientado para introduzir um item na lista e o outro

para o retirar, assim que estiver definida a sua UA de destino. Esta conexão é visível na Figura 5.11.

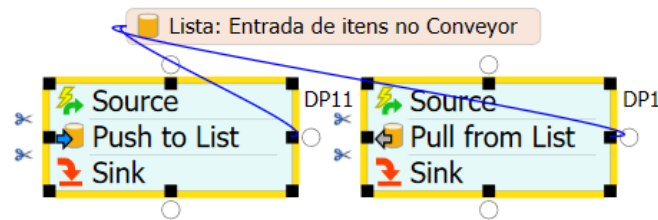


Figura 5.11: *Process Flow* generalizado para o sistema *crossdocking*

Como é possível verificar, existe uma conexão (linha azul na Figura 5.11) entre a lista e os dois restantes grupos de atividade. Quanto ao grupo constituído por uma *Source*, um *Push to List* e um *Sink*, o seu objetivo passa por inserir um item na lista ao percorrer o DP11 (considerado a *source* da atividade). Já o grupo constituído por uma *Source*, um *Pull from List* e um *Sink*, funciona de forma a que o item, já existente na lista devido à atividade realizada pelo outro grupo, assim que chegue ao local onde se encontra o DP1 (utilizando um *trigger* ‘*On Arrival*’), seja retirado da lista, caso ainda não tenham sido retirados para satisfazer uma das ordens de despacho existentes. Isto é, caso um item seja necessário numa das ordens existentes, este é retirado da lista antes de atingir o DP1, já com destino traçado. Desta forma é possível que múltiplas ordens possam ser atribuídas a cada UA sem criar problemas.

5.1.3 Estruturação e desenvolvimento do Order Process Flow

De forma a completar o raciocínio e a aplicar a lógica requerida às ordens de despacho existentes, foi criado um ‘*Order Process Flow*’. Aqui encontra-se todo o cérebro do processo, associando diversos fatores, desde a criação dos *tokens* ao transporte dos itens entre as UA e os objetos de junção dos mesmos com as paletes. Diversos conceitos foram explorados e diversas adições serão realizadas no modelo.

Para criar uma ponte com a secção anterior, de forma resumida, será gerada uma tabela com o intuito de especificar quais as ordens de despacho, ou seja, as quantidades de cada tipo de item. No momento em que os itens passam pelo DP11, são inseridos numa lista através da atividade ‘*Push to List*’, no PF. De seguida, são retirados da lista através do OPF e enviados para a UA correta. Assim que a ordem estiver completa, ou seja, todos os itens da ordem de despacho estiverem presentes na mesma UA, serão movidos com recurso a um robô automatizado para um elemento específico cuja função é agrupar os itens da ordem numa palete, embalá-la e enviá-la para expedição.

Com vista a uma maior organização no ambiente de trabalho do *software* e de maneira a facilitar a compreensão das fases de desenvolvimento deste fluxo, o OPF foi dividido em três blocos principais continuados, com a possibilidade de ligação entre ambos, sendo eles:

1. ‘*Start: Get Data From Table*’: recolha de dados da tabela de ordens de despacho.
2. ‘*Get Ordered Items Off List/Place in Queue*’: retirar os itens necessários para uma ordem da lista e colocá-los na UA.
3. ‘*Transport Items to Combiner*’: transportar os itens para o *combiner* para serem agrupados numa paleta.

A Figura 5.12 representa uma visão geral dos três blocos criados.

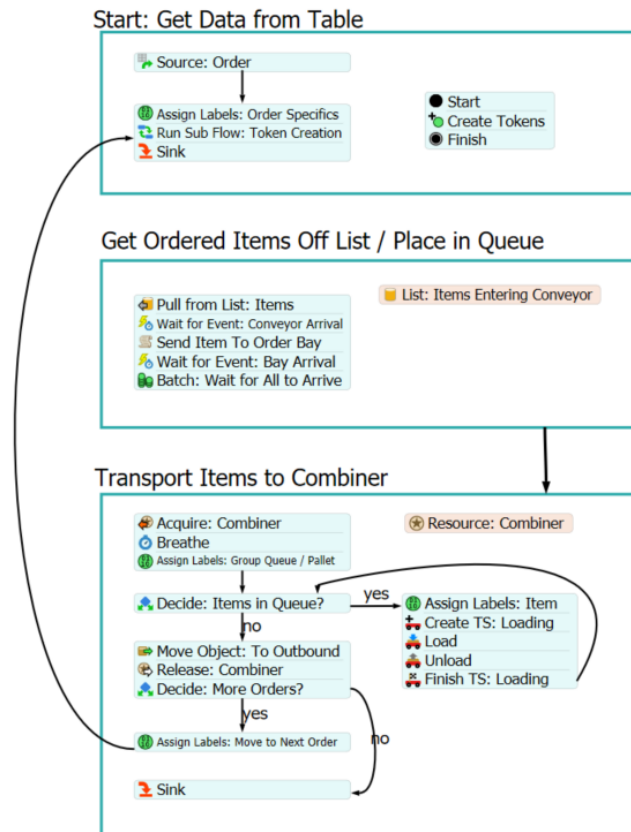


Figura 5.12: Visão geral do *Order Process Flow*

Relativamente ao bloco 1, é responsável por retirar as ordens de despacho da tabela. A Figura 5.13 demonstra a estrutura deste bloco.

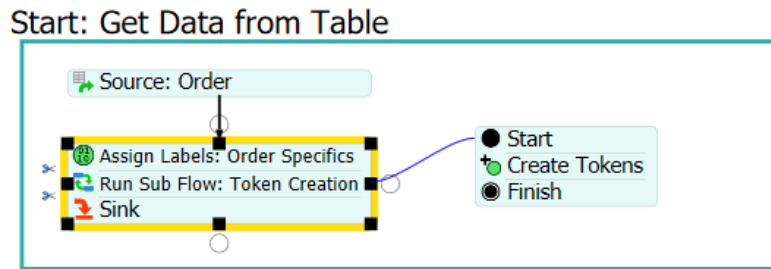


Figura 5.13: Fluxo de obtenção de dados da tabela de ordens de despacho

Como *source*, é utilizada uma *Schedule Source*. Aqui foi colocado no parâmetro *Offset Time* uma distribuição estatística com o objetivo de desfazar a chegada das ordens de despacho. Indica que entre cada ordem de despacho emitida por esta *source* existirá um desfasamento entre as mesmas. Desta forma é possível uma aproximação à entrada de ordens de despacho no sistema. A distribuição estatística utilizada foi do tipo $duniform(5,20,0)$, que seleciona números aleatoriamente dentro de um intervalo. A Figura 5.14 representa a distribuição utilizada.

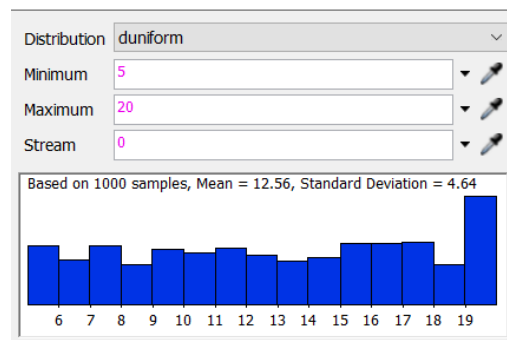


Figura 5.14: Distribuição estatística utilizada na chegada de ordens de despacho ao sistema

O desfasamento entre a chegada de ordens de despacho ao sistema pode variar de acordo com os seguintes parâmetros da distribuição:

- *minimum*: o menor número possível que pode ser gerado aleatoriamente;
- *maximum*: o maior número possível que pode ser gerado aleatoriamente;
- *stream*: de que forma são gerados os números dentro do intervalo definido. Não utilizado.

Assim, a chegada de ordens de despacho ao sistema varia entre cinco a vinte segundos, de acordo com a distribuição estatística aplicada ao parâmetro *Offset Time* da atividade *'Schedule Source'*.

Posto isto, tendo a origem de informação, o próximo passo é baseado na atribuição de *labels* ou rótulos da ordem de despacho, através da atividade *'Assign Labels: Order Specifics'*. Estas duas atividades estão interligadas através de uma seta de cor preta, como é visível. Sempre que uma ordem de despacho seja introduzida na tabela, esta será a primeira atividade a desenvolver, seguida da atribuição de rótulos. Estes criam um *array* dos dados da tabela e contam o número de *tokens* necessários. Após estar identificada, foi agrupada uma atividade com o intuito de executar um *subflow* exclusivamente para a criação de *tokens* como é visível. A informação relativa aos *tokens* é armazenada num grupo criado e nomeado de *'OrderBays'*. É através deste grupo que reunindo informação será possível distribuir os itens pelas UA.

A atividade *'Assign Labels'* cria dois rótulos diferentes. Um para os itens existentes na ordem de despacho e outro para o número total de itens na ordem. A atividade seguinte, *'Run Subflow'* só irá ser executada quando o número de *tokens* for igual ao tamanho da ordem, ou seja, igual ao rótulo do número total de itens da ordem. Desta forma previne-se a criação de *tokens* desnecessários ao processo, evitando eventuais conflitos de fluxo. A atividade do *'Sink'* tem como objetivo terminar a atividade do presente bloco e preserva um histórico da criação de *tokens*.

O bloco 2, *'Get Ordered Items Off List/Place in Queue'*, tem como principal função retirar os itens da lista e colocá-los na UA destinada à ordem a satisfazer. As suas atividades estão representadas na Figura 5.15.

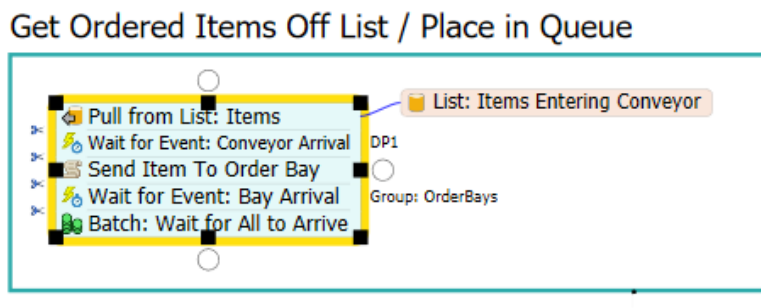


Figura 5.15: Fluxo de distribuição de ordens de despacho pelas UA destinadas

Este bloco vai buscar os itens na lista (*'List: Items Entering Conveyor'*) baseado no seu tipo, o qual foi atribuído na atividade *'Create Tokens'* do bloco 1. Como é possível verificar na Figura 5.15, a atividade *'Pull from List: Items'* encontra-se conectada à própria lista *'List: Items Entering Conveyor'*. Esta é a

lista utilizada no PF generalizado, daí o seu tipo de recurso ser um *shared asset* pois pode ser partilhado e utilizado por diversos PF. Os itens são retirados da lista momentos antes de atingirem o DP1, localizado no CS antes das divisões existentes para cada UA. Para que tal seja permitido, criou-se uma atividade ‘*Wait for Event: Conveyor Arrival*’, cujo objetivo é realizar um *match* entre os itens que chegam ao DP1 e os que estão presentes na lista. A Figura 5.16 representa as propriedades desta atividade, considerada importantíssima para que seja possível satisfazer as ordens de despacho.

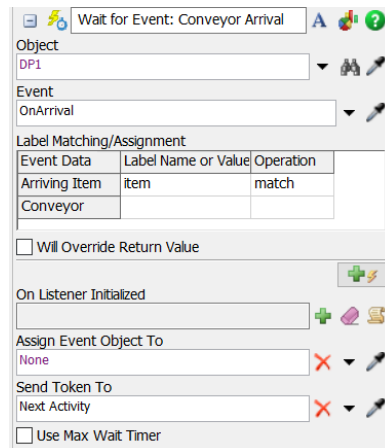


Figura 5.16: Propriedades da atividade ‘*Wait for Event: Conveyor Arrival*’

Analisando a Figura 5.16 o *object* corresponde ao DP1 pois é neste elemento que se dará uma alteração no fluxo. Foi criado também um evento, semelhante aos *triggers* já referidos anteriormente, pois a atividade só atua no momento em que o item chega ao local do DP1 daí ser um evento ‘*On Arrival*’. Este evento vai desencadear uma comparação entre os rótulos dos itens e, assim que corresponderem ao esperado, o processo prossegue para a próxima atividade, que consiste no envio do item presente no tapete para uma UA. Esta atividade é necessária pois é um elemento de personalização de lógica, visto que o item que se pretende transportar para uma determinada UA está presente no tapete. Para que resulte, a atividade permite definir que o *token* referente ao item deve ser enviado para uma UA, evento que é decifrado na atividade seguinte, ‘*Wait for Event: Bay Arrival*’. Aqui, os itens chegam à UA e como o grupo criado anteriormente (‘*Order Bays*’) contém informação referente aos mesmos e às UA disponíveis, são finalmente distribuídos. Por fim, a atividade relativa à criação de lotes, ‘*Batch: Wait for All to Arrive*’ tem como objetivo garantir que todos os itens de uma ordem de despacho foram coletados.

Nesta fase da lógica do fluxo, foram adicionados diversos recursos ao modelo de modo a completar o sistema *crossdocking*. Após os itens de uma ordem de despacho estarem todos reunidos numa UA, o próximo passo passa por agrupar

esses mesmos itens numa paleta. Através de um recurso *source*, nomeado de *pallet_source*, são geradas as paletes suficientes para satisfazer este agrupamento. Ou seja, uma paleta por ordem de despacho. Visto que existe a necessidade de combinar dois itens distintos, foi necessário recorrer a um novo recurso. Inicialmente optou-se pela utilização de um *combiner*, porém, devido ao facto deste tipo de recurso possuir a sua própria lógica, foi utilizada uma *queue* cuja aparência foi transformada num clone de um *combiner*. É mais fácil utilizar algo semelhante a um *combiner* sem a sua lógica anexada pois permite personalizar a lógica do recurso no OPF. A *queue* utilizada simula um paletizador no modelo e, tal como foi apresentado no Capítulo 4, Figura 4.1, foi também acrescentada uma máquina destinada ao embrulho de cada ordem. Tendo em conta todos os recursos fornecidos pelo *software*, foi necessário proceder à personalização de um *processor* de forma a que o seu funcionamento e aparência fossem semelhantes a este tipo de máquinas. A Figura 5.17 mostra a diferença entre um *processor* normalizado e a máquina de embrulho construída.

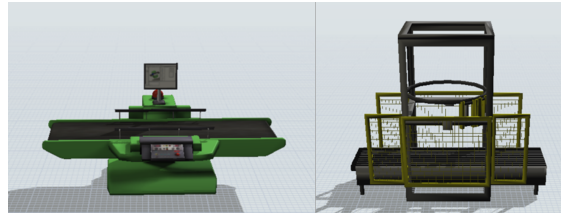


Figura 5.17: Transformação *Processor* -> *Wrapper*

O transporte entre qualquer recurso no software é realizado por *Task Executors* e, neste caso, o transporte dos itens entre a UA e os paletizadores é realizado por um robô.

O modelo de simulação apresenta três paletizadores, três máquinas de embrulho (*Wrappers*), e um robô de transporte de itens, como é nítido na Figura 5.18.

Logicamente, os três corredores visíveis, na área alaranjada, estão conectados a recursos *sink*, nomeados de *Outbound_1/2/3*. que finalizam o processo e o funcionamento do sistema *crossdocking*.

Realizada a implementação destes recursos, o passo seguinte irá realçar a lógica por detrás do transporte de itens, do agrupamento dos mesmos por parte dos paletizadores e da sua expedição. Ou seja, prosseguir para o bloco 3 do OPF, destinado ao transporte dos itens entre as UA e os *combiners*. A Figura 5.19 demonstra as atividades implementadas para esta fase da simulação.

Sucintamente, o bloco 3 tem como objetivo transportar os itens para os *combiners*, onde serão colocados em paletes e enviados de seguida para expedição.

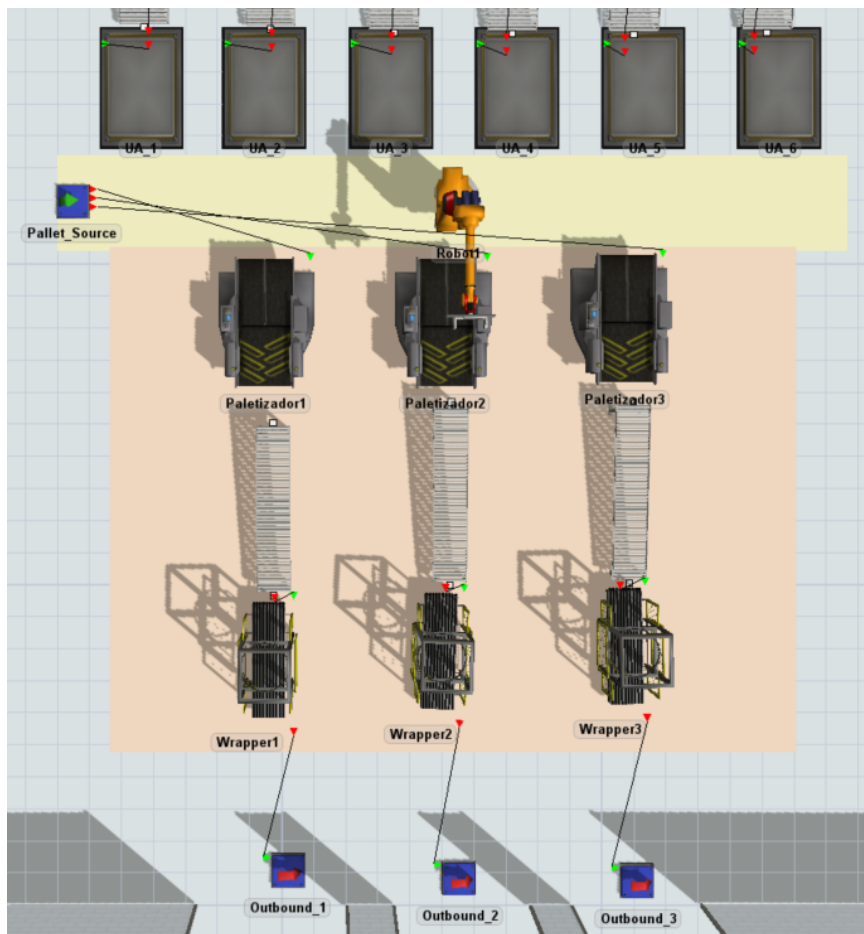


Figura 5.18: Implementação de robôs de transporte, paletizadores e máquinas de embrulho (*wrappers*)

Antes de serem expedidos passam por um processo de embrulho, porém este não necessita de estar incluído no OPF visto que é um processo meramente representativo. A ordem é satisfeita, embrulhada e enviada para expedição. A atividade do embrulho é pré-definida, sendo possível controlar o seu *Setup Time* e *Process Time*. A expedição é simulada pelo recurso *sink*.

Tendo em conta o que está representado na Figura 5.19, a mesma só é iniciada assim que os itens estiverem prontos para serem paletizados. Para que isto seja exequível, os blocos 2 e 3 estão interligados entre si, adquirindo assim um seguimento lógico continuado.

Num *process flow*, sempre que é necessário utilizar mais do que um recurso para um conjunto de atividades, é utilizado um *shared asset*, mais concretamente um *Resource*. Funciona como um ativo de fornecimento limitado que pode ser adquirido e libertado. Ao utilizar esta ferramenta, o PF está a adquirir um

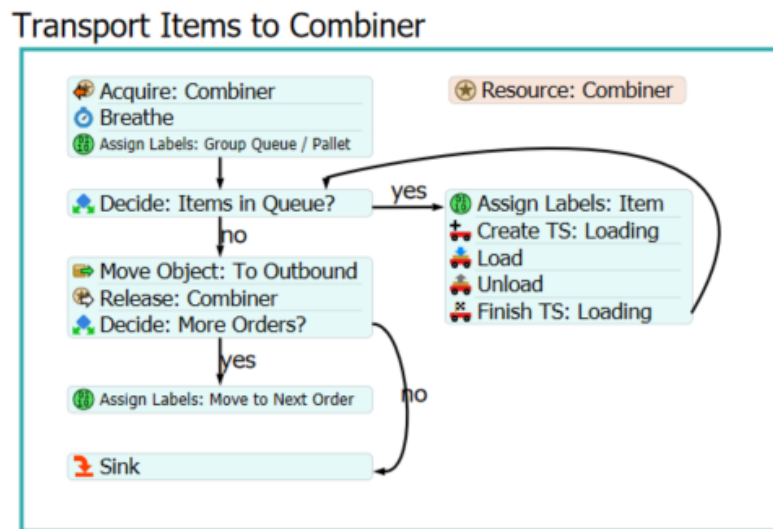


Figura 5.19: Lógica aplicada ao bloco responsável pelo transporte de itens entre UA e *combiners*

recurso. Assim, ao inserir no fluxo um ‘*Resource: Combiner*’, é necessário adquiri-lo e, ao fim da sua utilização, libertá-lo. O ‘*Resource: Combiner*’ é referenciado a um grupo nomeado de ‘*Combiners*’, onde estão presentes todos os que estão a ser utilizados na simulação.

No primeiro grupo é então adquirido o *combiner* e designada qual a UA que irá enviar os itens para esse determinado *combiner*, através da atividade ‘*Assign Labels: Group Queue/Pallet*’. É um processo de atribuição simples, sendo a única restrição o facto da UA estar ou não livre. Visto que existem três *combiners* diferentes é possível realizar este procedimento três vezes para três UA diferentes, satisfazendo três ordens de despacho.

Adquiridos os *combiners*, o fluxo segue para a próxima atividade que funciona como um ponto decisivo. Os *tokens* entram na atividade ‘*Decide: Items in Queue?*’, onde é decidido qual o próximo passo a executar, como expressa a Figura 5.20.

Dependendo da decisão tomada, o processo a executar varia:

- **SIM:** caso existam itens na UA em decisão, irá ser executado um fluxo secundário com vista à carga e descarga desses mesmos itens. Para isso, foram agrupadas várias atividades que, de forma sequencial, irão completar este processo. É criada uma *Task Sequence*, ou uma sequência de tarefas. A primeira atividade, ‘*Create TS: Loading*’ serve para a criação dessa mesma sequência e para definição de que elemento irá realizar as atividades. É definido o *Robot1* como o *Task Executer*. Isto indica que o transporte de itens

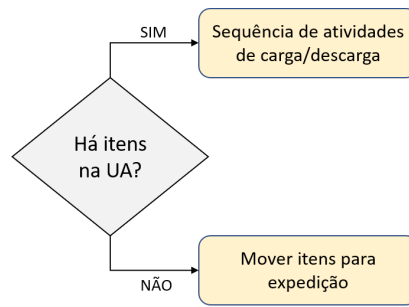


Figura 5.20: Funcionamento da atividade ‘*Decide: Items in Queue?*’

entre a UA e o *combiner* será realizado pelo robô (presente na Figura 5.18). De seguida é ligada uma atividade de carga (‘*Load*’) e uma de descarga (‘*Unload*’), a qual foi definida para descarregar os itens coletados na paleta existente no *combiner*. Por fim, uma atividade de conclusão da sequência de tarefas, ‘*Finish TS>Loading*’. A partir desta atividade é conectada uma seta que indica que após a conclusão deste fluxo secundário, é suposto voltar à atividade principal, ‘*Decide: Items in Queue?*’, que irá avaliar de novo a situação e voltar a tomar uma decisão.

- **NÃO:** se não existirem mais itens na UA, a decisão passa por avançar para a fase de expedição. Os itens encontram-se todos agrupados na paleta, suportada pelo *combiner* que os envia para o tapete transportador a caminho da expedição, passando entretanto pela máquina de embrulho (‘*Wrapper*’). É para isto criada uma atividade direcionada à movimentação dos itens para expedição, ‘*Move Object: To Outbound*’ e, logo de seguida, é libertado o *combiner* através de um *shared asset* ‘*Release: Combiner*’. Desta forma fica disponível para satisfazer outra ordem de despacho, caso ainda haja ordens para satisfazer. Se a resposta for positiva, o fluxo segue para uma atividade ‘*Assign Labels: Move to Next Order*’, que fará com que o processo seja reiniciado no bloco 1. Caso seja negativa, a simulação termina.

A decisão tomada é referente a um *token* e não a uma ordem. Em termo de exemplo, havendo três itens numa UA, cada um com um *token* correspondente no OPF, a decisão vai recair sobre o sequenciamento de atividades de carga e descarga. Resumindo, a decisão é tomada individualmente. Apenas quando a UA se encontrar vazia a decisão será a movimentação dos itens para expedição. Algo que merece uma especial atenção neste fluxo é o facto da atividade de decisão ‘*Decide: Items in Queue?*’ ser efectuada por item e não por ordem de despacho. É por esta propriedade que o erro de um item ser colocado numa ordem de despacho diferente não acontece na presente simulação.

De forma a melhorar a eficiência da presente simulação do sistema *crossdocking*, foi realizada uma adição importante relativamente ao que já foi descrito.

Como é visível na Figura 5.18, o modelo contém seis UA, três paletizadores (*combiners*) e apenas um robô (ou UMP) a realizar o transporte de itens entre estes dois recursos. Este facto aumenta a probabilidade do sistema se tornar mais lento e, possivelmente, criar um *bottleneck* nesta área. De forma a contornar este eventual problema, foram adicionados dois novos robôs, tendo exatamente o mesmo objetivo do já existente. Esta adição permite aumentar a rapidez de processos e de satisfação de ordens de despacho.

Para tal, foram realizadas alterações, tanto ao nível do modelo como do OPF. Relativamente ao modelo, a Figura 5.21 comprova o posicionamento de três robôs afetos à realização do transporte de itens. Nesta fase não se definiu qualquer regra de prioridade de ordens de despacho, tendo em conta que o considerado *default* associa-se à chegada de todos os itens de uma ordem a uma UA. Isto é, a primeira ordem a ser satisfeita será a que tiver os itens reunidos na UA primeiro.

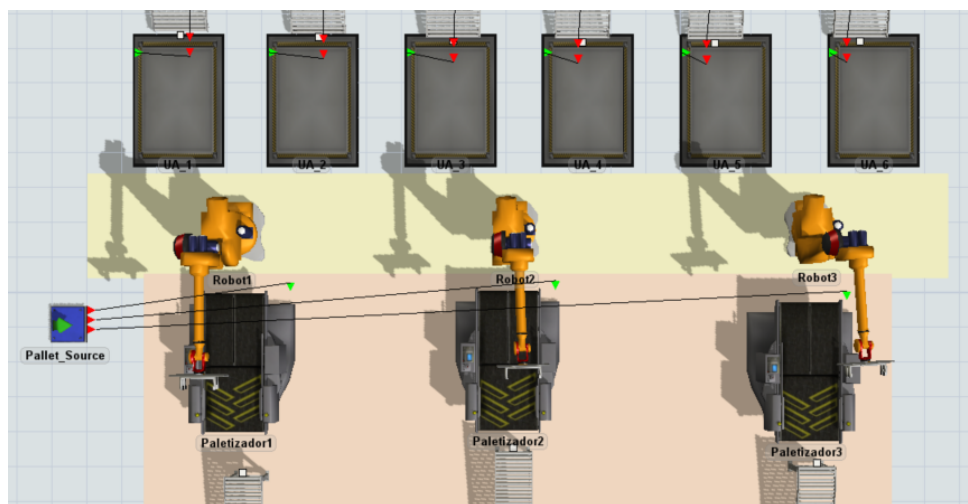


Figura 5.21: Aplicação de três robôs no transporte de itens no *crossdocking*

Quanto à lógica, embora semelhante à existente com apenas um robô, necessita da adição de *shared assets* no fluxo secundário do bloco 3 e da criação de um grupo exclusivamente para os robôs. Em termos de nomenclatura, foi criado o grupo '*Robots*', que contém os *Robot1*, *Robot2* *Robot3*. A criação deste grupo permite que, ao referenciar o mesmo, todos os robôs sejam utilizados como *Task Executors*. No OPF foi adicionado mais um recurso no bloco 3, '*Resource: Robots*', juntando-se assim ao recurso relativo aos *combiners* já existente. Antes da atividade de criação de uma sequência de tarefas, foi colocada a atividade '*Acquire: Robots*', o que permite que os robôs possam ser referenciados na atividade '*Create TS: Loading*' como *Task Executors*. Após a conclusão deste fluxo

secundário, é necessário a colocação de um ‘*Release: Robots*’ para que os mesmos sejam libertados e disponibilizados para uma nova execução de processos. A Figura 5.22 apresenta as alterações realizadas.

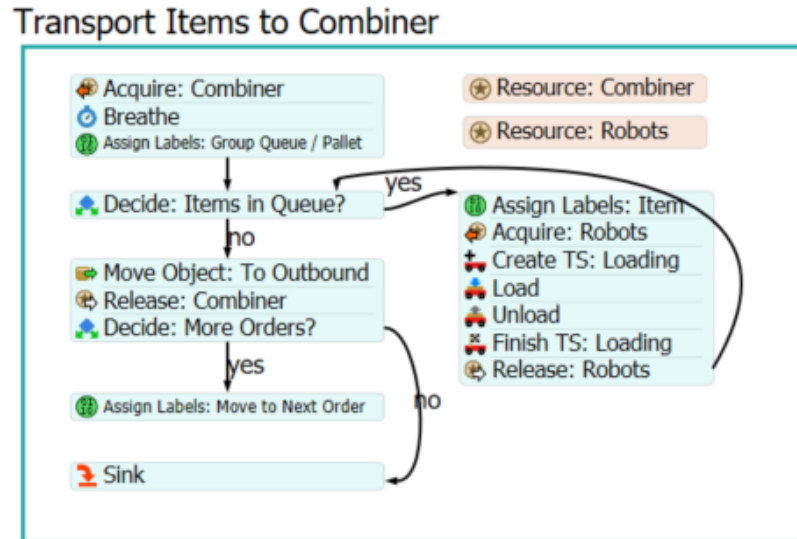


Figura 5.22: Alteração de fluxo no bloco 3 do OPF

Visto que o modelo contém seis UA, três robôs e três paletizadores, é possível dividir encargos. Ou seja, o *Robot1* encarrega-se apenas das UA_1 e UA_2, o *Robot2* das UA_3 e UA_4 e o *Robot3* das UA_5 e UA_6. Para que o funcionamento do sistema siga esta lógica, é necessário implementá-la. Para tal, foi necessário proceder à criação de um *label* partilhado por estes dois tipos de elementos (*Fixed Resources* e *Task Executors*), isto é, a todas as UA existentes foi atribuído um *label* nomeado de ‘*Number*’, assim como aos três robôs. O valor associado é que difere, tal como é demonstrado na Tabela 5.3.

Tabela 5.3: *Label* associado aos elementos UA e *Robot*

<i>Element</i>	<i>Label Value</i>
UA1 & UA2	1
UA3 & UA4	2
UA5 & UA6	3
<i>Robot1</i>	1
<i>Robot2</i>	2
<i>Robot3</i>	3

Para completar o funcionamento da lógica em implementação, foi necessário adicionar a seguinte *query* à atividade ‘*Acquire Robots*’:

WHERE Number = puller.Queue.Number

Desta maneira, assim que a atividade seja executada, a *query* força a que os robôs apenas realizem tarefas onde o seu *label value* é igual ao *label value* das UA existentes. Visto que, por exemplo, as UA_1 e UA_2 têm o mesmo *label value* que o *Robot1*, este só irá efetuar o transporte de itens existentes nas mesmas. Posto isto, o processo é repetido, utilizando o mesmo *label* e *query* mas desta vez aplicado aos três paletizadores, como é visível na Tabela 5.4.

Tabela 5.4: *Label* associado aos paletizadores

Element	Label Value
<i>Palletizer1</i>	1
<i>Palletizer2</i>	2
<i>Palletizer3</i>	3

Quanto ao bloco 3, foi necessário realizar uma pequena alteração, para que apenas se atribuisse o *palletizer* após o modelo saber de qual UA os itens são transportados. Para isso, dividiu-se a atividade ‘*Assign Labels: Group/Pallet*’ em duas atividades diferentes, sendo que a atividade ‘*Acquire: Combiner*’ passa a situar-se entre as mesmas, como demonstra a Figura 5.23.

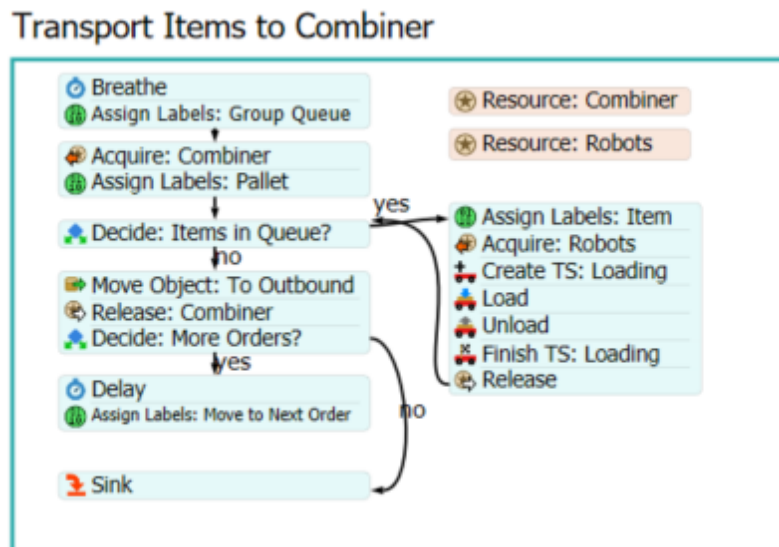


Figura 5.23: Alteração de atividades no bloco 3

Assim, o *Palletizer1* associa-se apenas ao *Robot1*, o *Palletizer2* ao *Robot2* e o *Palletizer3* ao *Robot3*, tal como está representado na Figura 5.24. O modelo torna-se mais completo com estas associações, permitindo que os robôs estejam destinados a determinadas UA. Para além disso, à medida que o robô está a efetuar o transporte de itens entre uma UA e o paletizador, a outra UA associada encontra-se a receber itens até completar a ordem de despacho. De salientar que,

utilizando a ferramenta do PF, não é necessário aplicar algumas ligações entre elementos, como por exemplo *S-Connections* entre as UA, os robôs e o paletizador.

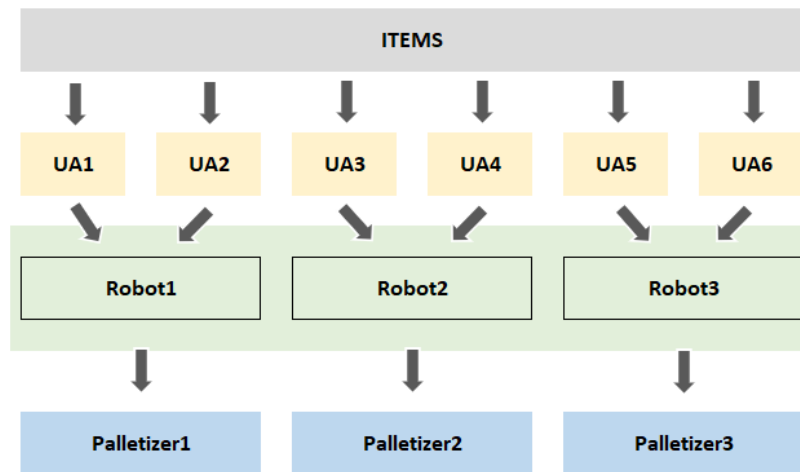


Figura 5.24: Associação entre elementos na concretização de ordens de despacho

Na Figura 5.23, é possível verificar uma atividade de atraso, ‘*Delay*’, imediatamente antes da atividade ‘*Assign Labels: Move to Next Order*’. Esta ‘*Delay*’ tem o intuito criar um compasso de espera entre a execução de ordens de despacho. Assim, quanto maior for o valor do atraso menor será o número de ordens de despacho satisfeitas. Esta atividade terá impacto, consoante o seu valor, na eficiência do sistema.

Tendo em conta todos estes processos, assim que a simulação é executada, no PF é possível acompanhar o estado de cada *token*, percebendo qual a atividade que está a desempenhar em determinado momento, e qual o seu estado. Caso a cor do *token* seja verde, o seu fluxo está normalizado e a decorrer sem qualquer problema; caso seja vermelha, indica que está parado em alguma atividade (possível espera para desempenhar o processo seguinte) ou que algum erro ocorreu no local onde se encontra. É uma funcionalidade extremamente importante pois permite detetar eventuais erros e problemas existentes no modelo.

Quanto à implementação da lógica e do modelo do sistema *crossdocking*, uma particularidade interessante e importante que foi aplicada no *Order Process Flow*, foi a utilização da propriedade *Process Flow Instances*, ou instâncias do fluxo de processos. Uma instância é uma ocorrência única de um fluxo secundário, fluxo de processo de recurso fixo ou fluxo secundário de executor de tarefa. Ao utilizar um fluxo de instâncias é possível anexar objetos (instâncias) ao fluxo de processo. No caso da presente simulação, as instâncias são as UA, uma vez que é nestas que as ordens de despacho se fixam. Esta propriedade traz diversas vantagens ao sistema de *crossdocking* pois permite que, caso necessário, algumas UA não sejam

utilizadas, sendo que a única restrição é que pelo menos uma UA tem que estar ativa ou anexada ao fluxo. A Figura 5.25 representa a janela de propriedades do OPF e os objetos que nele se encontram anexados (UA_1, UA_2, UA_3, UA_4, UA_5 e UA_6).

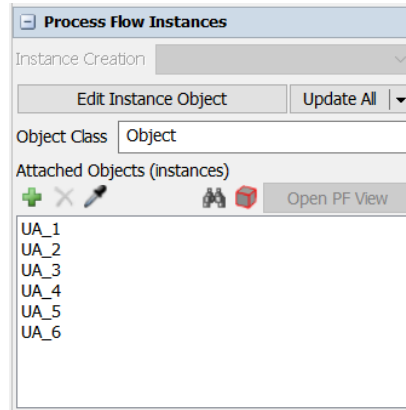


Figura 5.25: Objetos/instâncias utilizadas na lógica do OPF

É possível avaliar o desempenho do sistema tendo em conta o número de UA utilizadas. Caso seja necessário adicionar alguma UA para além das já existentes, para que esta entre no fluxo de processos, apenas deve ser conectada aos restantes elementos do modelo e anexada no OPF como um objeto.

Todos os procedimentos explícitos ao longo desta secção foram realizados de forma contínua e sempre com o foco num sistema eficiente. A maior dificuldade no desenvolvimento do sistema recaiu sobre de que forma providenciar uma melhor replicação das ideias para a lógica no fluxo de processos. Tendo em vista a próxima etapa da presente dissertação, a implementação de um AS/RS, o sistema *crossdocking* contém um *output* que simula a passagem de itens para o AS/RS. Será necessário também a adição de um *input* para a recuperação de itens armazenados.

Ao longo do desenvolvimento da implementação, foi necessário alterar as formas dos itens utilizados, apresentadas anteriormente na Figura 5.2. A nova forma dos mesmos está presente na Figura 5.26.

5.2 Implementação do AS/RS

Através do *software* FlexSim, a simulação de sistemas de armazenamento é facilitada e mais favorável a resultados credíveis devido aos elementos que o mesmo oferece. No que diz respeito a um AS/RS, o FlexSim, para além de oferecer diversos recursos relacionados com armazenamento (como prateleiras, elevadores, entre outros), possibilita o uso de *ASRS Vehicles*.

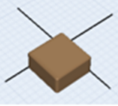
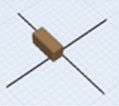
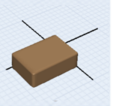
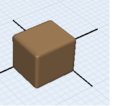
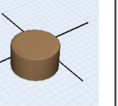
Source	Inbound_1	Inbound_2	Inbound_3	Inbound_4	Inbound_5
Item					
Cores	6	6	6	6	7

Figura 5.26: Alteração de itens utilizados na simulação

Começou por ser implementado no modelo o *layout* apresentado no Capítulo 4. Inicialmente foram criados *conveyors* de forma a ligar o local de transição de itens entre o *crossdocking* e o *ASRS*. De forma a melhorar e facilitar o armazenamento e recuperação de itens foi definido que os itens seriam recolhidos para armazenamento numa das pontas dos corredores e recuperados para inserção no fluxo de itens do *crossdocking* na ponta oposta. Na área destinada ao armazenamento de itens, os *conveyors* ligam-se a três *queues* distintas, nomeadas de *input1*, *input2*, *input3*. O mesmo acontece com as *queues* de recuperação de itens, *output1*, *output2*, *output3*, as quais estão ligadas ao *conveyor* de retorno ao *crossdocking*. Ou seja, os elementos utilizados foram:

- 6 *queues*: *input1*, *input2*, *input3*, *output1*, *output2*, *output3*;
- 4 *racks* ou prateleiras: *rack1*, *rack2*, *rack3*, *rack4*;
- 3 *ASRSVehicles*: *ASRSVehicle1*, *ASRSVehicle2*, *ASRSVehicle3*.

Tal como na implementação do *crossdocking*, foi necessário criar conexões entre os elementos para que a lógica do sistema seja apurada e bem definida. Assim sendo, foram efectuadas conexões do tipo *A-Connections* e *S-Connections*, entre os seguintes elementos:

- *A-Connections*: entre cada *conveyor* e cada *input* e entre cada *output* e cada *conveyor* de saída. Entre cada *input* e as duas *racks* correspondentes ao corredor onde se encontram. Entre cada *rack* e cada *output* no corredor onde se encontram.
- *S-Connections*: entre cada *input* e *ASRSVehicle*, *ASRSVehicle* e *rack* e entre cada *ASRSVehicle* e *output*.

Ao implementar um armazém, um dos principais fatores a ter em conta é o seu espaço útil, tal como indicado no Capítulo 4, onde também foram definidas

as dimensões das *racks* em utilização. Quanto à política de armazenamento a utilizar, de acordo com o estudado no Capítulo 2, várias possibilidades quanto a este tema são exequíveis. De forma a criar alguma organização no sistema, tanto o armazenamento aleatório, como o armazenamento no local livre mais próximo, foram descartados. Tendo em conta que na presente simulação estão a ser utilizados cinco tipos de itens diferentes, a possibilidade mais favorável baseia-se num armazenamento dedicado.

Recorrendo ao armazenamento dedicado, os itens são sempre armazenados num local específico, onde o reabastecimento dos mesmos é efectuado sempre no mesmo local. É aplicável ao caso pois cada corredor do armazém contém um *input*, facilitando o reabastecimento dos itens. Quanto ao local de armazenamento, através da ferramenta ‘*Paint Slot Labels*’, presente na biblioteca vocacionada para este tipo de sistemas, foi possível determinar um local de armazenamento para cada tipo de item. Tal como referido anteriormente, cada item contém um *label* associado. Visto que circulam pelos sistemas cinco tipos de itens diferentes, utilizam-se então cinco tipos de *labels* diferentes. Cada *label* será armazenado num local específico de cada *rack* e terá uma cor associada, como está definido na Tabela 5.5.

Tabela 5.5: Cor correspondente a cada *label* no armazenamento dedicado

Label	1	2	3	4	5
Cor	Vermelho	Verde	Azul	Amarelo	Laranja

De forma a que o sistema seja equilibrado, e visto que em cada *rack* os itens de um determinado *label* podem ser armazenados em duas baías diferentes (cinco itens para dez baías), foram distribuídas cores em cada *rack* equidistantes quer dos *inputs* como dos *outputs*. Desta forma atenua-se o facto do *ASRSVehicle* demorar mais tempo a percorrer uma determinada distância para armazenar um item de um determinado *label* do que outro. A Figura 5.27 representa a distribuição de locais de armazenamento, conforme a cor, de cada *rack*.

Como os itens não são todos semelhantes, diferem pelas dimensões e pela sua forma, a capacidade de armazenamento de cada um é variável. Caso seja necessário alterar (aumentar ou diminuir) as capacidades de armazenamento de cada item, basta utilizar a ferramenta ‘*Paint Slot Labels*’ e adequar ao pretendido. Por cada baía, é possível armazenar as seguintes quantidades:

- item tipo 1: 16 unidades;
- item tipo 2: 16 unidades;
- item tipo 3: 12 unidades;

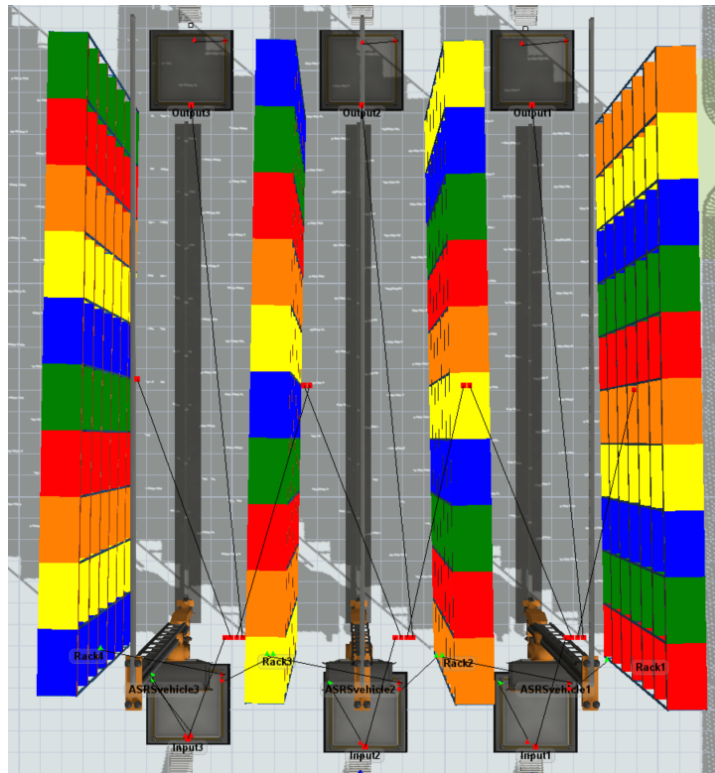


Figura 5.27: Distribuição de locais de armazenamento por *rack*

- item tipo 4: 32 unidades;
- item tipo 5: 196 unidades.

De forma a que os itens seguissem a política de armazenamento implementada ao entrarem no sistema AS/RS, foi necessário definir, para cada *rack* em utilização, a estratégia de atribuição de *slots* ou, tal como é nomeado no *software*, ‘*Slot Assignment Strategy*’. Foi então utilizada a estratégia ‘*Matching Labels*’, que indica ao *Task Executer*, que neste caso é o *ASRSVehicle*, que os itens de um determinado *label* só podem ser armazenados no local destinado para o mesmo.

Quanto às propriedades pertencentes ao *ASRSVehicle*, é de salientar que a sua capacidade é de doze itens por cada viagem. Ou seja, no momento que estejam itens presentes no *input*, o *ASRSVehicle* pode recolher doze de uma só vez e armazená-los de seguida no local destinado a cada um. Para o caso, não foi definida nenhuma regra de prioridade de armazenamento, pois não se considera necessário, sendo a distribuição de itens feita por ordem de recolha.

Com o objetivo de facilitar a visualização do armazenamento de itens, a Figura 5.28 é um bom exemplo para perceber a sua disposição. Estão na mesma representados os cinco tipos de itens existentes na simulação. Os itens não são

armazenados numa palete ou recipiente próprio, são colocados diretamente na *slot* de armazenamento. Em cada prateleira a quantidade de itens armazenados varia consoante o espaço disponível. Isto é, em termos de área, as prateleiras contêm o mesmo espaço. Porém, devido à diferente dimensão de cada tipo de itens, o número dos mesmos que será armazenada no mesmo espaço difere. Assim sendo, num espaço ou *slot* existente, serão armazenados o número máximo de itens possível.

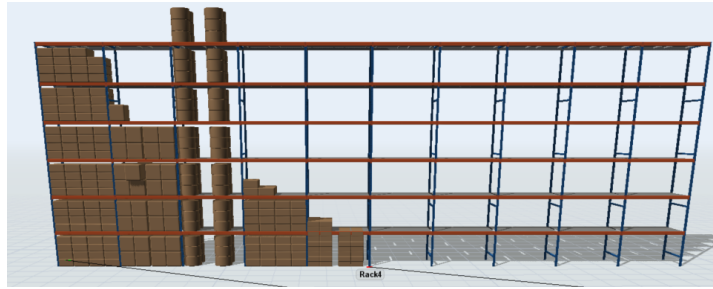


Figura 5.28: Distribuição de itens armazenados por *label* numa *rack*

Quanto à aplicação de *racks*, *ASRSVehicles*, *inputs* e *outputs*, nenhuma ação é acrescentada até ao momento. A próxima fase não é relativa à implementação de elementos mas sim ao funcionamento do sistema. Tal como no sistema *crossdocking*, é necessário utilizar lógica através do PF e de DP. Quanto à utilização do PF será abordada na secção seguinte devido à necessidade de partilha de alguns *shared assets* e listas de ordens de despacho. Relativamente aos DP, foram utilizados na fase de recolha de itens para armazenamento, para facilitar a distribuição dos mesmos para os locais de *input*.

Os itens que não se encontram associados a uma ordem de despacho no fluxo do *crossdocking* seguem para armazenamento e são recolhidos nos *inputs* do sistema AS/RS. No momento antes de serem distribuídos pelos três *inputs* existentes, foi colocado um DP com o intuito de definir qual a distribuição mais adequada dos mesmos para cada entrada, tal como está esquematizado na Figura 5.29.

O DP que toma a decisão é o DP12, como é possível identificar na Figura 5.29. De modo a criar uma distribuição equilibrada pelos três *inputs*, foi criado um *trigger* ‘*On Arrival*’ no DP12. Tem como objetivo enviar o item que o atravessa (‘*Send Item*’) para um destino específico. Esse destino é atribuído de acordo com uma funcionalidade do *software* intitulada de ‘*Round Robin Output Port*’. Esta técnica é incremental, ou seja, de três itens que atravessarem o DP12 de forma contínua, o primeiro é enviado para o primeiro objeto disponível, o segundo para o segundo objeto e terceiro para o terceiro objeto. Como só há três objetos, o item que atravessar o DP12 a seguir será reencaminhado para o primeiro objeto e assim se repete o processo, em forma de *loop*. Considera-se que

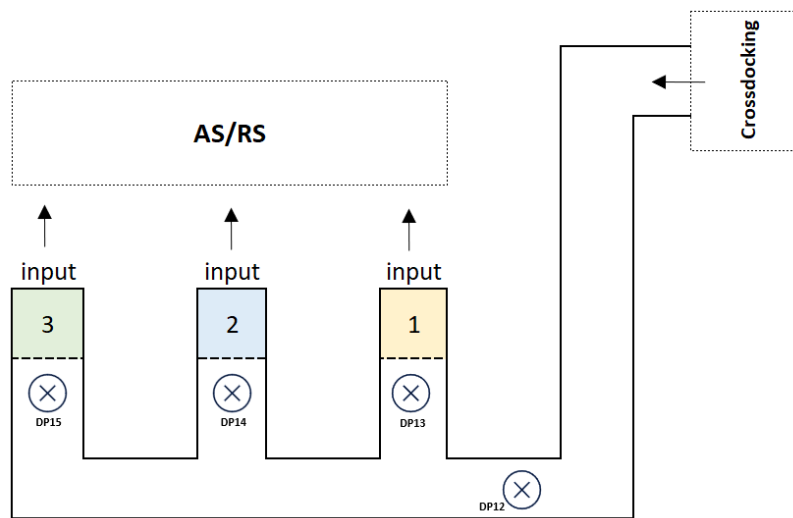


Figura 5.29: Colocação estratégica de *decision points* na distribuição de itens para armazenamento

o primeiro objeto é o DP13, o segundo objeto o DP14 e o terceiro objeto o DP15. Estes três *decision points* estão conectados ao DP decisivo que é o DP12.

A recuperação de itens será feita através dos *outputs* 1, 2, 3, porém, devido à complexidade associada a este processo, será realizada através da lógica no PF.

A implementação de um sistema AS/RS revelou-se mais simples do que a implementação do sistema *crossdocking*, em parte devido a uma necessidade inferior de elementos e de ligações e também devido à vantagem existente na utilização de *racks* que facilita a atribuição de políticas de armazenamento. Na secção seguinte será abordada a ponte de ligação entre o sistema *crossdocking* e o sistema AS/RS, sendo que esta se foca na recuperação de itens armazenados e no envio dos mesmos para o fluxo. Ou seja, é possível satisfazerem as ordens de despacho existentes que, no momento em que é requisitado um item do armazém, se encontram em espera.

5.3 Integração do sistema *crossdocking* com o AS/RS

5.3.1 Enquadramento da integração entre sistemas

De maneira a que haja uma ponte de ligação entre os dois sistemas simulados, a integração dos mesmos é feita através da transição de itens de um sistema para o outro, mas maioritariamente na transição de itens entre o AS/RS e o *crossdocking*. Esta secção funciona como uma reformulação do OPF desenvolvido na Secção 5.1, diretamente relacionado com o que foi implementado na Secção 5.2. O objetivo passa por atualizar e adequar os processos já definidos à integração entre os

sistemas de *crossdocking* e AS/RS, adicionando também atividades fulcrais para o seu bom funcionamento.

Um fator importante passa pela entrada de itens através das *sources* do *crossdocking*, que têm como objetivo simular a descarga de produtos de veículos. Tal como em contexto real, a descarga não é constante, sendo necessário ajustar a simulação a este facto. Ao longo da implementação não se revelou um fator importante porém, nesta fase, convém que os tempos sejam pré-definidos de forma a melhorar o fluxo de itens ao longo da simulação, tornando-a mais acessível. Duas hipóteses podem ser aplicadas, sendo que a primeira passa por atrasar a entrada de itens no sistema (por exemplo, cada *source* enviar um item para o sistema de trinta em trinta segundos), e a segunda por criar itens de acordo com uma distribuição estatística. Para a presente simulação, no cenário base de funcionamento, foi utilizada a primeira hipótese, definindo, aleatoriamente, os seguintes tempos de lançamento de itens para o sistema, através do campo ‘*Inter-Arrivaltime*’ existente em cada *source*, os quais estão representados na Tabela 5.6.

Tabela 5.6: Tabela de tempos de chegada de itens ao sistema

Source	Time (s)
<i>Inbound_1</i>	25
<i>Inbound_2</i>	35
<i>Inbound_3</i>	38
<i>Inbound_4</i>	28
<i>Inbound_5</i>	33

Previsivelmente, ao desenvolver a integração entre os dois sistemas, foi necessário proceder a diversas alterações. No que diz respeito ao modelo de simulação, o que foi apresentado na secção anterior mantém-se inalterado. Já o OPF correspondente ao *crossdocking* sofreu algumas alterações. A partir desta etapa, o OPF não diz respeito apenas ao sistema *crossdocking* mas também ao AS/RS, visto que uma ordem de despacho é dependente de ambos os sistemas e, anteriormente, apenas o *crossdocking* teria influência no OPF.

Relembrando o que foi referido na secção anterior, o OPF era constituído por três blocos, cada um destinado à execução de um processo. Para possibilitar a recuperação de itens armazenados no AS/RS, procedeu-se à criação de dois novos blocos, realizaram-se alterações num deles e, para além disso, acrescentaram-se diversas atividades. Foram acrescentados:

- *New Items to Queue*: bloco destinado às atividades dos itens provenientes das *sources*.
- *Stored Items to Queue*: bloco destinado às atividades dos itens já existentes, que são necessários para satisfazer uma ordem de despacho, e que se

encontram armazenados no armazém.

5.3.2 Aplicação para a integração entre sistemas

Visto que já estava em utilização uma lista para guardar os itens oriundos das *sources* do *crossdocking*, não foi necessário criar uma nova referente aos itens armazenados no AS/RS. Foi utilizada a mesma lista (*List: Items Entering Conveyor*) e acrescentado um *label* na mesma que ordena os itens por prioridade. Assim, os itens provenientes das *sources* têm prioridade 1 e os itens armazenados no AS/RS prioridade 0. No modelo, os itens com prioridade 0 são os que percorrem o caminho que passa pelo DP9 (localização na Figura 5.30), localizado no tapete que encaminha os itens exclusivamente para o AS/RS. Os itens retirados do AS/RS e enviados para o CS destinados a uma ordem de despacho, para que seja possível distingui-los, a sua cor assim que saem do armazém automático passa a azul.

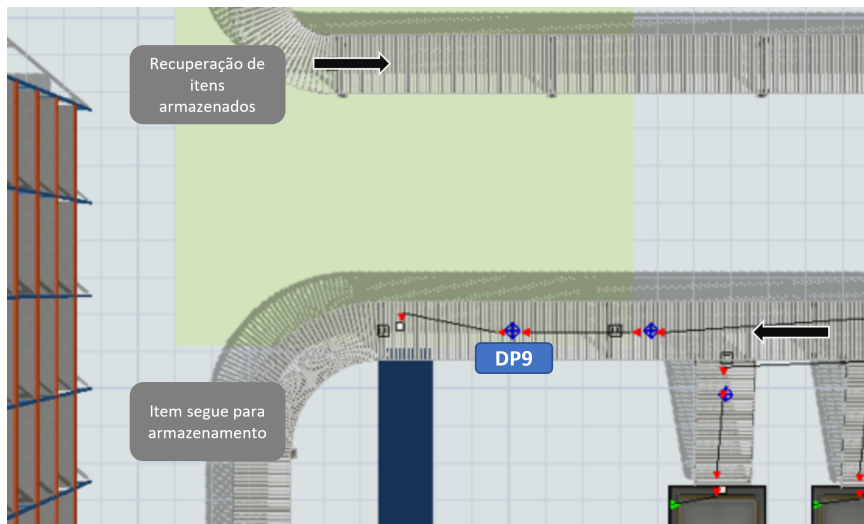


Figura 5.30: Localização estratégica do DP9

Para definir estas prioridades foram criados dois *labels* inexistentes até ao momento:

- *Label 'Priority 1'*: criado um *trigger* em cada *source* que, no momento que fornece um item ao sistema, é atribuído o *label priority 1* ao mesmo, através da utilização de um *trigger 'Set Label', 'On Creation'*.
- *Label 'Priority 0'*: assim que um item passar pelo DP9 (localizado no *conveyor* que segue exclusivamente para o AS/RS), é atribuído o *label priority 0* através da utilização de um *trigger 'Set Label', 'On Arrival'*.

Desta forma, todos os itens se encontram identificados com um *label* de prioridade e não foi necessário alterar nenhuma lógica de extração (*pulling logic*) a não ser a adição da *query* de ordenação por prioridade na lista, representada na Figura 5.31.

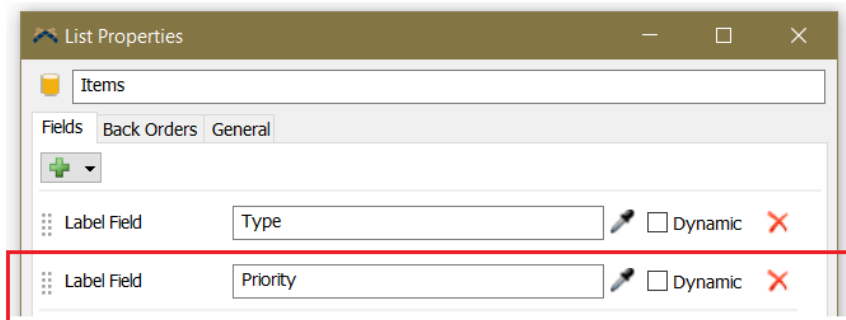


Figura 5.31: Criação de um *label* para definição de prioridade de itens na lista dos mesmos

As UA apenas irão ‘chamar’ um item do AS/RS caso não haja itens deste tipo fornecidos por nenhuma *source*.

No PF generalizado foi necessário acrescentar um conjunto de atividades para que os itens armazenados se mantenham na lista em utilização. A Figura 5.32 apresenta a ligação existente entre a lista ‘*List: Items Entering Conveyor*’ e os conjuntos de atividades já existentes e as criadas correspondentes ao armazenamento.

De uma forma geral, o funcionamento da lista engloba três fases:

- fase 1: os itens, ao entrarem no sistema através das cinco *sources*, são introduzidos na lista;
- fase 2: assim que ultrapassem o DP1 (localizado imediatamente antes da distribuição para as UA), são retirados da lista pois pressupõe-se que serão utilizados para satisfazer uma ordem de despacho;
- fase 3: caso não sejam enviados para uma UA, continuam o seu percurso até atingirem o DP9 e voltam a ser inseridos na mesma lista, embora sigam para armazenamento.

O PF generalizado existe simplesmente para que os itens sejam introduzidos ou retirados da lista, de forma a mantê-la atualizada no decorrer do OPF.

Posto isto, a próxima etapa passa por reformular os blocos criados anteriormente, como o bloco 2, nomeado ‘*Get Ordered Items Off List/Place in Queue*’.

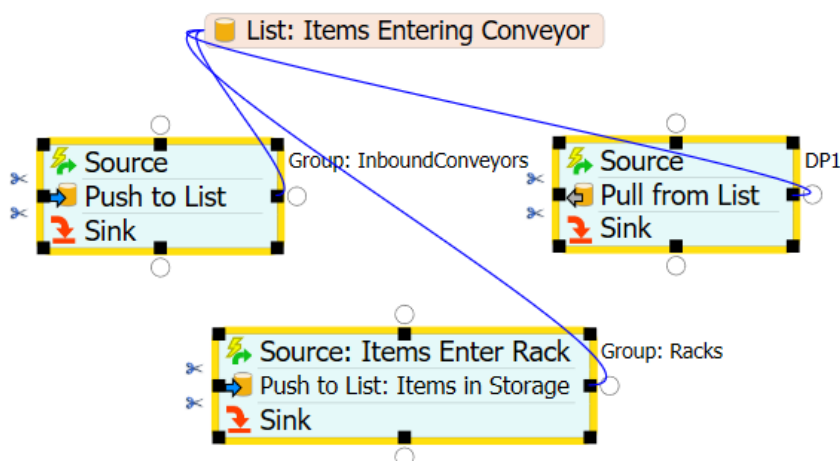


Figura 5.32: Ligações existentes na lógica do *Process Flow* generalizado

Como através deste desenvolvimento foi introduzido o facto de itens em armazenamento serem capazes de reentrar no *crossdocking*, este bloco adquiriu a função de decidir qual item utilizar para satisfazer a ordem de despacho. Assim, foi colocada uma atividade ‘*Pull from List: Items*’ correlacionada com uma atividade de decisão *Decide*. Isto permite que o sistema tenha a “inteligência” de verificar os itens existentes na lista, e decidir se vão ser utilizados itens que se encontrem em circulação no sistema ou se irá necessitar dos que se encontram armazenados. Para ordenar os itens por prioridade, a atividade ‘*Pull from List: Items*’ contém um campo especializado para *queries*, que foi preenchido com o comando ‘ORDER BY Priority DESC’, de forma a utilizar os itens ordenados por prioridade, de forma decrescente.

A atividade de decisão gera dois novos blocos, um para cada opção. Relativamente aos itens que já se encontram presentes no sistema (oriundos das *sources*) foi criado o bloco *New Items to Queue* e para os que se encontram em armazenamento o bloco *Stored Items to Queue*.

Em termos práticos, o bloco 2 juntamente com o novo bloco destinado aos itens já presentes no sistema é uma cópia do bloco 2 antes de sofrer uma reformulação. Isto porque foi necessário dividir o bloco em dois devido à adição da nova funcionalidade. No momento da decisão, caso os itens requeridos estejam em circulação no sistema, o *token* segue para o bloco *New Items to Queue*, onde será executado um *match* e enviado para a UA correspondente à ordem de despacho onde é necessário.

Quanto ao bloco *Stored Items to Queue*, foi criada uma sequência de tarefas (*Task Sequence*) para que seja possível recuperar o item do armazém e transportá-lo até ao *conveyor system*. As atividades utilizadas estão representadas na Figura

5.33.

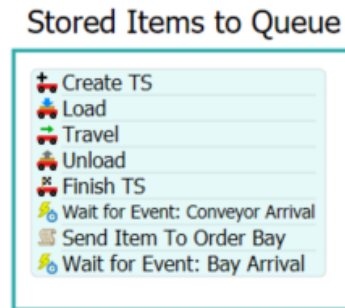


Figura 5.33: Bloco do OPF de recuperação de itens armazenados no AS/RS

Para a sequência de tarefas presente na Figura 5.33, foram utilizadas atividades *Load*, *Travel* e *Unload*. A atividade *Create TS* é responsável por criar esta sequência (sendo que o *Task Executer* é o *ASRSVehicle*). Ou seja, o item é retirado das prateleiras pelo *ASRSVehicle* (*Load*), transportado para o *output* do armazém (*Travel*) e finalmente descarregado no mesmo *output* (*Unload*). Aqui termina a sequência de tarefas e o item entra num *conveyor system* secundário responsável por transportá-lo até ao *conveyor system* que dá acesso à zona de distribuição de itens pelas UA. Assim que o item atinge o DP1, os processos são uma cópia dos executados no bloco *New Items to Queue*. No momento em que o item é recuperado do AS/RS e se dirige para uma UA, foi adicionado um *trigger* no meio do seu percurso com o objetivo de mudar o item de cor. Isto possibilita visualizar quais os itens de uma ordem de despacho que foram recuperados do armazém e distingui-los dos que foram fornecidos diretamente por uma das *sources*.

Após a execução de um destes blocos, a lógica de ambos converge para uma atividade destinada à ordem de despacho que se encontra a ser satisfeita, nomeada '*Batch: Wait for All to Arrive*'. Esta atividade fazia antes parte do bloco 2 porém agora encontra-se isolada no OPF. Tem como objetivo criar um período de espera para que os itens de uma ordem só sejam transportados para um *combiner* quando estiverem todos reunidos na UA em causa. A Figura 5.34 representa a lógica existente entre os blocos descritos.

Neste momento da implementação, o sistema *crossdocking* encontra-se integrado com o sistema AS/RS, sendo possível recorrer à utilização de itens armazenados para satisfazer uma ordem de despacho a ser cumprida no *crossdocking*.

Ao executar a simulação do modelo, aparece no ecrã do *software* uma janela *pop-up* com o objetivo de visualizar o comportamento da lista de itens existentes no sistema. Os itens que entram na lista (*Items Entries*) e os que estão associados a pedidos em espera (*Items Back Orders*) são exibidos através de uma tabela

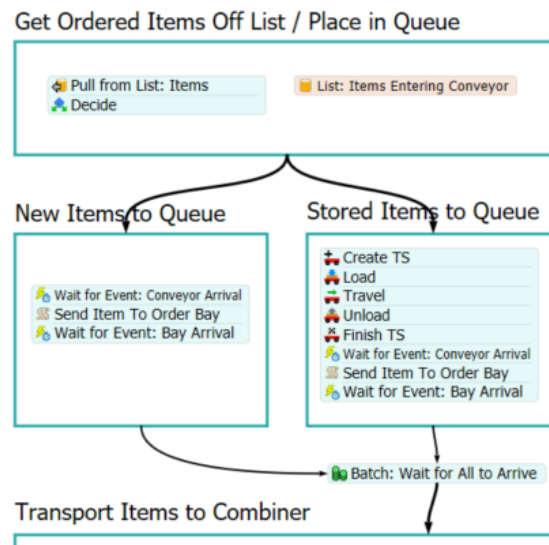


Figura 5.34: Reestruturação do OPF para integração do AS/RS com o *crossdocking*

dinâmica que atualiza o seu conteúdo em tempo real, à medida que a simulação evolui. A Figura 5.35, captada num momento aleatório da simulação, é um exemplo deste tipo de tabela.

Items Entries					
value	Type	Priority	age	distance	queueSize
Partition ID: 2					
Partition ID: 4					
/Conveyor1/Item3~2.4	1	1	32.67	Puller Require 8	
/Conveyor1/Item3~3.4	1	1	8.67	Puller Require 8	
Partition ID: 5					
/Conveyor1/Cylinder~3.5	1	1	16.67	Puller Require 8	
Partition ID: 3					
/Conveyor22/Item2.3	1	1	0.67	Puller Require 1	

Items Back Orders					
puller	query	requested	required	fulfilled	
Partition ID: 2					
id:116 instance: UA_4 ORDER BY Priority DESC		1	1	0	
id:129 instance: UA_5 ORDER BY Priority DESC		1	1	0	
id:140 instance: UA_6 ORDER BY Priority DESC		1	1	0	
Partition ID: 1					
id:128 instance: UA_5 ORDER BY Priority DESC		1	1	0	
id:138 instance: UA_6 ORDER BY Priority DESC		1	1	0	
id:139 instance: UA_6 ORDER BY Priority DESC		1	1	0	

Figura 5.35: Janela *pop-up* de acompanhamento da lista de itens

5.4 Conclusão do Capítulo 5

O presente capítulo descreve de forma sucinta toda a implementação, quer do sistema *crossdocking*, como do AS/RS. Foram descritos os passos e etapas reali-

zadas considerados de maior relevância. Na implementação do *layout* inicial do *crossdocking* foram descritos os passos embrionários desta simulação. A colocação de todos os elementos e recursos, a interligação e conexões existentes entre ambos e a utilização de várias estratégias para distribuição de itens por cor e por UA, que se revelaram fundamentais para uma melhor percepção do considerado necessário para o sistema funcionar. De seguida, foram estruturados e desenvolvidos dois tipos de *process flows*. Um generalizado, de forma a que a lista utilizada se mantenha atualizada de acordo com o fluxo de itens, e um *Order Process Flow* encarregado de gerir a lógica relativa às ordens de despacho e à forma como as mesmas eram cumpridas. Tendo um sistema de *crossdocking* funcional, o próximo passo respeita à criação de um armazém automático capaz de armazenar itens não utilizados no sistema adjacente. Foi explorada a implementação deste AS/RS no modelo. Por fim, tendo todos estes fatores alinhados, conclui-se o capítulo com a integração dos dois sistemas, a qual se baseia na lógica presente no OPF. Foram realizadas várias alterações e adições de forma a promover um bom funcionamento entre ambos os sistemas.

Uma alteração feita após todos os passos descritos até ao momento, foi a remodelação da tabela de ordens de despacho. Esta foi transposta de forma a facilitar a visualização das ordens existentes.

De salientar que, caso haja uma adição de elementos, tal como *sources* (representa uma doca de entrada de produtos no sistema) ou de *sinks* (representa a saída de produtos do sistema, para expedição), o modelo é adaptável, sendo apenas necessário, para além da adição dos elementos no modelo, adicionar os mesmos ao *process flow*, sem que seja preciso reestruturá-lo.

De forma a promover uma visão geral do modelo e do OPF, estes encontram-se representados, respetivamente, através da Figura 5.36 e Figura 5.37.

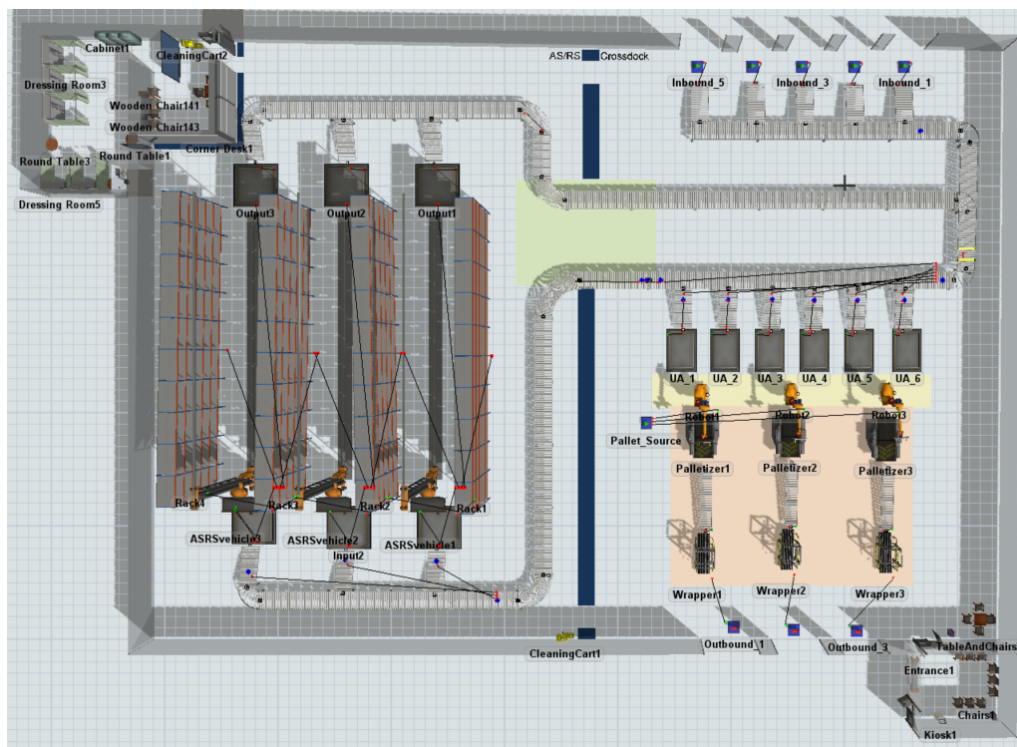
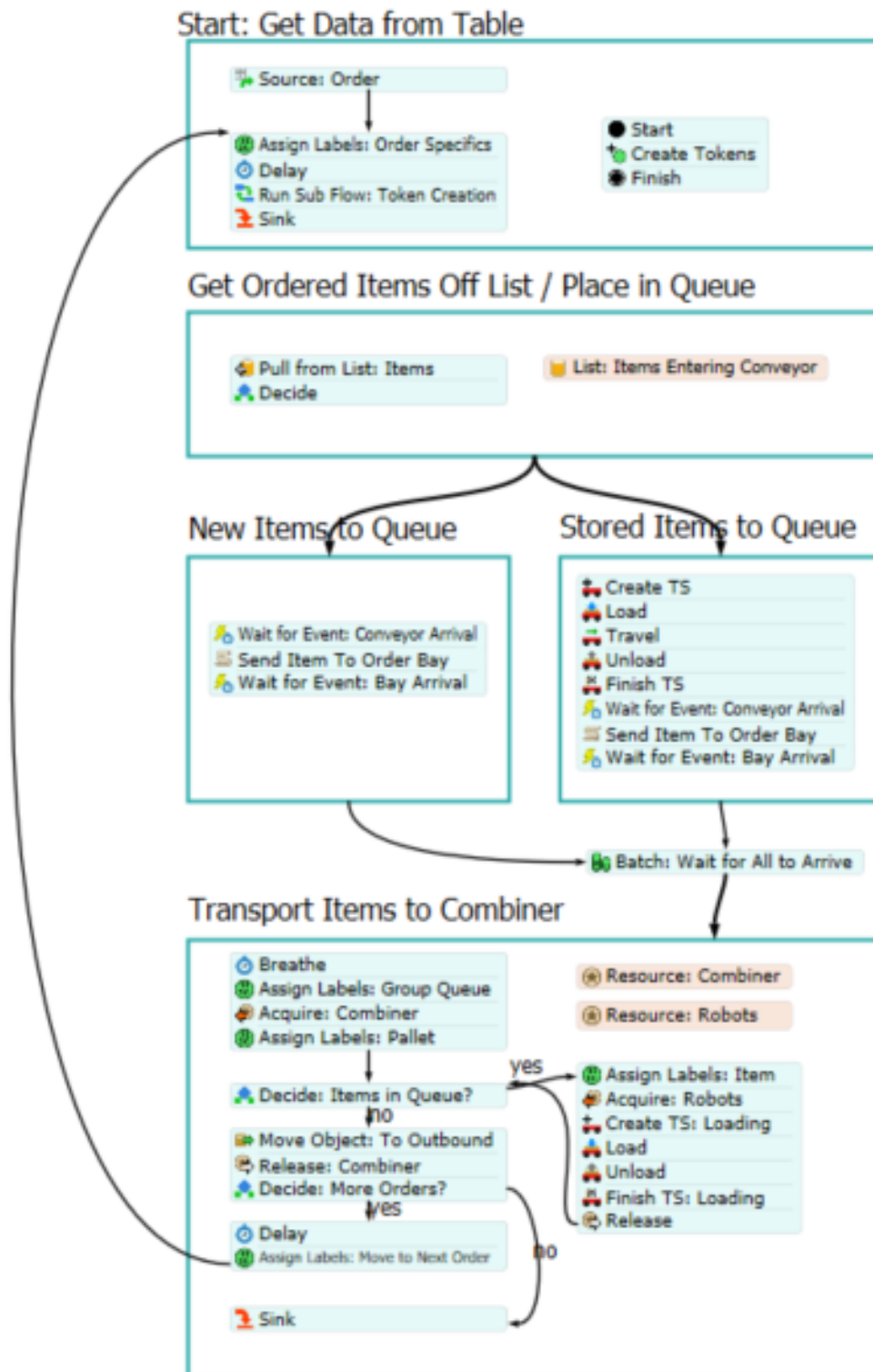


Figura 5.36: Visão geral do modelo da simulação

Figura 5.37: Visão geral do *Order Process Flow*

Capítulo 6

Cenários de Atividade e Resultados Obtidos

O presente capítulo tem como objetivo apresentar algumas variações na atividade, tanto do sistema *crossdocking* como do sistema AS/RS. Estas variações representam possíveis cenários. Com isto, é possível testar a eficiência do modelo e se o mesmo se adequa à realidade, tendo em conta o número de recursos e elementos em utilização, como por exemplo o número de unidades acumuladoras ativas, ou qualquer outro tipo recurso. Estas variações terão diferentes impactos no modelo, podendo afetar em grande escala a sua eficiência ou nem sequer afetar a mesma. Os cenários estudados são os considerados mais relevantes, visto que como existem diversos recursos e elementos na simulação, seria impossível testar o modelo criando variações em todos.

O *software* FlexSim possui uma ferramenta, ‘*Dashboards*’, capaz de analisar o modelo e gerar gráficos de acordo com os parâmetros que se pretendem avaliar, facilitando assim uma análise tanto global como pormenorizada ao modelo.

6.1 Cenários de atividade

Ao iniciar a criação de diferentes cenários do modelo, é de salientar um fator importante no que diz respeito à simulação. De forma a obter resultados credíveis, é necessário estabelecer um limite de itens que compõe uma ordem, de modo a que o número de itens por ordem não seja exagerado. Assim, foi definido que as ordens de despacho poderão ter um número variável de itens mas não superior a dez. Sendo a variável x o número de itens por ordem de despacho, então $x \leq 10$.

A Figura 6.1 é um exemplo da tabela de ordens. Este exemplo contém vinte ordens de despacho, sendo construída em Microsoft Excel e importada posterior-

mente para o FlexSim. A tabela existente no FlexSim contém cerca de trezentas ordens de despacho, geradas aleatoriamente e cumprindo a restrição da soma total de itens não poder exceder as dez unidades. Anteriormente as ordens estavam representadas por colunas. Para facilitar a distribuição de ordens em termos visuais, foi redefinida a tabela em utilização, passando as ordens a estarem expressas nas linhas da mesma.

	Type1	Type2	Type3	Type4	Type5	x
Order1	2	2	1	3	1	9
Order2	3	4	1	1	1	10
Order3	1	0	1	1	2	5
Order4	1	4	4	0	1	10
Order5	4	0	5	0	1	10
Order6	2	1	3	2	0	8
Order7	0	1	1	4	2	8
Order8	2	2	2	0	0	6
Order9	1	3	2	1	3	10
Order10	0	1	3	3	1	8
Order11	0	3	2	1	4	10
Order12	1	2	2	0	1	6
Order13	1	3	2	1	1	8
Order14	2	1	2	4	0	9
Order15	1	5	1	2	0	9
Order16	1	2	1	0	3	7
Order17	3	1	2	1	3	10
Order18	2	0	1	4	1	8
Order19	2	0	0	5	0	7
Order20	1	0	1	0	3	5

Figura 6.1: Exemplo de tabela de ordens utilizada na simulação

Para avaliar os processos executados durante a simulação, visto que a mesma não é considerada infinita, foi necessário estabelecer um intervalo de tempo fixo. Ou seja, os resultados obtidos para cada cenário de atividade têm como base um similar intervalo de tempo de simulação. Assim, foi estabelecido um intervalo de tempo de 3600 segundos, equivalente a 1 hora. Quanto à expedição de ordens de despacho, o número de *outbounds* (representados por *sinks*, podendo ser interpretados como cais de expedição), mantém-se o mesmo em todos os cenários (três *outbounds*).

Quanto aos cenários a avaliar no presente capítulo, devido à existência de imensas possibilidades diferentes, foram definidos os considerados mais relevantes no funcionamento do modelo e na sua eficiência. Visto que estão em causa dois sistemas integrados num só, na criação de cenários, é necessário criar também uma envolvimento entre ambos. Assim sendo, os cenários diferenciam-se maioritariamente devido a:

- unidades acumuladoras;
- ‘Delay’ aplicado entre ordens de despacho;

De modo a medir a eficiência do sistema, uma das hipóteses recai sobre o tempo de funcionamento dos robôs. Ou seja, a percentagem de tempo em que os mesmos se encontram a desempenhar funções de transporte de itens entre uma UA e um paletizador. Estas funções estão diretamente relacionadas com a quantidade de ordens de despacho satisfeitas num determinado período de tempo. É possível medir a percentagem de tempo em que se encontram em *stand by*, normalmente referido como *idle speed*, em que se encontram a transportar um item (*offset travel loaded*) e em que se encontram no processo de largar um item no paletizador e recolher outro na UA em utilização (*offset travel empty*). É também necessário ter em conta as características dos robôs, transversais aos três existentes:

- capacidade: 1 item por viagem UA-Paletizador;
- velocidade máxima: 3 m/s ;
- aceleração: 1 m/s^2 .

Em cada cenário, de forma a avaliar vários pontos do sistema, serão gerados vários gráficos utilizados para avaliação dos processos tendo em conta algumas restrições. No que diz respeito à análise do AS/RS, é de salientar que a *Rack1* do sistema não aparece nas representações gráficas pois esta é considerada uma prateleira de recurso, que apenas é utilizada caso as restantes se encontrem totalmente ocupadas. O título dos gráficos a gerar é o seguinte:

- número de ordens de despacho expedidas - '*Output per Expedition Point*';
- taxa de utilização dos robôs - '*Robot Utilization*';
- taxa de utilização das UA - '*UA Utilization*';
- número de itens armazenados no AS/RS - '*Throughput by Type - Storage*';
- número de itens recuperados no AS/RS - '*Throughput by Type - Retrieval*';
- conteúdo existente por *rack* ao longo do tempo - '*Content Vs Time*'.
- tempo médio de permanência de um item numa *rack* - '*Staytime per Rack*';

Foram gerados quatro cenários, com algumas variações em alguns dos seus parâmetros, como mostra a Tabela 6.1.

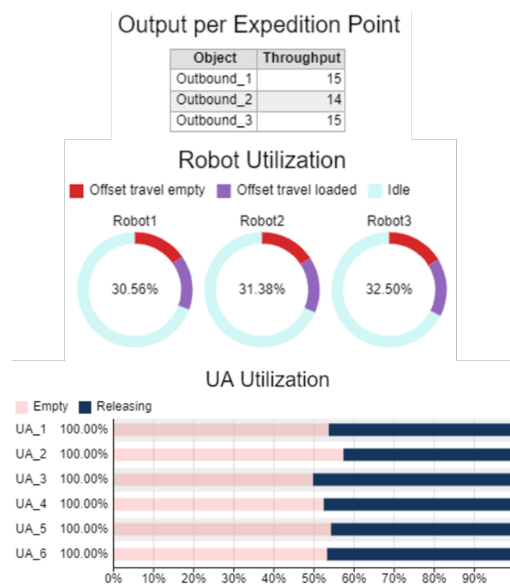
Tabela 6.1: Quatro cenários distintos simulados

Cenário	Nº de UA	Delay (s)
Cenário 1	6	200
Cenário 2	6	0
Cenário 3	3	200
Cenário 4	3	0

6.1.1 Cenário 1: 6 UA com delay entre ordens

No Cenário 1 não ocorre nenhuma alteração no modelo até ao momento. Este cenário tem o intuito de avaliar o mesmo sem nenhuma alteração ser executada, servindo como termo de comparação com os cenários seguintes.

Realizando a simulação do modelo, ao longo de 3600 segundos, foram gerados resultados que serão demonstrados através de gráficos fornecidos pela ferramenta ‘Dashboards’ do FlexSim. Os resultados mais direcionados com o sistema *crossdocking* estão representados na Figura 6.2 e com o AS/RS na Figura 6.3.

Figura 6.2: Resultados do Cenário 1 relativos ao sistema *crossdocking*

A Figura 6.2 apresenta os resultados obtidos, mais direcionados para o desempenho do sistema *crossdocking*. Relativamente às ordens de despacho (‘*Output per Expedition Point*’) e sabendo que o modelo contém três cais de expedição, no total foram expedidas 44 ordens de despacho. As saídas *Outbound_1* e *Outbound_3* expediram 15 ordens cada e a saída *Outbound_2*, 14.

Quanto ao gráfico ‘*Robot Utilization*’, o seu tempo de utilização ronda os 30%, os quais se encontram divididos entre o tempo que o mesmo transporta um item

entre uma UA e o paletizador e o tempo de viagem do paletizador de volta para a UA. No restante tempo os robôs encontram-se em *‘Idle Speed’*, ou seja, em repouso, à espera que todos os itens de uma ordem de despacho dêem entrada numa das UA à qual estão destinados. A diferença de utilização entre robôs é mínima e acontece devido à atribuição de ordens de despacho às UA associadas a cada robô. O tempo médio de utilização neste cenário é de 31,48%.

A utilização das UA (*‘UA Utilization’*), visto que se encontram todas em funcionamento, é, em média, 47%. Este valor deve-se maioritariamente ao facto de ter um *delay* elevado associado entre ordens de despacho.

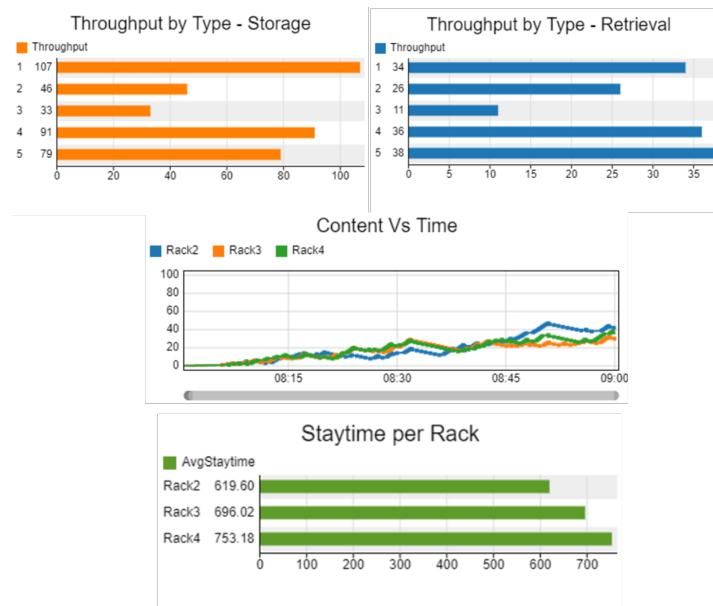


Figura 6.3: Resultados do Cenário 1 relativos ao AS/RS

Quanto aos resultados mais focados no AS/RS, a Figura 6.3 demonstra quatro abordagens. O número de itens armazenados e recuperados, foi possível desenvolver uma análise por tipo, ou seja, por formato de item. Relativamente ao número de itens armazenados durante 1 hora (*‘Throughput by Type - Storage’*), no total foram 356, sendo que o item mais armazenado foi o do tipo 1. Já os itens recuperados durante o mesmo período de tempo, no total foram 145.

O gráfico *‘Content Vs Time’* apresenta a variação de itens existentes nas *racks* ao longo da simulação, onde é possível notar um aumento de itens armazenados a cerca de minutos do fim da mesma. Esta variação é calculada tendo em conta os itens que entram e que saem do AS/RS. Por fim, foi elaborada uma análise ao tempo médio de permanência de cada item, por *rack*. É possível verificar que os itens permanecem mais tempo na *rack4* do que nas restantes. Esta é a *rack* mais afastada do sistema *crossdocking*, podendo ser esta uma das justificações para que

este dado se destaque. O tempo médio de permanência é de, aproximadamente, 690 segundos.

6.1.2 Cenário 2: 6 UA sem delay entre ordens

No cenário 2 o número de UA mantém-se inalterado, porém o *delay* existente entre execução de ordens de despacho é reduzido para 0 segundos. Ou seja, neste cenário, caso uma ordem de despacho seja satisfeita, as UA podem de imediato começar a receber itens destinados à ordem de despacho seguinte na fila. Neste cenário espera-se um aumento de eficiência do sistema devido à inexistência de pausas ou atrasos propositados nos fluxos e operações.

Como, ao existir um *delay*, os itens não param de entrar no sistema, os números relativos ao armazenamentos são superiores aos esperados neste cenário. Não havendo um atraso propositado espera-se que haja uma diminuição do número de itens armazenados e, por consequência, do número de itens recuperados.

Seguindo a mesma lógica utilizada para o cenário 1, a Figura 6.4 demonstra os resultados obtidos para os gráficos direcionados para o sistema *crossdocking* e a Figura 6.5 para o AS/RS.

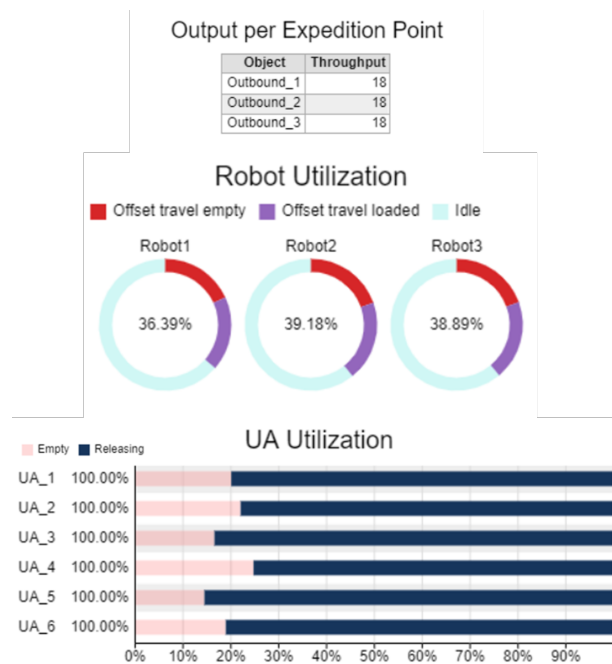


Figura 6.4: Resultados do Cenário 2 relativos ao sistema *crossdocking*

Quanto ao ‘*Output per Expedition Point*’, como esperado, o número de ordens de despacho satisfeitas aumentou, não só no total, mas também em cada saída.

Neste cenário, todas expediram o mesmo número de ordens, 18, perfazendo um total de 54. Comparativamente com o cenário 1, deu-se um aumento de 10 ordens de despacho.

O gráfico *‘Robot Utilization’* mostra que os robôs têm uma utilização média de 38,15%, ou seja, aumentaram a sua produtividade em, aproximadamente, 7% cada um, em relação ao cenário 1. Visto que já não está aplicado um *delay*, o seu tempo de funcionamento aumenta, diminuindo assim o tempo que se encontram em repouso à espera que todos os itens referentes a uma ordem de despacho cheguem à UA atribuída.

Pelo mesmo motivo, o gráfico de utilização das UA (*‘UA Utilization’*) mostra um aumento drástico da taxa de utilização das mesmas. Enquanto que para um *delay* associado de duzentos segundos a sua média de utilização ronda os 47%, se o mesmo for nulo a média de utilização das UA sofre um aumento para 80,53%.

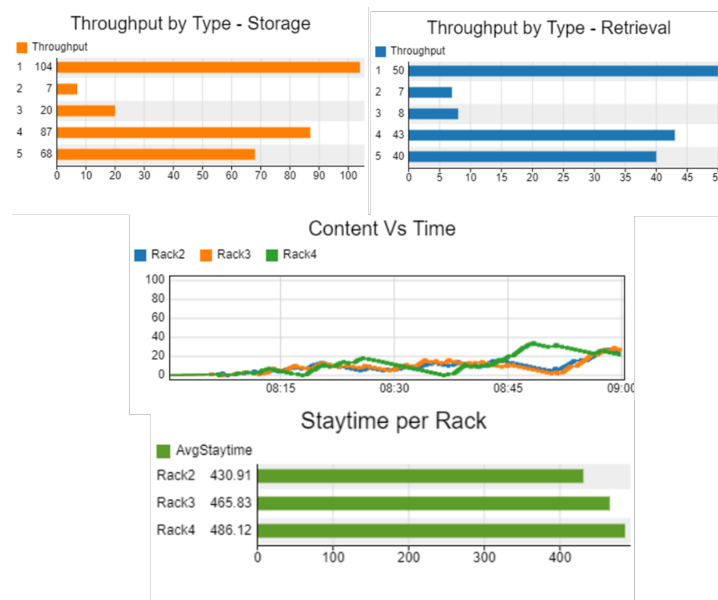


Figura 6.5: Resultados do Cenário 2 relativos ao AS/RS

O número de itens armazenados e recuperados diminuiu comparativamente com o cenário 1. Esta diminuição não foi muito significativa mas, tal como esperado, os dados apresentados são inferiores aos apresentados na Figura 6.3 do cenário 1. Como o *delay* é nulo, a distribuição de itens para as UA é realizada de forma mais rápida, impedindo alguns itens de se dirigirem para o AS/RS. Por isto, os itens que se encontram no CS são diretamente distribuídos para as UA sem passarem pela fase de armazenamento temporário.

O gráfico *‘Content Vs Time’* indica que o número de itens armazenados ao longo da hora de simulação é inferior ao que foi obtido no cenário 1. Já o tempo

de permanência em cada *rack* também sofreu um decréscimo em cerca de 200 segundos por *rack*, sendo o seu tempo médio aproximado de 461 segundos. Estas diminuições devem-se à maior utilização associada às UA e aos robôs.

6.1.3 Cenário 3: 3 UA com delay entre ordens

No que respeita ao cenário 3, o número de UA existentes no sistema, ou em funcionamento, é modificado. Espera-se um impacto significativo na eficiência do sistema. Desta forma, a UA_1, UA_3 e UA_5 mantêm-se em funcionamento, enquanto que as restantes são desconectadas. Quanto ao *delay*, fator importante no que diz respeito à eficiência do sistema, é reposto a 200 segundos. Através deste cenário, devido à possibilidade de retirar uma ou mais UA existentes no sistema, é possível avaliar a eficiência em caso de alguma falha. Na Figura 6.6 encontram-se disponíveis os resultados relacionados com o *crossdocking*.

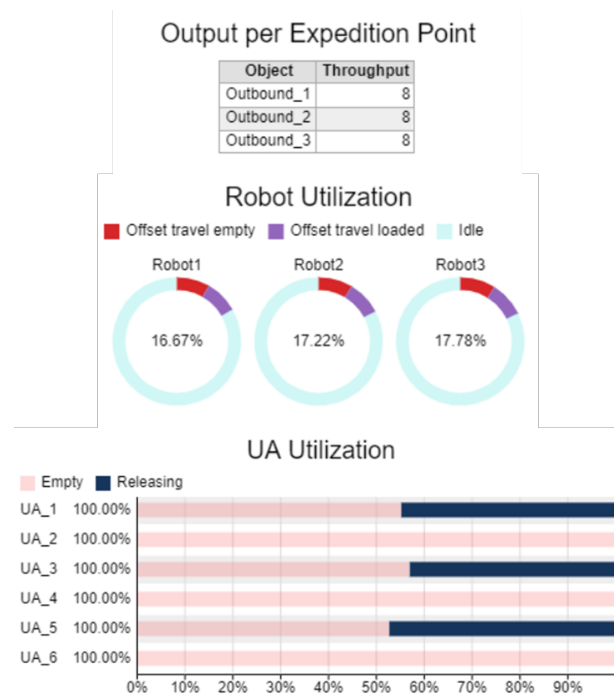


Figura 6.6: Resultados do Cenário 3 relativos ao sistema *crossdocking*

Assim, com um *delay* de 200 segundos e apenas três UA em funcionamento, é notável uma redução do número de ordens de despacho totalizadas (24). Comparativamente com o cenário 1, que também tem associado um *delay* de 200 segundos, nota-se um decréscimo de 20 unidades. Criando uma analogia, o número de UA existentes neste cenário é metade do existente no cenário 1, daí ser lógico que o número de ordens de despacho concretizadas também seja reduzido para, aproximadamente, metade.

Recorrendo ao gráfico da utilização dos robôs, ‘*Robot Utilization*’, os quais se mantêm os três em funcionamento, a sua utilização diminuiu para, em média, 17,22%.

Já a utilização das UA, visto que a UA_2, UA_4 e UA_6 não se encontram disponíveis ou em funcionamento, a sua utilização é nula. Quanto às restantes (UA_1, UA_3, UA_5), a sua utilização média ronda os 45%. Este valor é 2% inferior ao obtido no cenário 1, algo justificável por um simples fator. No cenário 1, um *robot* está encarregue de duas UA. À medida que o mesmo realiza o transporte de uma UA para o *combiner*, a UA adjacente está a receber itens até se completar a ordem de despacho. Este processo repete-se à medida que vão sendo atribuídas as ordens, o que pode influenciar a utilização das UA e justificar esta diferença de 2%.

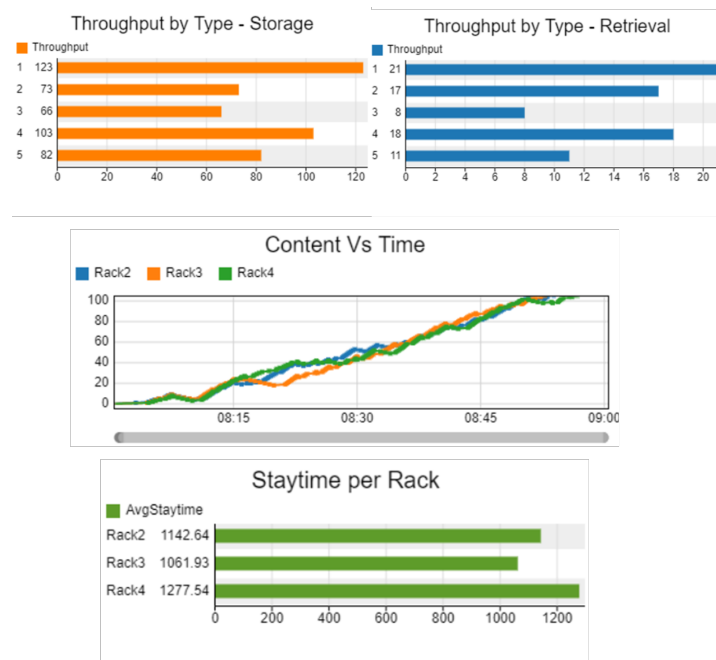


Figura 6.7: Resultados do Cenário 3 relativos ao AS/RS

Quanto ao armazém automático, o número de itens armazenados (‘*Throughput by Type - Storage*’), comparativamente com o cenário 1 e o cenário 2, alterou-se. Isto porque, reduzindo a capacidade do sistema no cumprimento de ordens de despacho em 50% e mantendo o fluxo de chegada de itens ao sistema, os mesmos apenas se distribuem neste cenário para três e não seis ordens de despacho, aumentando consideravelmente o número de itens encaminhados para o AS/RS. Analisando este gráfico, é possível perceber que, respeitando aos itens de tipo 1, o aumento não foi considerável, ao invés dos itens do tipo 2, 3 e 4. Isto é indicativo de que estes três tipos de itens são bastantes utilizados nas ordens

de despacho que o sistema recebe na primeira hora de funcionamento. Quanto ao número de itens recuperados do AS/RS, é reduzido. A lógica aplicada para os itens armazenados, para a recuperação, é invertida. Reduzindo a capacidade do sistema, menos ordens de despacho serão concretizadas no espaço temporal simulado, o que, por consequência, vai reduzir o número de itens armazenados necessários para concluir uma ordem.

Relacionado com os dados de armazenamento e recuperação de itens, o gráfico ‘*Content Vs Time*’ demonstra um número elevado de itens armazenados à medida que a simulação decorre, algo que se previa visto que o número de itens a entrar no AS/RS é superior em comparação com o cenário 1 e 2.

Aliado a este gráfico, o ‘*Staytime per Rack*’ segue o mesmo caminho, sendo os tempos de permanência elevados, de aproximadamente 1160 segundos, em média.

6.1.4 Cenário 4: 3 UA sem delay entre ordens

Neste último cenário simulado, o *delay* é nulo, tal como no cenário 2, e o número de UA em funcionamento é o mesmo que no cenário 3. A Figura 6.8 demonstra os gráficos associados ao funcionamento do sistema *crossdocking* e a Figura 6.9 ao AS/RS.

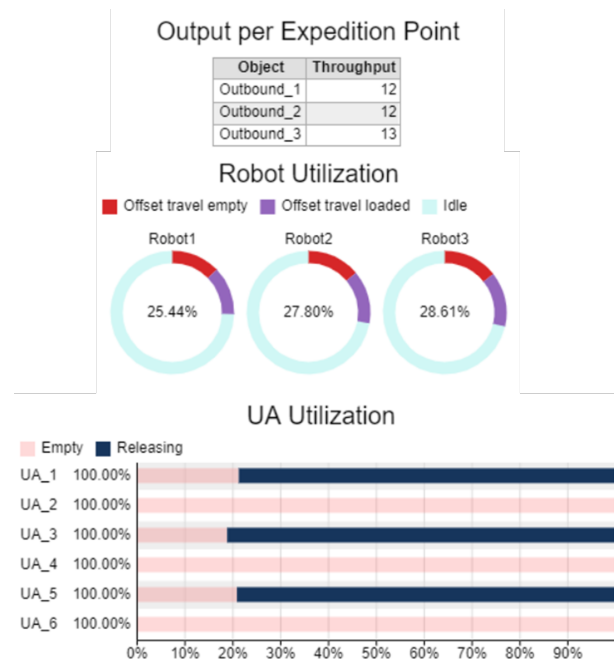


Figura 6.8: Resultados do Cenário 4 relativos ao sistema *crossdocking*

Não existindo um *delay*, o número de ordens de despacho aumenta (relativamente ao cenário 3), passando para 37 unidades expedidas. Comparativamente ao

cenário 2 (também com *delay* associado nulo), são expedidas menos 17 unidades.

A utilização dos robôs é, em média, 27% (aproximadamente). Comparativamente com o cenário 2, é uma diferença de cerca de 11%, justificável pela, mais uma vez, diferença de UA em funcionamento.

Não existindo a pausa entre atribuição de ordens de despacho (*delay*), a utilização das UA estabelece-se, em média, nos 80%.

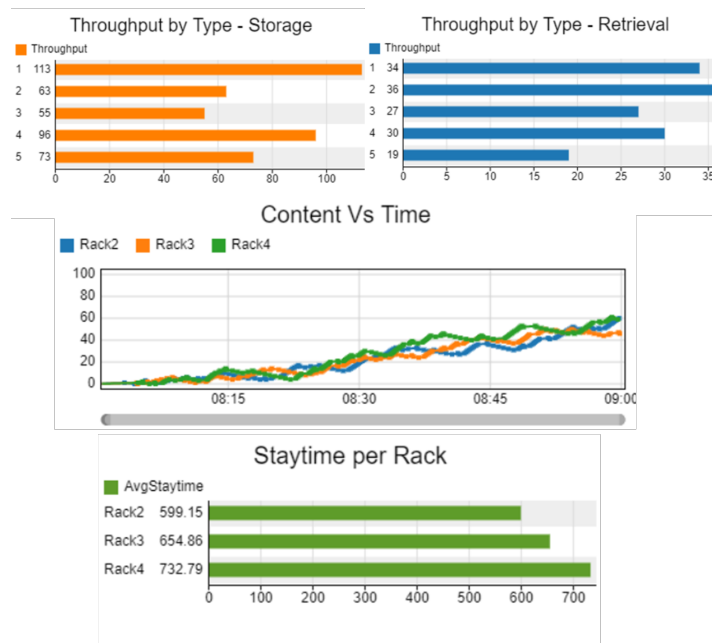


Figura 6.9: Resultados do Cenário 4 relativos ao AS/RS

Quanto ao AS/RS, o número de itens armazenados, comparativamente com o cenário 3, o número de itens armazenados é menor. Já o valor dos itens recuperados é superior pois, se não há um *delay* associado, mais ordens de despacho serão concretizadas, o que indica que o sistema terá mais necessidade de recorrer aos itens armazenados para satisfazerem uma ordem.

Tanto os gráficos '*Content Vs Time*' e '*Staytime per Rack*' apresentam valores inferiores aos gerados no cenário 3, pelas mesmas justificações apresentadas no cenário 2. O tempo médio de permanência aproximado é de 664 segundos.

6.2 Resultados finais

Os resultados obtidos devem ser comparados entre si, de forma a perceber qual o cenário mais vantajoso e eficiente, em termos de concretização de ordens de despacho, visto que esse é o objetivo principal da simulação. Quanto maior o número de ordens de despacho concretizadas, num mesmo período de tempo,

mais eficiente é considerado o sistema. Outro parâmetro de avaliação passa pelo número de itens armazenados no AS/RS. Quanto menor o número de itens armazenados mais eficiente se considera o sistema, pois não foi necessário recorrer ao armazenamento temporário de itens, o que indica que seguiram para a concretização de uma ordem de despacho. É também necessário ter em conta as diferenças entre cenários pois, em contexto real, existe sempre a probabilidade da existência de falhas associadas ao funcionamento das unidades acumuladoras. A taxa de utilização dos robôs está diretamente relacionada com o número de ordens de despacho concretizadas, tal como a taxa de utilização das UA.

Como os dois sistemas simulados se encontram integrados, o funcionamento do sistema *crossdocking* influencia o funcionamento do AS/RS. Se um número menor de itens é distribuído pelas UA, mantendo o mesmo fluxo de entrada no sistema, o AS/RS irá albergar um número maior dos mesmos, e vice-versa.

De maneira a aglomerar e resumir os resultados obtidos em cada cenário, foi desenvolvida a Tabela 6.2. Os valores apresentados encontram-se arredondados à unidade.

Tabela 6.2: Síntese dos resultados obtidos nos quatro cenários simulados

	Cenário 1	Cenário 2	Cenário 3	Cenário 4
Nº ordens expedidas	44	54	24	37
Utilização média robôs (%)	32	38	17	27
Utilização média UA (%)	47	81	45	80
Itens armazenados/recuperados	356/145	286/148	447/75	400/146
Tempo médio de permanência (s)	690	461	1160	664

Realizando uma análise à Tabela 6.2, é possível perceber que o cenário mais vantajoso e eficiente é o cenário 2. Neste cenário, o número de ordens expedidas, a taxa de utilização dos robôs e das unidades acumuladoras apresentam melhores valores comparativamente com os demais. Quanto ao número de itens armazenados e recuperados, a diferença entre ambos é a menor, o que indica que há um maior fluxo de itens a serem armazenados e recuperados simultaneamente. Criando um rácio entre itens armazenados e recuperados:

- cenário 1: $\frac{356}{145} = 2,44$
- cenário 2: $\frac{286}{148} = 1,93$
- cenário 3: $\frac{447}{75} = 5,96$

- cenário 4: $\frac{400}{146} = 2,74$

Quanto menor este rácio, mais eficiente se considera o sistema. Visto que o objetivo do AS/RS é o armazenamento temporário de produtos, então, ao longo de uma hora de simulação, quanto menor o rácio entre produtos que entram para armazenamento e que saem para concretizar ordens de despacho, mais eficiente é o sistema. Para além disso, o cenário 2 apresenta um menor número de itens armazenados, o que indica que o sistema em si, com estes parâmetros é realmente mais eficiente. O tempo de permanência associado a cada cenário varia de acordo com dois fatores em estudo: a taxa de utilização das UA e o número de itens armazenados/recuperados. Quanto maior a taxa de utilização das UA, maior o fluxo de itens do sistema o que conduz a uma maior probabilidade de recuperação de itens, diminuindo o tempo de permanência dos mesmos no AS/RS.

As comparações entre cenários podem ser feitas através de várias combinações. Como os mesmos dependem de dois fatores distintos, o *delay* e o número de UA, as analogias podem variar. Por exemplo:

- Cenário 1 *vs.* Cenário 2: o fator diferenciador nesta comparação é o *delay* existente entre a atribuição de ordens de despacho às UA. A diferença entre ter um *delay* estabelecido em 200 segundos ou nulo é bastante significativa, tendo um grande impacto no número de ordens expedidas. O cenário 2 consegue expedir mais dez ordens de despacho do que o cenário 1. Isto porque, não havendo um atraso, a taxa de utilização das UA, no cenário 2, é consideravelmente mais elevada. Havendo um maior fluxo de itens, maior é a utilização das UA, dos robôs, menor o rácio entre itens armazenados e recuperados e menor o tempo de permanência. É admissível afirmar que o *delay* aplicado no cenário 1 tem um grande impacto na eficiência do sistema.
- Cenário 1 *vs.* Cenário 3: ambos estes cenários têm um *delay* associado de 200 segundos. Diferenciados pelo número de UA, que desce para metade no cenário 3, os resultados obtidos devem-se à diminuição de capacidade do sistema. É por este motivo que tanto o número de ordens expedidas como a taxa de utilização dos robôs é significativamente menor no cenário 3. Quanto à taxa de utilização das UA é semelhante, pois em ambos os cenários o atraso associado é o mesmo, o que indica que a atribuição de ordens de despacho às UA é realizada de forma idêntica. O rácio de itens armazenados e recuperados é significativamente superior no cenário 3, pois como o número de ordens expedidas é menor, serão armazenados mais itens e recuperados menos. Esta justificação também se aplica aos dados obtidos relativos ao tempo de permanência.
- Cenário 2 *vs.* Cenário 4: sendo o *delay* nulo, a única variação ocorrida na simulação destes cenários é o número de UA em funcionamento. Devido

a este fator, já era esperado que o número de ordens expedidas, a taxa de utilização de robôs diminuísse no cenário 4. Quanto à taxa de utilização das UA, mantêm-se a par entre ambos os cenários, isto porque embora o cenário 4 possua menos três UA do que o cenário 2, a mesma mantém-se. O rácio itens armazenados/recuperados é maior no cenário 4 pois são armazenados mais itens comparativamente com o cenário 2, e recuperados, aproximadamente, os mesmos. Daí a diferença existente entre o tempo de permanência.

- Cenário 3 *vs.* Cenário 4: com o mesmo número de UA, a comparação existente entre estes cenários é realizada tendo em conta o *delay*, tal como acontece na comparação entre os cenários 1 e 2. Sendo que se espera que os resultados sofram alterações idênticas. O número de ordens expedidas aumenta no cenário 4, o qual tem um *delay* nulo. Devido a este fator, a taxa de utilização dos robôs e das UA é superior também no cenário 4. Quanto menor o rácio entre itens armazenados e recuperados, mais eficiente é considerado o sistema. O cenário 4 apresenta um rácio consideravelmente inferior ao do cenário 3 devido à necessidade de recuperação de mais itens para satisfazer as ordens de despacho. Isto leva a que o tempo de permanência de itens no AS/RS também seja menor no cenário 4.

Através destas analogias e comparações, o cenário simulado mais eficiente, quer relativamente ao sistema *crossdocking*, quer ao AS/RS, é o cenário 2. Comprovando que todos os valores obtidos para este cenário, em comparação com os restantes, são definitivamente melhores:

- maior número de unidades expedidas: 54 (cenário 2);
- maior taxa de utilização de robôs: 38% (cenário 2);
- maior taxa de utilização de UA: 81% (cenário 2, a par com o cenário 4, que teve uma taxa de utilização de UA de 80%);
- menor rácio itens armazenados/recuperados: 1,93 (cenário 2);
- menor tempo de permanência de produtos no armazém: 461 segundos (cenário 2).

Sendo o cenário 2 o mais eficiente, conclui-se que num sistema *crossdocking* com um armazém automático integrado, de acordo com os parâmetros definidos e propriedades do modelo simulado, o tempo de atribuição de uma ordem de despacho a uma unidade acumuladora, onde os itens se agrupam para serem expedidos, deve ser mínimo de modo a aumentar a eficiência. Foi comprovado que com a redução deste período de tempo (*delay*), mais ordens serão expedidas e

melhor rendimento se retira dos *Task Executors*, como é o caso dos robôs (*Robot1*, *Robot2*, *Robot3*). Quanto ao AS/RS, os dados do seu funcionamento dependem do tipo de funcionamento aplicado no sistema *crossdocking*. Isto porque, qualquer item que siga para armazenamento teve uma passagem prévia pelo CS, não tendo sido atribuído a nenhuma ordem de despacho. Ao ficar armazenado, o objetivo é que seja recuperado, após uns instantes, para satisfazer outra ordem de despacho. Quanto menor o número de unidades acumuladoras em funcionamento, menor o número de ordens de despacho expedidas, o que leva a um aumento do número de itens armazenados, aumentando também o seu tempo de permanência no AS/RS. Este fator pode ser atenuado caso a entrada de itens no fluxo diminua, impedindo assim a entrada constante de produtos no sistema.

Observando os restantes cenários, constata-se que, seguidamente ao cenário 2, o considerado mais eficiente através dos dados obtidos é o cenário 1.

6.3 Conclusão do Capítulo 6

Através dos quatro cenários distintos simulados, foi possível obter resultados do modelo simulado.

Devido ao leque de opções de avaliação do sistema disponíveis, o *delay* associado à atribuição de ordens de despacho às UA e o número de UA em funcionamento, consideraram-se os mais relevantes e com possibilidade de criação de um maior impacto nos resultados. Porém, poderiam-se ter analisado outros parâmetros, tal como uma maior variação de entrada de itens no sistema, a utilização de mais robôs, paletizadores e até cais de expedição (*outbounds*). Diminuir o número de *racks* do AS/RS, o seu espaço útil de armazenamento, ou até a capacidade de transporte de cada *ASRSVehicle*, poderiam ter efeito no bom funcionamento do sistema *crossdocking*. Diversas opções, para além das apresentadas, poderiam ter sido exploradas e obtidos outros tipos de resultados.

Desta forma, o presente capítulo revela-se de extrema importância para finalizar o estudo realizado nesta Tese/Dissertação. Através da criação dos quatro cenários abordados na Subsecção 6.1 e da explicação dos resultados obtidos em cada um deles na Subsecção 6.2, foi possível concluir com sucesso a simulação do modelo desenvolvido.

Capítulo 7

Conclusões

O presente capítulo tem como objetivo efetuar uma ponderação de todo o conteúdo estudado, desenvolvido e analisado ao longo da presente Tese/Dissertação. Numa primeira secção, serão abordados os objetivos definidos e enunciados no Capítulo 1 e feita uma análise à sua execução, retirando as conclusões relativas à mesma. Foi também elaborada uma secção para possíveis desenvolvimentos futuros, promovendo algumas ideias do que se considera o próximo passo do projeto modelado e simulado, de maneira a aprimorar ao máximo o mesmo.

7.1 Principais conclusões

Após uma reflexão geral acerca de todo o trabalho desenvolvido é possível chegar a uma conclusão global de tudo o que foi feito. Este trabalho permitiu uma análise aprofundada às tipologias, conceitos e funcionamentos de sistemas *crossdocking* e de armazéns automáticos em modo AS/RS. A integração destes temas tornou o desenvolvimento do modelo de simulação mais interessante e desafiante, simulando assim um modelo complexo de um centro logístico aplicável em contexto real.

Ao nível de sistemas *crossdocking*, foi possível estudar as tipologias existentes, as suas dinâmicas de funcionamento e de que modo estas variam consoante a alteração de dimensões físicas ou de qualquer outro tipo de propriedades. Permitiu também alargar o conhecimento em gestão de cadeias de distribuição e perceber quais os processos e operações que os produtos sofrem até ao momento em que chegam ao seu destino final.

Quanto ao AS/RS, os tipos de armazéns automáticos existentes e todas as políticas de armazenamento estudadas, contribuíram para uma implementação mais completa. A conjugação de conceitos foi fundamental, permitindo modelar

o armazém de forma mais conveniente, facilitando a troca de produtos (armazenamento e recuperação) entre o mesmo e o sistema *crossdocking*.

Ao longo do Capítulo 5 foi explicado todo o desenvolvimento do trabalho para que, no Capítulo 6, fosse possível analisar o implementado através de dados gerados com o auxílio do FlexSim. A criação de diversos cenários de atividade possibilitou comparar o funcionamento do sistema tendo em conta algumas variáveis. Foram analisados dois parâmetros fundamentais no funcionamento do sistema. O intervalo de tempo existente na atribuição de ordens de despacho (*delay*) a uma UA específica, e o número de UA em funcionamento. Com isto, foi possível concluir que o *delay* é um fator com elevado grau de importância. Quanto maior, menor eficiência terá o sistema, e maior a probabilidade de criação de um *bottleneck* no AS/RS. Os produtos entram no sistema e não são atribuídos a nenhuma ordem de despacho. Assim, logicamente, seguem para armazenamento até serem requeridos. O número de unidades acumuladoras também foi considerado um fator crítico no sistema, pois relaciona-se diretamente com o número de ordens expedidas num determinado espaço temporal. Diversos outros fatores são influenciados pela atuação do *delay* e do número de UA em funcionamento, como é o caso da taxa de utilização dos robôs. Uma maior taxa de utilização possibilita um maior número de ordens de despacho concretizadas.

Posto isto, após uma análise aos diversos dados obtidos nos diferentes cenários simulados, como esperado, o cenário 2 revelou-se o mais eficiente. Este cenário continha todas as unidades acumuladoras em funcionamento e um *delay* nulo. Com um *delay* associado de 200 segundos, o cenário 1 também é considerado eficiente, embora não tanto.

Em suma, completado o modelo de simulação, totalmente funcional, entende-se que o principal objetivo do trabalho foi atingido com sucesso. O sistema funciona sem qualquer limitação e adequa-se a novas ordens de despacho, novos fluxos de entrada de itens e possibilita o acréscimo de elementos fulcrais para o seu funcionamento, sem alterar a lógica de funcionamento utilizada. Todo o trabalho realizado foi significativo. No que diz respeito à aplicação dos conceitos explícitos no Capítulo 2 no modelo de simulação, os objetivos também foram cumpridos. Foram aplicadas políticas de armazenamento no AS/RS e diversos conceitos relacionados com *crossdocking*.

7.2 Possíveis desenvolvimentos futuros

Quanto aos possíveis futuros desenvolvimentos, no que toca ao modelo implementado, a alteração do seu formato e configurações forneceria uma perspetiva diferente do funcionamento destes sistemas. Uma análise mais aprofundada, realizando a simulação de várias configurações, geraria resultados interessantes e

passíveis de comparação, o que permitiria compreender melhor qual o tipo de sistema mais eficiente.

Para além disso, um aspeto importante identificado ao longo do desenvolvimento do modelo, que não foi aplicado, passa pela calendarização, quer da entrada de itens no sistema, quer da expedição dos mesmos, e também da chegada de ordens de despacho aos mesmos. Esta calendarização iria permitir um funcionamento mais realista, podendo estar a par com o contexto real de um determinado centro logístico com características diferentes. O sistema possuindo as horas de entrada de itens (chegada de veículos para descarga) e a hora de saída de uma ordem de despacho, já com os itens agrupados e embalados (chegada de veículos para carga) seria uma excelente adição, fornecendo um maior realismo. A calendarização de chegada de ordens de despacho seria outro ponto interessante, principalmente na análise de comportamento e desempenho do sistema AS/RS.

Para além disto, a simulação de um maior número de cenários, com outro tipo de variáveis, forneceria uma perspetiva mais íntegra do funcionamento e eficiência do sistema.

Concluindo, aproximar mais o modelo de um contexto real e a realização da simulação de mais testes e cenários, traria melhorias nos processos e permitia uma análise de todo o trabalho desenvolvido ao pormenor.

Referências Bibliográficas

- [1] U. M. Apte and S. Viswanathan, “Effective cross docking for improving distribution efficiencies,” *International Journal of Logistics Research and Applications*, vol. 3, no. 3, pp. 291–302, 2000. [cited on p. iii, 10, 11, 13, 14]
- [2] N. Boysen and M. Flidner, “Cross dock scheduling: Classification, literature review and research agenda,” *Omega*, vol. 38, no. 6, pp. 413–422, 2010. [cited on p. iii, 15, 16]
- [3] J. Van Belle, P. Valckenaers, and D. Cattrysse, “Cross-docking: State of the art,” *Omega*, vol. 40, no. 6, pp. 827–846, 2012. Special Issue on Forecasting in Management Science. [cited on p. iii, 10, 12, 16, 17, 18, 19, 20]
- [4] K. J. Roodbergen and I. F. Vis, “A survey of literature on automated storage and retrieval systems,” *European Journal of Operational Research*, vol. 194, no. 2, pp. 343–362, 2009. [cited on p. iii, 23, 24, 25, 27, 28, 31, 34, 35]
- [5] R. Manzini, M. Gamberi, and A. Regattieri, “Design and control of an as/rs,” *The International Journal of Advanced Manufacturing Technology*, vol. 28, pp. 766–774, 01 2006. [cited on p. iii, 24, 31, 32, 33, 34, 35, 36]
- [6] Roger Duke, “What is a Gantt Chart?.” <https://www.gantt.com/>, Last access: 2021. [cited on p. 4]
- [7] M. H. F. Zarandi, H. Khorshidian, and M. A. Shirazi, “A constraint programming model for the scheduling of JIT cross-docking systems with pre-emption,” *Journal of Intelligent Manufacturing*, vol. 27, pp. 297–313, April 2016. [cited on p. 10, 17]
- [8] G. Ertek, “A tutorial on crossdocking,” in *Proceedings of 3rd International Logistics & Supply Chain Congress*, (Istanbul, Turkey), pp. 1–17, November 2005. [cited on p. 11]

- [9] A.-L. Ladier and G. Alpan, “Cross-docking operations: Current research versus industry practice,” *Omega*, vol. 62, pp. 145–162, 2016. [cited on p. 12, 15, 19, 22]
- [10] Z. Li, W. He, C. Sim, and C. Chen, “A solution for cross-docking operations planning, scheduling and coordination,” in *Journal of Service Science and Management Chain Congress*, vol. 05, pp. 2957 – 2962, November 2008. [cited on p. 12]
- [11] J. III and K. Gue, “The best shape for a crossdock,” *Transportation Science*, vol. 38, pp. 235–244, 05 2004. [cited on p. 15, 16]
- [12] Fretefy, “O que é Cross Docking? .” <https://www.fretefy.com.br/blog/o-que-e-cross-docking-tudo-que-voce-precisa-saber>, Last access: April 8, 2021 2021. [cited on p. 18, 19, 20, 21]
- [13] Y. H. Lee, J. W. Jung, and K. M. Lee, “Vehicle routing scheduling for cross-docking in the supply chain,” *Comput. Ind. Eng.*, vol. 51, p. 247–256, Oct. 2006. [cited on p. 21]
- [14] J. P. van den Berg and A. Gademann, “Simulation study of an automated storage/retrieval system,” *International Journal of Production Research*, vol. 38, no. 6, pp. 1339–1356, 2000. [cited on p. 23, 32, 35]
- [15] J. Verriet, R. Hamberg, J. Caarls, and B. van Wijngaarden, “Warehouse simulation through model configuration,” in *Proceedings - 27th European Conference on Modelling and Simulation, ECMS 2013s*, vol. 05, pp. 629–635, May 2013. [cited on p. 24, 26]
- [16] X. Cai, *Performance evaluation of warehouses with automated storage and retrieval technologies*. PhD thesis, University of Louisville, 2010. [cited on p. 26]
- [17] J. Tutak and J. Wiech, “Horizontal automated storage and retrieval system,” *Advances in Science and Technology Research Journal*, vol. 11, pp. 82–95, 03 2017. [cited on p. 26, 27]
- [18] W. Hausman, L. Schwarz, and S. Graves, “Optimal storage assignment in automatic warehousing systems,” *Management Science*, vol. 22, pp. 629–638, 02 1976. [cited on p. 30, 31]
- [19] C. Kallina and J. Lynn, “Application of the Cube-Per-Order Index Rule for Stock Location in a Distribution Warehouse,” *Interfaces*, vol. 7, pp. 37–46, November 1976. [cited on p. 30]
- [20] M. J. ROSENBLATT, Y. ROLL, and D. V. ZYSER, “A combined optimization and simulation approach for designing automated storage/retrieval systems,” *IIE Transactions*, vol. 25, no. 1, pp. 40–50, 1993. [cited on p. 31]

- [21] M. J. Rosenblatt and A. Eynan, “Deriving the optimal boundaries for class-based automatic storage/retrieval systems,” *Management Science*, vol. 35, no. 12, pp. 1519–1524, 1989. [cited on p. 31]
- [22] Y. Jaghbeer, R. Hanson, and M. Johansson, “Automated order picking systems and the links between design and performance: a systematic literature review,” *International Journal of Production Research*, vol. 58, pp. 1–17, 07 2020. [cited on p. 32]
- [23] M.-H. Han, L. F. McGinnis, J. S. Shieh, and J. A. White, “On sequencing retrievals in an automated storage/retrieval system,” *IIE Transactions*, vol. 19, no. 1, pp. 56–66, 1987. [cited on p. 35]
- [24] Visual Components, “Intralogistics and Warehousing.” <https://www.visualcomponents.com/industries/intralogistics-warehousing/>, Last access: May 5, 2021 2021. [cited on p. 39, 40]
- [25] Mandlik and B. B. R, “Simulation in food processing-a review,” 2015. [cited on p. 39]
- [26] The AnyLogic Company , “Simulation Modeling Software Tools and Solutions.” <https://www.anylogic.com/>, Last access: May 5, 2021 2021. [cited on p. 40]
- [27] FlexSim Software Products, Inc, “3D Simulation Modeling and Analysis Software.” <https://www.flexsim.com/>, Last access: May 5, 2021 2021. [cited on p. 40, 50]
- [28] SIMIO LLC, “Simulation, Production Planning and Scheduling Software.” <https://www.simio.com/>, Last access: May 5, 2021 2021. [cited on p. 40]
- [29] SIMUL8, “Fast, Intuitive Simulation Software Desktop and Web.” <https://www.simul8.com/>, Last access: May 5, 2021 2021. [cited on p. 40]
- [30] GetApp, “Business Software, Reviews and Comparasions.” <https://www.getapp.com/>, Last access: May 10, 2021 2021. [cited on p. 40]
- [31] Microsoft, “Microsoft Excel.” <https://www.microsoft.com/pt-pt/microsoft-365/excel>, Last access: Jun 15, 2021 2021. [cited on p. 67]

